

Sequencer64 Developer Reference Manual

0.95.1

Generated by Doxygen 1.8.14

Contents

1	Sequencer64 Classes and Functions	1
1.1	Introduction	1
2	Licenses	3
2.1	License Terms for the This Project.	3
2.2	XPC Application License	3
2.3	XPC Library License	4
2.4	XPC Documentation License	4
2.5	XPC Affero License	5
2.6	XPC License Summary	5
3	Todo List	7
4	Deprecated List	11
5	Bug List	13
6	Namespace Index	15
6.1	Namespace List	15
7	Hierarchical Index	17
7.1	Class Hierarchy	17
8	Data Structure Index	21
8.1	Data Structures	21

9 Namespace Documentation	27
9.1 Gtk Namespace Reference	27
9.2 seq64 Namespace Reference	27
9.2.1 Detailed Description	43
9.2.2 Typedef Documentation	44
9.2.2.1 midibyte	44
9.2.2.2 midistring	44
9.2.2.3 bussbyte	44
9.2.2.4 midishort	44
9.2.2.5 midilong	44
9.2.2.6 colorbyte	44
9.2.2.7 midipulse	44
9.2.2.8 midibpm	45
9.2.2.9 rterror_callback	45
9.2.2.10 rtmidi_callback_t	45
9.2.3 Enumeration Type Documentation	45
9.2.3.1 wave_type_t	45
9.2.3.2 seq_modifier_t	46
9.2.3.3 seq_event_type_t	46
9.2.3.4 seq_scroll_direction_t	47
9.2.3.5 clock_e	47
9.2.3.6 interaction_method_t	48
9.2.3.7 mute_group_handling_t	48
9.2.3.8 c_music_scales	49
9.2.3.9 draw_type_t	49
9.2.3.10 edit_mode_t	49
9.2.3.11 loop_record_t	50
9.2.3.12 mouse_action_e	50
9.2.3.13 edit_action_t	50
9.2.3.14 rtmidi_api	51

9.2.4	Function Documentation	51
9.2.4.1	swap()	51
9.2.4.2	wave_type_name()	52
9.2.4.3	extract_timing_numbers()	52
9.2.4.4	tokenize_string()	53
9.2.4.5	pulses_to_string()	53
9.2.4.6	pulses_to_measurestring()	54
9.2.4.7	pulses_to_midi_measures()	54
9.2.4.8	pulses_to_timestring() [1/2]	55
9.2.4.9	pulses_to_timestring() [2/2]	55
9.2.4.10	measurestring_to_pulses()	56
9.2.4.11	midi_measures_to_pulses()	56
9.2.4.12	timestring_to_pulses() [1/2]	57
9.2.4.13	string_to_pulses()	57
9.2.4.14	string_to_midibyte()	57
9.2.4.15	shorten_file_spec()	58
9.2.4.16	string_not_void()	58
9.2.4.17	string_is_void()	59
9.2.4.18	strings_match()	59
9.2.4.19	log2_time_sig_value()	60
9.2.4.20	zoom_power_of_2()	60
9.2.4.21	beat_pow2()	60
9.2.4.22	beat_log2()	61
9.2.4.23	tempo_us_from_bytes()	61
9.2.4.24	tempo_us_to_bytes()	61
9.2.4.25	tempo_to_note_value()	62
9.2.4.26	note_value_to_tempo()	62
9.2.4.27	ppqn_is_valid()	63
9.2.4.28	tempo_us_from_bpm()	63
9.2.4.29	bpm_from_tempo_us()	64

9.2.4.30 bpm_from_bytes()	64
9.2.4.31 pulse_length_us()	64
9.2.4.32 delta_time_us_to_ticks()	65
9.2.4.33 ticks_to_delta_time_us()	65
9.2.4.34 clock_tick_duration_bogus()	66
9.2.4.35 clock_ticks_from_ppqn()	67
9.2.4.36 double_ticks_from_ppqn()	67
9.2.4.37 pulses_per_measure()	67
9.2.4.38 measures_to_ticks()	67
9.2.4.39 ticks_to_measures()	68
9.2.4.40 pulse_divide()	69
9.2.4.41 wave_func()	69
9.2.4.42 extract_port_names()	70
9.2.4.43 extract_bus_name()	70
9.2.4.44 extract_port_name()	71
9.2.4.45 current_date_time()	71
9.2.4.46 help_check()	71
9.2.4.47 parse_options_files()	72
9.2.4.48 parse_mute_groups()	72
9.2.4.49 parse_o_options()	73
9.2.4.50 parse_log_option()	73
9.2.4.51 parse_command_line_options()	74
9.2.4.52 write_options_files()	74
9.2.4.53 build_details()	74
9.2.4.54 check_daemonize()	75
9.2.4.55 daemonize()	75
9.2.4.56 undaemonize()	75
9.2.4.57 reroute_stdio()	75
9.2.4.58 is_note_off_velocity()	76
9.2.4.59 to_string()	76

9.2.4.60	create_tempo_event()	76
9.2.4.61	file_access()	76
9.2.4.62	file_exists()	77
9.2.4.63	file_readable()	77
9.2.4.64	file_writable()	77
9.2.4.65	file_accessible()	77
9.2.4.66	file_executable()	78
9.2.4.67	file_is_directory()	78
9.2.4.68	make_directory()	78
9.2.4.69	get_current_directory()	79
9.2.4.70	get_full_path()	79
9.2.4.71	normalize_path()	79
9.2.4.72	strip_quotes()	80
9.2.4.73	file_extension()	80
9.2.4.74	strcasecompare()	81
9.2.4.75	file_extension_match()	81
9.2.4.76	jack_shutdown_callback()	81
9.2.4.77	jack_timebase_callback()	81
9.2.4.78	jack_transport_callback()	83
9.2.4.79	create_jack_client()	83
9.2.4.80	show_jack_statuses()	84
9.2.4.81	get_current_jack_position()	85
9.2.4.82	jack_session_callback()	85
9.2.4.83	invalid_key()	86
9.2.4.84	keyval_normalize()	86
9.2.4.85	create_lash_driver()	86
9.2.4.86	lash_driver()	87
9.2.4.87	delete_lash_driver()	87
9.2.4.88	millisleep()	87
9.2.4.89	is_null_midipulse()	87

9.2.4.90	output_thread_func()	87
9.2.4.91	input_thread_func()	88
9.2.4.92	rc()	88
9.2.4.93	usr()	88
9.2.4.94	choose_ppqn()	89
9.2.4.95	timestring_to_pulses() [2/2]	89
9.2.4.96	get_compound_option()	89
9.2.4.97	set_current_directory()	90
9.2.4.98	message_concatenate()	90
9.2.4.99	info_message()	91
9.2.4.100	error_message()	91
9.2.4.101	casecompare()	91
9.2.4.102	jack_dummy_callback()	92
9.2.4.103	seq_app_name()	92
9.2.4.104	seq_client_name()	92
9.2.4.105	seq_version()	92
9.2.4.106	make_section_name()	92
9.2.4.107	font_render()	93
9.2.4.108	adjustment_dummy()	93
9.2.4.109	is_ctrl_key() [1/3]	93
9.2.4.110	is_shift_key() [1/3]	94
9.2.4.111	is_no_modifier()	94
9.2.4.112	is_ctrl_key() [2/3]	94
9.2.4.113	is_shift_key() [2/3]	95
9.2.4.114	is_ctrl_key() [3/3]	95
9.2.4.115	is_shift_key() [3/3]	95
9.2.4.116	is_ctrl_shift_key()	96
9.2.4.117	is_super_key()	96
9.2.4.118	test_widget_click()	96
9.2.4.119	is_left_drag()	97

9.2.4.120	is_drag_motion()	97
9.2.4.121	keyval_name()	97
9.2.4.122	update_mainwid_sequences()	98
9.2.4.123	update_perfedit_sequences()	98
9.2.4.124	FF_RW_timeout()	98
9.2.4.125	clamp()	98
9.2.4.126	silence_jack_errors()	99
9.2.4.127	silence_jack_info()	99
9.2.4.128	midi_api_name()	99
9.2.4.129	midi_probe()	99
9.2.4.130	midi_input_test()	99
9.2.5	Variable Documentation	99
9.2.5.1	c_controller_names	99
9.2.5.2	EVENT_STATUS_BIT	100
9.2.5.3	EVENT_ANY	100
9.2.5.4	EVENT_NOTE_OFF	100
9.2.5.5	EVENT_NOTE_ON	100
9.2.5.6	EVENT_AFTERTOUCH	100
9.2.5.7	EVENT_CONTROL_CHANGE	100
9.2.5.8	EVENT_PROGRAM_CHANGE	100
9.2.5.9	EVENT_CHANNEL_PRESSURE	101
9.2.5.10	EVENT_PITCH_WHEEL	101
9.2.5.11	EVENT_CTRL_VOLUME	101
9.2.5.12	EVENT_CTRL_BALANCE	101
9.2.5.13	EVENT_CTRL_PAN	101
9.2.5.14	EVENT_CTRL_EXPRESSION	101
9.2.5.15	EVENT_MIDI_REALTIME	101
9.2.5.16	EVENT_MIDI_SYSEX	102
9.2.5.17	EVENT_MIDI_QUARTER_FRAME	102
9.2.5.18	EVENT_MIDI_SONG_POS	102

9.2.5.19	EVENT_MIDI_SONG_SELECT	102
9.2.5.20	EVENT_MIDI_SONG_F4	102
9.2.5.21	EVENT_MIDI_SONG_F5	102
9.2.5.22	EVENT_MIDI_TUNE_SELECT	102
9.2.5.23	EVENT_MIDI_SYSEX_END	102
9.2.5.24	EVENT_MIDI_SYSEX_CONTINUE	103
9.2.5.25	EVENT_MIDI_CLOCK	103
9.2.5.26	EVENT_MIDI_SONG_F9	103
9.2.5.27	EVENT_MIDI_START	103
9.2.5.28	EVENT_MIDI_CONTINUE	103
9.2.5.29	EVENT_MIDI_STOP	103
9.2.5.30	EVENT_MIDI_SONG_FD	103
9.2.5.31	EVENT_MIDI_ACTIVE_SENSE	103
9.2.5.32	EVENT_MIDI_RESET	104
9.2.5.33	EVENT_MIDI_META	104
9.2.5.34	EVENT_META_SEQ_NUMBER	104
9.2.5.35	EVENT_META_TEXT_EVENT	104
9.2.5.36	EVENT_META_COPYRIGHT	104
9.2.5.37	EVENT_META_TRACK_NAME	104
9.2.5.38	EVENT_META_INSTRUMENT	104
9.2.5.39	EVENT_META_LYRIC	105
9.2.5.40	EVENT_META_MARKER	105
9.2.5.41	EVENT_META_CUE_POINT	105
9.2.5.42	EVENT_META_MIDI_CHANNEL	105
9.2.5.43	EVENT_META_MIDI_PORT	105
9.2.5.44	EVENT_META_END_OF_TRACK	105
9.2.5.45	EVENT_META_SET_TEMPO	105
9.2.5.46	EVENT_META_SMPTE_OFFSET	105
9.2.5.47	EVENT_META_TIME_SIGNATURE	106
9.2.5.48	EVENT_META_KEY_SIGNATURE	106

9.2.5.49	EVENT_META_SEQSPEC	106
9.2.5.50	EVENT_META_ILLEGAL	106
9.2.5.51	EVENT_NULL_CHANNEL	106
9.2.5.52	EVENT_GET_CHAN_MASK	106
9.2.5.53	EVENT_CLEAR_CHAN_MASK	106
9.2.5.54	EVENTS_ALL	107
9.2.5.55	EVENTS_UNSELECTED	107
9.2.5.56	c_midibus_output_size	107
9.2.5.57	c_midibus_input_size	107
9.2.5.58	c_midibus_sysex_chunk	107
9.2.5.59	c_midibus	108
9.2.5.60	c_midich	109
9.2.5.61	c_midiclocks	109
9.2.5.62	c_triggers	109
9.2.5.63	c_notes	109
9.2.5.64	c_timesig	109
9.2.5.65	c_bpmtag	109
9.2.5.66	c_triggers_new	110
9.2.5.67	c_mutegroups	110
9.2.5.68	c_gap_A	110
9.2.5.69	c_gap_B	110
9.2.5.70	c_gap_C	110
9.2.5.71	c_gap_D	110
9.2.5.72	c_gap_E	110
9.2.5.73	c_gap_F	110
9.2.5.74	c_midictrl	111
9.2.5.75	c_musickey	111
9.2.5.76	c_musicscale	111
9.2.5.77	c_backsequence	111
9.2.5.78	c_transpose	111

9.2.5.79	c_perf_bp_mes	111
9.2.5.80	c_perf_bw	111
9.2.5.81	c_tempo_map	112
9.2.5.82	c_reserved_1	112
9.2.5.83	c_reserved_2	112
9.2.5.84	c_tempo_track	112
9.2.5.85	c_seq_color	112
9.2.5.86	c_seq_edit_mode	112
9.2.5.87	c_midi_track_ctrl	113
9.2.5.88	c_midi_control_bpm_up	113
9.2.5.89	c_midi_control_bpm_dn	113
9.2.5.90	c_midi_control_ss_up	113
9.2.5.91	c_midi_control_ss_dn	113
9.2.5.92	c_midi_control_mod_replace	113
9.2.5.93	c_midi_control_mod_snapshot	114
9.2.5.94	c_midi_control_mod_queue	114
9.2.5.95	c_midi_control_mod_gmute	114
9.2.5.96	c_midi_control_mod_glearn	114
9.2.5.97	c_midi_control_play_ss	114
9.2.5.98	c_midi_controls	114
9.2.5.99	c_midi_control_playback	114
9.2.5.100	c_midi_control_song_record	114
9.2.5.101	c_midi_control_solo	115
9.2.5.102	c_midi_control_thru	115
9.2.5.103	c_midi_control_bpm_page_up	115
9.2.5.104	c_midi_control_bpm_page_dn	115
9.2.5.105	c_midi_control_ss_set	115
9.2.5.106	c_midi_control_record	115
9.2.5.107	c_midi_control_quan_record	115
9.2.5.108	c_midi_control_reset_seq	115

9.2.5.109 c_midi_controls_extended	116
9.2.5.110 g_midi_control_limit	116
9.2.5.111 c_status_replace	116
9.2.5.112 c_status_snapshot	116
9.2.5.113 c_status_queue	117
9.2.5.114 c_status_oneshot	117
9.2.5.115 c_scales_policy	117
9.2.5.116 c_scales_transpose_up	118
9.2.5.117 c_scales_transpose_dn	118
9.2.5.118 c_scales_text	119
9.2.5.119 c_key_text	119
9.2.5.120 c_interval_text	119
9.2.5.121 c_chord_text	119
9.2.5.122 c_chord_number	120
9.2.5.123 c_chord_table_text	120
9.2.5.124 c_chord_size	120
9.2.5.125 c_chord_table	120
9.2.5.126 c_max_instruments	120
9.2.5.127 c_max_busses	120
9.2.5.128 versiontext	120
9.2.5.129 long_options	121
9.2.5.130 s_arg_list	121
9.2.5.131 s_help_1a	121
9.2.5.132 s_help_1b	121
9.2.5.133 s_help_2	121
9.2.5.134 s_help_3	121
9.2.5.135 s_help_4	122
9.2.5.136 s_help_5	122
9.2.5.137 s_bitness	122
9.2.5.138 s_status_pairs	122
9.2.5.139 s_character_mapping	122
9.2.5.140 s_global_lash_driver	122
9.2.5.141 g_rc_settings	122
9.2.5.142 g_user_settings	123
9.2.5.143 s_jitter_amount	123
9.2.5.144 gs_mainwid_pointer	123
9.2.5.145 gs_perfedit_pointer_0	123
9.2.5.146 gs_perfedit_pointer_1	123
9.2.5.147 s_handlesize	123

10 Data Structure Documentation	125
10.1 seq64::automutex Class Reference	125
10.1.1 Detailed Description	125
10.1.2 Constructor & Destructor Documentation	125
10.1.2.1 automutex() [1/3]	126
10.1.2.2 automutex() [2/3]	126
10.1.2.3 automutex() [3/3]	126
10.1.2.4 ~automutex()	126
10.1.3 Member Function Documentation	126
10.1.3.1 operator=()	126
10.1.4 Field Documentation	126
10.1.4.1 m_safety_mutex	126
10.2 seq64::busarray Class Reference	127
10.2.1 Constructor & Destructor Documentation	128
10.2.1.1 busarray()	128
10.2.1.2 ~busarray()	128
10.2.2 Member Function Documentation	128
10.2.2.1 add() [1/2]	128
10.2.2.2 add() [2/2]	129
10.2.2.3 initialize()	129
10.2.2.4 count()	130
10.2.2.5 bus()	130
10.2.2.6 start()	130
10.2.2.7 stop()	130
10.2.2.8 continue_from()	130
10.2.2.9 init_clock()	130
10.2.2.10 clock()	131
10.2.2.11 sysex()	131
10.2.2.12 play()	131
10.2.2.13 set_clock()	131

10.2.2.14	set_all_clocks()	132
10.2.2.15	get_clock()	132
10.2.2.16	get_midi_bus_name()	132
10.2.2.17	print()	133
10.2.2.18	port_exit()	133
10.2.2.19	set_input()	133
10.2.2.20	set_all_inputs()	134
10.2.2.21	get_input()	134
10.2.2.22	is_system_port()	134
10.2.2.23	poll_for_midi()	135
10.2.2.24	get_midi_event()	135
10.2.2.25	replacement_port()	135
10.2.3	Field Documentation	136
10.2.3.1	m_container	136
10.3	seq64::businfo Class Reference	136
10.3.1	Detailed Description	137
10.3.2	Constructor & Destructor Documentation	137
10.3.2.1	businfo() [1/3]	138
10.3.2.2	businfo() [2/3]	138
10.3.2.3	businfo() [3/3]	138
10.3.2.4	~businfo()	138
10.3.3	Member Function Documentation	138
10.3.3.1	remove()	139
10.3.3.2	bus() [1/3]	139
10.3.3.3	bus() [2/3]	139
10.3.3.4	active()	139
10.3.3.5	initialize()	139
10.3.3.6	initialized()	140
10.3.3.7	init_clock() [1/3]	140
10.3.3.8	init_input() [1/2]	140

10.3.3.9	bus() [3/3]	140
10.3.3.10	activate()	140
10.3.3.11	deactivate()	140
10.3.3.12	init_clock() [2/3]	140
10.3.3.13	init_input() [2/2]	141
10.3.3.14	start()	141
10.3.3.15	stop()	141
10.3.3.16	continue_from()	141
10.3.3.17	init_clock() [3/3]	141
10.3.3.18	clock()	141
10.3.3.19	sysex()	141
10.3.3.20	print()	142
10.3.4	Friends And Related Function Documentation	142
10.3.4.1	busarray	142
10.3.5	Field Documentation	142
10.3.5.1	m_bus	142
10.3.5.2	m_active	142
10.3.5.3	m_initialized	142
10.3.5.4	m_init_clock	142
10.3.5.5	m_init_input	142
10.4	seq64::click Class Reference	143
10.4.1	Detailed Description	144
10.4.2	Constructor & Destructor Documentation	144
10.4.2.1	click() [1/3]	144
10.4.2.2	click() [2/3]	144
10.4.2.3	click() [3/3]	144
10.4.3	Member Function Documentation	145
10.4.3.1	operator=()	145
10.4.3.2	is_press()	145
10.4.3.3	is_left()	145

10.4.3.4	is_middle()	145
10.4.3.5	is_right()	145
10.4.3.6	x()	146
10.4.3.7	y()	146
10.4.3.8	button()	146
10.4.3.9	modifier()	146
10.4.3.10	mod_control()	146
10.4.3.11	mod_control_shift()	146
10.4.3.12	mod_super()	146
10.4.4	Field Documentation	146
10.4.4.1	m_is_press	147
10.4.4.2	m_x	147
10.4.4.3	m_y	147
10.4.4.4	m_button	147
10.4.4.5	m_modifier	147
10.5	seq64::condition_var Class Reference	147
10.5.1	Detailed Description	148
10.5.2	Constructor & Destructor Documentation	149
10.5.2.1	condition_var()	149
10.5.3	Member Function Documentation	149
10.5.3.1	wait()	149
10.5.3.2	signal()	149
10.5.4	Field Documentation	149
10.5.4.1	sm_cond	149
10.5.4.2	m_cond	149
10.6	seq64::configfile Class Reference	150
10.6.1	Constructor & Destructor Documentation	151
10.6.1.1	configfile()	151
10.6.1.2	~configfile()	151
10.6.2	Member Function Documentation	151

10.6.2.1	next_data_line()	152
10.6.2.2	line_after()	152
10.6.2.3	at_section_start()	152
10.6.2.4	parse()	153
10.6.2.5	write()	153
10.6.2.6	get_error_message()	153
10.6.2.7	set_error_message()	153
10.6.3	Field Documentation	153
10.6.3.1	m_error_message	153
10.6.3.2	m_name	154
10.6.3.3	m_d	154
10.6.3.4	m_line	154
10.7	seq64::editable_event Class Reference	154
10.7.1	Detailed Description	158
10.7.2	Member Enumeration Documentation	158
10.7.2.1	category_t	158
10.7.2.2	timestamp_format_t	159
10.7.3	Constructor & Destructor Documentation	159
10.7.3.1	editable_event() [1/4]	159
10.7.3.2	editable_event() [2/4]	160
10.7.3.3	editable_event() [3/4]	160
10.7.3.4	editable_event() [4/4]	160
10.7.3.5	~editable_event()	160
10.7.4	Member Function Documentation	161
10.7.4.1	value_to_name()	161
10.7.4.2	name_to_value()	161
10.7.4.3	meta_event_length()	161
10.7.4.4	operator=()	162
10.7.4.5	parent()	162
10.7.4.6	category() [1/3]	162

10.7.4.7	category() [2/3]	162
10.7.4.8	category_string()	162
10.7.4.9	category() [3/3]	163
10.7.4.10	timestamp_string()	163
10.7.4.11	timestamp() [1/3]	163
10.7.4.12	timestamp() [2/3]	163
10.7.4.13	timestamp() [3/3]	163
10.7.4.14	time_as_pulses()	164
10.7.4.15	time_as_measures()	164
10.7.4.16	time_as_minutes()	164
10.7.4.17	set_status_from_string()	164
10.7.4.18	format_timestamp()	165
10.7.4.19	stock_event_string()	165
10.7.4.20	ex_data_string()	165
10.7.4.21	status_string()	165
10.7.4.22	meta_string()	165
10.7.4.23	seqspec_string()	166
10.7.4.24	channel_string()	166
10.7.4.25	data_string()	166
10.7.4.26	analyze()	166
10.7.5	Field Documentation	167
10.7.5.1	sm_category_names	167
10.7.5.2	sm_channel_event_names	167
10.7.5.3	sm_system_event_names	167
10.7.5.4	sm_meta_event_names	167
10.7.5.5	sm_prop_event_names	167
10.7.5.6	sm_category_arrays	168
10.7.5.7	sm_meta_lengths	168
10.7.5.8	m_parent	168
10.7.5.9	m_category	168

10.7.5.10 m_name_category	168
10.7.5.11 m_format_timestamp	168
10.7.5.12 m_name_timestamp	168
10.7.5.13 m_name_status	169
10.7.5.14 m_name_meta	169
10.7.5.15 m_name_seqspect	169
10.7.5.16 m_name_channel	169
10.7.5.17 m_name_data	169
10.8 seq64::editable_events Class Reference	169
10.8.1 Member Typedef Documentation	171
10.8.1.1 Key	172
10.8.1.2 EventsPair	172
10.8.1.3 Events	172
10.8.1.4 iterator	172
10.8.1.5 const_iterator	172
10.8.1.6 reverse_iterator	172
10.8.1.7 const_reverse_iterator	172
10.8.2 Constructor & Destructor Documentation	172
10.8.2.1 editable_events() [1/3]	173
10.8.2.2 editable_events() [2/3]	173
10.8.2.3 editable_events() [3/3]	174
10.8.2.4 ~editable_events()	174
10.8.3 Member Function Documentation	174
10.8.3.1 operator=()	174
10.8.3.2 timing()	175
10.8.3.3 string_to_pulses()	175
10.8.3.4 load_events()	175
10.8.3.5 save_events()	175
10.8.3.6 events()	175
10.8.3.7 begin() [1/2]	176

10.8.3.8	begin() [2/2]	176
10.8.3.9	end() [1/2]	176
10.8.3.10	end() [2/2]	176
10.8.3.11	dref() [1/2]	176
10.8.3.12	dref() [2/2]	176
10.8.3.13	count()	177
10.8.3.14	get_length()	177
10.8.3.15	add() [1/2]	177
10.8.3.16	add() [2/2]	177
10.8.3.17	replace()	178
10.8.3.18	remove()	178
10.8.3.19	clear()	178
10.8.3.20	sort()	178
10.8.3.21	current_event() [1/2]	178
10.8.3.22	is_valid_iterator()	178
10.8.3.23	print()	178
10.8.3.24	current_event() [2/2]	178
10.8.4	Friends And Related Function Documentation	179
10.8.4.1	eventslots	179
10.8.5	Field Documentation	179
10.8.5.1	m_events	179
10.8.5.2	m_current_event	179
10.8.5.3	m_sequence	179
10.8.5.4	m_midi_parameters	179
10.9	seq64::event Class Reference	180
10.9.1	Detailed Description	184
10.9.2	Member Typedef Documentation	185
10.9.2.1	SysexContainer	185
10.9.3	Constructor & Destructor Documentation	185
10.9.3.1	event() [1/2]	185

10.9.3.2	event() [2/2]	185
10.9.3.3	~event()	185
10.9.4	Member Function Documentation	186
10.9.4.1	operator=()	186
10.9.4.2	operator<()	186
10.9.4.3	set_timestamp()	187
10.9.4.4	get_timestamp()	187
10.9.4.5	get_channel()	187
10.9.4.6	check_channel()	187
10.9.4.7	is_channel_msg()	188
10.9.4.8	is_one_byte_msg()	188
10.9.4.9	is_two_byte_msg()	188
10.9.4.10	is_sysex_msg()	189
10.9.4.11	is_note_msg()	189
10.9.4.12	is_strict_note_msg()	189
10.9.4.13	is_desired_cc_or_not_cc()	190
10.9.4.14	mod_timestamp()	190
10.9.4.15	set_status() [1/2]	191
10.9.4.16	set_status() [2/2]	191
10.9.4.17	set_meta_status()	192
10.9.4.18	set_status_keep_channel()	192
10.9.4.19	set_channel()	192
10.9.4.20	get_status()	193
10.9.4.21	non_cc_match()	193
10.9.4.22	cc_match()	193
10.9.4.23	set_data() [1/2]	193
10.9.4.24	set_data() [2/2]	194
10.9.4.25	get_data() [1/2]	194
10.9.4.26	get_data() [2/2]	194
10.9.4.27	increment_data1()	194

10.9.4.28 decrement_data1()	194
10.9.4.29 increment_data2()	195
10.9.4.30 decrement_data2()	195
10.9.4.31 append_sysex() [1/2]	195
10.9.4.32 append_sysex() [2/2]	195
10.9.4.33 append_meta_data() [1/2]	195
10.9.4.34 append_meta_data() [2/2]	196
10.9.4.35 restart_sysex()	196
10.9.4.36 set_sysex()	197
10.9.4.37 get_sysex() [1/2]	197
10.9.4.38 get_sysex() [2/2]	197
10.9.4.39 set_sysex_size()	197
10.9.4.40 get_sysex_size()	197
10.9.4.41 link()	197
10.9.4.42 get_linked()	198
10.9.4.43 is_linked()	198
10.9.4.44 clear_link()	198
10.9.4.45 paint()	198
10.9.4.46 unpaint()	198
10.9.4.47 is_painted()	198
10.9.4.48 mark()	199
10.9.4.49 unmark()	199
10.9.4.50 is_marked()	199
10.9.4.51 select()	199
10.9.4.52 unselect()	199
10.9.4.53 is_selected()	199
10.9.4.54 make_clock()	199
10.9.4.55 data()	199
10.9.4.56 get_note()	199
10.9.4.57 set_note()	199

10.9.4.58 transpose_note()	200
10.9.4.59 get_note_velocity()	200
10.9.4.60 set_note_velocity()	200
10.9.4.61 is_note_on()	200
10.9.4.62 is_note_off()	201
10.9.4.63 is_note()	201
10.9.4.64 is_note_off_recorded()	201
10.9.4.65 adjust_note_off()	201
10.9.4.66 is_one_byte()	202
10.9.4.67 is_two_bytes()	202
10.9.4.68 is_sysex()	202
10.9.4.69 is_meta()	202
10.9.4.70 is_ex_data()	202
10.9.4.71 is_tempo()	202
10.9.4.72 tempo()	202
10.9.4.73 set_tempo()	203
10.9.4.74 is_time_signature()	203
10.9.4.75 is_key_signature()	203
10.9.4.76 print()	203
10.9.4.77 get_rank()	203
10.9.5 Field Documentation	203
10.9.5.1 m_timestamp	204
10.9.5.2 m_status	204
10.9.5.3 m_channel	204
10.9.5.4 m_data	204
10.9.5.5 m_sysex	204
10.9.5.6 m_linked	204
10.9.5.7 m_has_link	205
10.9.5.8 m_selected	205
10.9.5.9 m_marked	205

10.9.5.10 m_painted	205
10.10seq64::event_list::event_key Class Reference	205
10.10.1 Detailed Description	205
10.10.2 Constructor & Destructor Documentation	206
10.10.2.1 event_key() [1/2]	206
10.10.2.2 event_key() [2/2]	206
10.10.3 Member Function Documentation	206
10.10.3.1 operator<()	206
10.10.4 Field Documentation	207
10.10.4.1 m_timestamp	207
10.10.4.2 m_rank	207
10.11seq64::event_list Class Reference	207
10.11.1 Detailed Description	210
10.11.2 Member Typedef Documentation	210
10.11.2.1 Events	210
10.11.2.2 EventsPair	210
10.11.2.3 iterator	210
10.11.2.4 const_iterator	210
10.11.2.5 reverse_iterator	210
10.11.2.6 const_reverse_iterator	211
10.11.3 Constructor & Destructor Documentation	211
10.11.3.1 event_list() [1/2]	211
10.11.3.2 event_list() [2/2]	211
10.11.3.3 ~event_list()	211
10.11.4 Member Function Documentation	211
10.11.4.1 operator=()	211
10.11.4.2 begin() [1/2]	212
10.11.4.3 begin() [2/2]	212
10.11.4.4 end() [1/2]	212
10.11.4.5 end() [2/2]	212

10.11.4.6 count()	212
10.11.4.7 get_length()	212
10.11.4.8 empty()	212
10.11.4.9 add()	212
10.11.4.10 append()	213
10.11.4.11 push_back()	213
10.11.4.12 is_modified()	214
10.11.4.13 has_tempo()	214
10.11.4.14 has_time_signature()	214
10.11.4.15 unmodify()	214
10.11.4.16 remove()	214
10.11.4.17 clear()	214
10.11.4.18 merge()	215
10.11.4.19 sort()	215
10.11.4.20 dref() ^[1/2]	215
10.11.4.21 dref() ^[2/2]	216
10.11.4.22 link_new()	216
10.11.4.23 clear_links()	216
10.11.4.24 verify_and_link()	216
10.11.4.25 link_tempos()	216
10.11.4.26 clear_tempo_links()	217
10.11.4.27 mark_selected()	217
10.11.4.28 mark_out_of_range()	217
10.11.4.29 mark_all()	217
10.11.4.30 unmark_all()	218
10.11.4.31 remove_marked()	218
10.11.4.32 unpaint_all()	218
10.11.4.33 count_selected_notes()	218
10.11.4.34 any_selected_notes()	218
10.11.4.35 count_selected_events()	218

10.11.4.36	select_all()	219
10.11.4.37	unselect_all()	219
10.11.4.38	print()	219
10.11.4.39	events()	219
10.11.5	Friends And Related Function Documentation	219
10.11.5.1	editable_events	219
10.11.5.2	midifile	220
10.11.5.3	sequence	220
10.11.6	Field Documentation	220
10.11.6.1	m_events	220
10.11.6.2	m_is_modified	220
10.11.6.3	m_has_tempo	220
10.11.6.4	m_has_time_signature	220
10.12	seq64::eventedit Class Reference	221
10.12.1	Constructor & Destructor Documentation	224
10.12.1.1	eventedit()	225
10.12.1.2	~eventedit()	226
10.12.2	Member Function Documentation	226
10.12.2.1	enqueue_draw()	226
10.12.2.2	set_seq_title()	226
10.12.2.3	set_seq_time_sig()	226
10.12.2.4	set_seq_ppqn()	227
10.12.2.5	set_seq_count()	227
10.12.2.6	set_seq_length()	227
10.12.2.7	set_event_category()	227
10.12.2.8	set_event_timestamp()	227
10.12.2.9	set_event_name()	228
10.12.2.10	set_event_data_0()	228
10.12.2.11	set_event_data_1()	228
10.12.2.12	perf_modify()	228

10.12.2.13	set_dirty()	228
10.12.2.14	v_adjustment() [1/2]	229
10.12.2.15	v_adjustment() [2/2]	229
10.12.2.16	change_focus()	229
10.12.2.17	close_out()	230
10.12.2.18	handle_close()	230
10.12.2.19	handle_delete()	230
10.12.2.20	handle_insert()	230
10.12.2.21	handle_modify()	230
10.12.2.22	handle_save()	231
10.12.2.23	handle_cancel()	231
10.12.2.24	on_realize()	231
10.12.2.25	on_set_focus()	231
10.12.2.26	on_focus_in_event()	231
10.12.2.27	on_focus_out_event()	232
10.12.2.28	on_key_press_event()	232
10.12.2.29	on_delete_event()	232
10.12.3	Friends And Related Function Documentation	232
10.12.3.1	eventslots	233
10.12.4	Field Documentation	233
10.12.4.1	m_table	233
10.12.4.2	m_vadjust	233
10.12.4.3	m_vscroll	233
10.12.4.4	m_eventslots	233
10.12.4.5	m_htopbox	233
10.12.4.6	m_showbox	233
10.12.4.7	m_editbox	234
10.12.4.8	m_optsbox	234
10.12.4.9	m_bottbox	234
10.12.4.10	m_rightbox	234

10.12.4.11m_button_del	234
10.12.4.12m_button_ins	234
10.12.4.13m_button_modify	234
10.12.4.14m_button_save	234
10.12.4.15m_button_cancel	235
10.12.4.16m_label_seq_name	235
10.12.4.17m_label_time_sig	235
10.12.4.18m_label_ppqn	235
10.12.4.19m_label_channel	235
10.12.4.20m_label_ev_count	235
10.12.4.21m_label_seq_length	235
10.12.4.22m_label_spacer	235
10.12.4.23m_label_modified	236
10.12.4.24m_label_category	236
10.12.4.25m_entry_ev_timestamp	236
10.12.4.26m_entry_ev_name	236
10.12.4.27m_entry_ev_data_0	236
10.12.4.28m_entry_ev_data_1	236
10.12.4.29m_label_time_fmt	236
10.12.4.30m_label_right	236
10.12.4.31m_seq	237
10.12.4.32m_have_focus	237
10.13seq64::eventslots Class Reference	237
10.13.1 Constructor & Destructor Documentation	242
10.13.1.1 eventslots()	242
10.13.1.2 ~eventslots()	242
10.13.2 Member Function Documentation	242
10.13.2.1 get_length()	242
10.13.2.2 event_count()	242
10.13.2.3 count()	243

10.13.2.4	line_count()	243
10.13.2.5	line_maximum()	243
10.13.2.6	line_increment()	243
10.13.2.7	top_index()	243
10.13.2.8	current_index()	243
10.13.2.9	pager_index()	243
10.13.2.10	seq()	243
10.13.2.11	load_events()	244
10.13.2.12	set_current_event()	244
10.13.2.13	insert_event() [1/2]	244
10.13.2.14	insert_event() [2/2]	245
10.13.2.15	delete_current_event()	245
10.13.2.16	modify_current_event()	246
10.13.2.17	save_events()	247
10.13.2.18	select_event()	247
10.13.2.19	set_text()	248
10.13.2.20	enqueue_draw()	248
10.13.2.21	convert_y()	248
10.13.2.22	draw_event()	248
10.13.2.23	draw_events()	249
10.13.2.24	change_vert()	249
10.13.2.25	page_movement()	250
10.13.2.26	page_topper()	250
10.13.2.27	decrement_top()	250
10.13.2.28	increment_top()	250
10.13.2.29	decrement_current()	251
10.13.2.30	increment_current()	251
10.13.2.31	decrement_bottom()	251
10.13.2.32	increment_bottom()	251
10.13.2.33	calculate_measures()	252

10.13.2.34	on_realize()	252
10.13.2.35	on_expose_event()	252
10.13.2.36	on_button_press_event()	252
10.13.2.37	on_button_release_event()	252
10.13.2.38	on_focus_in_event()	252
10.13.2.39	on_focus_out_event()	253
10.13.2.40	on_scroll_event()	253
10.13.2.41	on_size_allocate()	253
10.13.2.42	on_move_up()	253
10.13.2.43	on_move_down()	253
10.13.2.44	on_frame_up()	253
10.13.2.45	on_frame_down()	253
10.13.2.46	on_frame_home()	254
10.13.2.47	on_frame_end()	254
10.13.3	Friends And Related Function Documentation	254
10.13.3.1	eventedit	254
10.13.4	Field Documentation	254
10.13.4.1	m_parent	254
10.13.4.2	m_seq	254
10.13.4.3	m_event_container	254
10.13.4.4	m_slots_chars	255
10.13.4.5	m_char_w	255
10.13.4.6	m_setbox_w	255
10.13.4.7	m_slots_x	255
10.13.4.8	m_slots_y	255
10.13.4.9	m_event_count	255
10.13.4.10	m_last_max_timestamp	255
10.13.4.11	m_measures	256
10.13.4.12	m_line_count	256
10.13.4.13	m_line_maximum	256

10.13.4.14	m_line_overlap	256
10.13.4.15	m_top_index	256
10.13.4.16	m_current_index	256
10.13.4.17	m_top_iterator	256
10.13.4.18	m_bottom_iterator	257
10.13.4.19	m_current_iterator	257
10.13.4.20	m_pager_index	257
10.14	seq64::font Class Reference	257
10.14.1	Member Enumeration Documentation	258
10.14.1.1	Color	258
10.14.2	Constructor & Destructor Documentation	259
10.14.2.1	font()	259
10.14.3	Member Function Documentation	259
10.14.3.1	init()	259
10.14.3.2	render_string_on_drawable()	259
10.14.3.3	char_width()	260
10.14.3.4	char_height()	260
10.14.3.5	padded_height()	260
10.14.4	Field Documentation	260
10.14.4.1	m_use_new_font	260
10.14.4.2	m_cell_w	260
10.14.4.3	m_cell_h	261
10.14.4.4	m_font_w	261
10.14.4.5	m_font_h	261
10.14.4.6	m_offset	261
10.14.4.7	m_padded_h	261
10.14.4.8	m_pixmap	261
10.14.4.9	m_black_pixmap	261
10.14.4.10	m_white_pixmap	262
10.14.4.11	m_b_on_y_pixmap	262

10.14.4.12m_y_on_b_pixmap	262
10.14.4.13m_b_on_c_pixmap	262
10.14.4.14m_c_on_b_pixmap	262
10.14.4.15m_clip_mask	262
10.15seq64::FruityPerfInput Class Reference	263
10.15.1 Constructor & Destructor Documentation	264
10.15.1.1 FruityPerfInput()	264
10.15.1.2 ~FruityPerfInput()	265
10.15.2 Member Function Documentation	265
10.15.2.1 update_mouse_pointer()	265
10.15.2.2 activate_adding()	265
10.15.2.3 handle_motion_key()	265
10.15.2.4 on_button_press_event()	265
10.15.2.5 on_button_release_event()	266
10.15.2.6 on_motion_notify_event()	266
10.15.2.7 on_left_button_pressed()	266
10.15.2.8 on_right_button_pressed()	267
10.16seq64::FruitySeqEventInput Class Reference	267
10.16.1 Constructor & Destructor Documentation	269
10.16.1.1 FruitySeqEventInput()	269
10.16.2 Member Function Documentation	269
10.16.2.1 update_mouse_pointer()	269
10.16.2.2 on_button_press_event()	270
10.16.2.3 on_button_release_event()	270
10.16.2.4 on_motion_notify_event()	271
10.16.3 Field Documentation	271
10.16.3.1 m_justselected_one	271
10.16.3.2 m_is_drag_pasting_start	271
10.16.3.3 m_is_drag_pasting	271
10.17seq64::FruitySeqRollInput Class Reference	272

10.17.1 Constructor & Destructor Documentation	273
10.17.1.1 FruitySeqRollInput()	273
10.17.2 Member Function Documentation	273
10.17.2.1 update_mouse_pointer()	273
10.17.2.2 on_button_press_event()	274
10.17.2.3 on_button_release_event()	274
10.17.2.4 on_motion_notify_event()	274
10.17.3 Field Documentation	275
10.17.3.1 m_can_add	275
10.17.3.2 m_erase_painting	275
10.17.3.3 m_drag_paste_start_pos	275
10.18seq64::gui_assistant Class Reference	276
10.18.1 Detailed Description	277
10.18.2 Constructor & Destructor Documentation	277
10.18.2.1 gui_assistant()	277
10.18.2.2 ~gui_assistant()	277
10.18.3 Member Function Documentation	277
10.18.3.1 quit()	277
10.18.3.2 jack_idle_connect()	278
10.18.3.3 lash_timeout_connect()	278
10.18.3.4 keys() [1/2]	278
10.18.3.5 keys() [2/2]	278
10.18.4 Field Documentation	278
10.18.4.1 m_keys_perform	278
10.19seq64::gui_assistant_gtk2 Class Reference	279
10.19.1 Constructor & Destructor Documentation	280
10.19.1.1 gui_assistant_gtk2()	280
10.19.1.2 ~gui_assistant_gtk2()	280
10.19.2 Member Function Documentation	280
10.19.2.1 quit()	280

10.19.2.2	lash_timeout_connect()	280
10.19.2.3	jack_idle_connect()	280
10.19.3	Field Documentation	281
10.19.3.1	sm_internal_keys	281
10.20	seq64::gui_drawingarea_gtk2 Class Reference	281
10.20.1	Detailed Description	284
10.20.2	Constructor & Destructor Documentation	284
10.20.2.1	gui_drawingarea_gtk2() [1/3]	285
10.20.2.2	gui_drawingarea_gtk2() [2/3]	285
10.20.2.3	gui_drawingarea_gtk2() [3/3]	285
10.20.2.4	~gui_drawingarea_gtk2()	285
10.20.3	Member Function Documentation	285
10.20.3.1	operator=()	285
10.20.3.2	window_x()	285
10.20.3.3	width()	286
10.20.3.4	window_y()	286
10.20.3.5	height()	286
10.20.3.6	current_x()	286
10.20.3.7	current_y()	286
10.20.3.8	drop_x()	286
10.20.3.9	drop_y()	286
10.20.3.10	get_color()	286
10.20.3.11	get_sequence_color()	287
10.20.3.12	force_draw()	287
10.20.3.13	perf() [1/2]	287
10.20.3.14	perf() [2/2]	287
10.20.3.15	clear_window()	287
10.20.3.16	set_line()	287
10.20.3.17	draw_line() [1/6]	288
10.20.3.18	draw_line() [2/6]	288

10.20.3.19	draw_line_on_pixmap() [1/2]	288
10.20.3.20	draw_line_on_pixmap() [2/2]	289
10.20.3.21	draw_line() [3/6]	289
10.20.3.22	draw_line() [4/6]	290
10.20.3.23	draw_line() [5/6]	290
10.20.3.24	draw_line() [6/6]	290
10.20.3.25	render_string()	291
10.20.3.26	render_string_on_pixmap()	291
10.20.3.27	draw_rectangle() [1/6]	292
10.20.3.28	draw_rectangle() [2/6]	292
10.20.3.29	draw_rectangle() [3/6]	292
10.20.3.30	draw_rectangle() [4/6]	293
10.20.3.31	draw_rectangle() [5/6]	293
10.20.3.32	draw_rectangle() [6/6]	294
10.20.3.33	draw_rectangle_on_pixmap() [1/2]	294
10.20.3.34	draw_rectangle_on_pixmap() [2/2]	295
10.20.3.35	draw_normal_rectangle_on_pixmap()	295
10.20.3.36	draw_drawable()	296
10.20.3.37	scroll_hadjust()	296
10.20.3.38	scroll_vadjust()	296
10.20.3.39	scroll_hset()	297
10.20.3.40	scroll_vset()	297
10.20.3.41	set_current_drop_x()	297
10.20.3.42	set_current_drop_y()	297
10.20.3.43	gtk_drawarea_init()	298
10.20.3.44	on_realize()	298
10.20.4	Field Documentation	298
10.20.4.1	m_gc	298
10.20.4.2	m_window	298
10.20.4.3	m_vadjust	298

10.20.4.4 <code>m_hadjust</code>	298
10.20.4.5 <code>m_pixmap</code>	298
10.20.4.6 <code>m_background</code>	299
10.20.4.7 <code>m_foreground</code>	299
10.20.4.8 <code>m_mainperf</code>	299
10.20.4.9 <code>m_window_x</code>	299
10.20.4.10 <code>m_window_y</code>	299
10.20.4.11 <code>m_current_x</code>	299
10.20.4.12 <code>m_current_y</code>	299
10.20.4.13 <code>m_drop_x</code>	300
10.20.4.14 <code>m_drop_y</code>	300
10.21 seq64::gui_palette_gtk2 Class Reference	300
10.21.1 Detailed Description	304
10.21.2 Member Typedef Documentation	304
10.21.2.1 <code>Color</code>	304
10.21.3 Constructor & Destructor Documentation	305
10.21.3.1 <code>gui_palette_gtk2()</code>	305
10.21.3.2 <code>~gui_palette_gtk2()</code>	305
10.21.4 Member Function Documentation	305
10.21.4.1 <code>initialize()</code>	305
10.21.4.2 <code>get_color()</code>	305
10.21.4.3 <code>get_color_ex()</code>	306
10.21.4.4 <code>load_inverse_palette()</code>	307
10.21.4.5 <code>is_inverse()</code>	307
10.21.4.6 <code>line_color()</code>	307
10.21.4.7 <code>progress_color()</code>	307
10.21.4.8 <code>black()</code>	307
10.21.4.9 <code>dark_red()</code>	308
10.21.4.10 <code>dark_green()</code>	308
10.21.4.11 <code>dark_orange()</code>	308

10.21.4.12dark_blue()	308
10.21.4.13dark_magenta()	308
10.21.4.14dark_cyan()	308
10.21.4.15white()	308
10.21.4.16grey_paint()	308
10.21.4.17dark_grey_paint()	309
10.21.4.18light_grey_paint()	309
10.21.4.19red()	309
10.21.4.20orange()	309
10.21.4.21yellow()	309
10.21.4.22green()	309
10.21.4.23magenta()	309
10.21.4.24blue()	309
10.21.4.25black_paint()	310
10.21.4.26white_paint()	310
10.21.4.27black_key_paint()	310
10.21.4.28white_key_paint()	310
10.21.4.29tempo_paint()	310
10.21.4.30sel_paint()	310
10.21.4.31bg_color() [1/2]	310
10.21.4.32bg_color() [2/2]	310
10.21.4.33fg_color() [1/2]	311
10.21.4.34fg_color() [2/2]	311
10.21.5 Field Documentation	311
10.21.5.1 m_palette	311
10.21.5.2 m_is_inverse	311
10.21.5.3 m_black	311
10.21.5.4 m_red	311
10.21.5.5 m_green	311
10.21.5.6 m_yellow	312

10.21.5.7 m_blue	312
10.21.5.8 m_magenta	312
10.21.5.9 m_cyan	312
10.21.5.10m_white	312
10.21.5.11m_dk_black	312
10.21.5.12m_dk_red	312
10.21.5.13m_dk_green	312
10.21.5.14m_dk_yellow	313
10.21.5.15m_dk_blue	313
10.21.5.16m_dk_magenta	313
10.21.5.17m_dk_cyan	313
10.21.5.18m_dk_white	313
10.21.5.19m_orange	313
10.21.5.20m_pink	313
10.21.5.21m_grey	313
10.21.5.22m_dk_orange	314
10.21.5.23m_dk_pink	314
10.21.5.24m_dk_grey	314
10.21.5.25m_grey_paint	314
10.21.5.26m_dk_grey_paint	314
10.21.5.27m_lt_grey_paint	314
10.21.5.28m_blk_paint	314
10.21.5.29m_wht_paint	314
10.21.5.30m_blk_key_paint	315
10.21.5.31m_wht_key_paint	315
10.21.5.32m_tempo_paint	315
10.21.5.33m_sel_paint	315
10.21.5.34m_line_color	315
10.21.5.35m_progress_color	315
10.21.5.36m_bg_color	315

10.21.5.37m_fg_color	315
10.22seq64::gui_window_gtk2 Class Reference	316
10.22.1 Constructor & Destructor Documentation	317
10.22.1.1 gui_window_gtk2()	317
10.22.1.2 ~gui_window_gtk2()	318
10.22.2 Member Function Documentation	318
10.22.2.1 perf() [1/2]	318
10.22.2.2 perf() [2/2]	318
10.22.2.3 quit()	318
10.22.2.4 redraw_period_ms()	318
10.22.2.5 is_realized()	318
10.22.2.6 scroll_hadjust()	318
10.22.2.7 scroll_vadjust()	319
10.22.2.8 scroll_hset()	319
10.22.2.9 scroll_vset()	319
10.22.2.10on_realize()	320
10.22.3 Field Documentation	320
10.22.3.1 m_mainperf	320
10.22.3.2 m_window_x	320
10.22.3.3 m_window_y	320
10.22.3.4 m_redraw_period_ms	320
10.22.3.5 m_is_realized	320
10.23seq64::jack_assistant Class Reference	320
10.23.1 Constructor & Destructor Documentation	324
10.23.1.1 jack_assistant()	324
10.23.1.2 ~jack_assistant()	325
10.23.2 Member Function Documentation	325
10.23.2.1 show_position()	325
10.23.2.2 get_state_name()	326
10.23.2.3 parent() [1/2]	326

10.23.2.4 parent() [2/2]	327
10.23.2.5 is_running()	327
10.23.2.6 is_master()	327
10.23.2.7 get_ppqn()	327
10.23.2.8 get_beat_width()	327
10.23.2.9 set_beat_width()	327
10.23.2.10get_beats_per_measure()	327
10.23.2.11set_beats_per_measure()	328
10.23.2.12get_beats_per_minute()	328
10.23.2.13set_beats_per_minute()	328
10.23.2.14transport_state()	328
10.23.2.15transport_not_starting()	328
10.23.2.16nit()	329
10.23.2.17deinit()	329
10.23.2.18session_event()	330
10.23.2.19activate()	330
10.23.2.20start()	330
10.23.2.21stop()	330
10.23.2.22position()	331
10.23.2.23output()	332
10.23.2.24set_ppqn()	332
10.23.2.25get_jack_tick()	333
10.23.2.26get_jack_pos()	333
10.23.2.27toggle_jack_mode()	333
10.23.2.28set_jack_mode()	333
10.23.2.29get_jack_mode()	333
10.23.2.30get_jack_stop_tick()	333
10.23.2.31set_jack_stop_tick()	333
10.23.2.32jack_frame_rate()	334
10.23.2.33get_follow_transport()	334

10.23.2.34	set_follow_transport()	334
10.23.2.35	toggle_follow_transport()	334
10.23.2.36	toggle_song_start_mode()	334
10.23.2.37	song_start_mode()	334
10.23.2.38	set_start_from_perfedit()	334
10.23.2.39	client()	335
10.23.2.40	client_name()	335
10.23.2.41	client_uuid()	335
10.23.2.42	set_jack_running()	335
10.23.2.43	tick_multiplier()	335
10.23.2.44	client_open()	335
10.23.2.45	get_jack_client_info()	336
10.23.2.46	sync()	336
10.23.2.47	info_message()	337
10.23.2.48	error_message()	337
10.23.3	Friends And Related Function Documentation	337
10.23.3.1	jack_transport_callback	337
10.23.3.2	jack_shutdown_callback	338
10.23.3.3	jack_timebase_callback	338
10.23.3.4	get_current_jack_position	340
10.23.3.5	jack_session_callback	340
10.23.4	Field Documentation	340
10.23.4.1	sm_status_pairs	341
10.23.4.2	m_jack_parent	341
10.23.4.3	m_jack_client	341
10.23.4.4	m_jack_client_name	341
10.23.4.5	m_jack_client_uuid	341
10.23.4.6	m_jack_frame_current	341
10.23.4.7	m_jack_frame_last	341
10.23.4.8	m_jack_pos	341

10.23.4.9 m_jack_transport_state	342
10.23.4.10m_jack_transport_state_last	342
10.23.4.11m_jack_tick	342
10.23.4.12m_jsession_ev	342
10.23.4.13m_jack_running	342
10.23.4.14m_jack_master	342
10.23.4.15m_jack_frame_rate	342
10.23.4.16m_toggle_jack	343
10.23.4.17m_jack_stop_tick	343
10.23.4.18m_follow_transport	343
10.23.4.19m_ppqn	343
10.23.4.20m_beats_per_measure	343
10.23.4.21m_beat_width	343
10.23.4.22m_beats_per_minute	343
10.24seq64::jack_scratchpad Class Reference	343
10.24.1 Detailed Description	344
10.24.2 Field Documentation	344
10.24.2.1 js_current_tick	344
10.24.2.2 js_total_tick	344
10.24.2.3 js_clock_tick	345
10.24.2.4 js_jack_stopped	345
10.24.2.5 js_dumping	345
10.24.2.6 js_init_clock	345
10.24.2.7 js_looping	345
10.24.2.8 js_playback_mode	345
10.24.2.9 js_ticks_converted	345
10.24.2.10js_ticks_delta	345
10.24.2.11js_ticks_converted_last	346
10.24.2.12s_delta_tick_frac	346
10.25seq64::jack_status_pair_t Struct Reference	346

10.25.1 Field Documentation	346
10.25.1.1 jf_bit	346
10.25.1.2 jf_meaning	346
10.26seq64::keybindentry Class Reference	346
10.26.1 Member Enumeration Documentation	347
10.26.1.1 type	347
10.26.2 Constructor & Destructor Documentation	348
10.26.2.1 keybindentry()	348
10.26.3 Member Function Documentation	348
10.26.3.1 set()	348
10.26.3.2 on_key_press_event()	348
10.26.4 Friends And Related Function Documentation	349
10.26.4.1 options	349
10.26.5 Field Documentation	349
10.26.5.1 m_key	349
10.26.5.2 m_type	349
10.26.5.3 m_perf	349
10.26.5.4 m_slot	349
10.27seq64::keys_perform Class Reference	350
10.27.1 Detailed Description	357
10.27.2 Member Typedef Documentation	357
10.27.2.1 SlotMap	357
10.27.2.2 RevSlotMap	357
10.27.3 Constructor & Destructor Documentation	357
10.27.3.1 keys_perform()	357
10.27.3.2 ~keys_perform()	357
10.27.4 Member Function Documentation	357
10.27.4.1 set_keys()	357
10.27.4.2 get_keys()	358
10.27.4.3 bpm_up() [1/2]	358

10.27.4.4 bpm_up() [2/2]	358
10.27.4.5 bpm_dn() [1/2]	358
10.27.4.6 bpm_dn() [2/2]	358
10.27.4.7 replace() [1/2]	359
10.27.4.8 replace() [2/2]	359
10.27.4.9 queue() [1/2]	359
10.27.4.10queue() [2/2]	359
10.27.4.11keep_queue() [1/2]	359
10.27.4.12keep_queue() [2/2]	360
10.27.4.13snapshot_1() [1/2]	360
10.27.4.14snapshot_1() [2/2]	360
10.27.4.15snapshot_2() [1/2]	360
10.27.4.16snapshot_2() [2/2]	360
10.27.4.17screenset_up() [1/2]	361
10.27.4.18screenset_up() [2/2]	361
10.27.4.19screenset_dn() [1/2]	361
10.27.4.20screenset_dn() [2/2]	361
10.27.4.21set_playing_screenset() [1/2]	361
10.27.4.22set_playing_screenset() [2/2]	361
10.27.4.23group_on() [1/2]	362
10.27.4.24group_on() [2/2]	362
10.27.4.25group_off() [1/2]	362
10.27.4.26group_off() [2/2]	362
10.27.4.27group_learn() [1/2]	362
10.27.4.28group_learn() [2/2]	363
10.27.4.29start() [1/2]	363
10.27.4.30start() [2/2]	363
10.27.4.31pause() [1/2]	363
10.27.4.32pause() [2/2]	363
10.27.4.33pattern_edit() [1/2]	364

10.27.4.34	pattern_edit() [2/2]	364
10.27.4.35	pattern_shift() [1/2]	364
10.27.4.36	pattern_shift() [2/2]	364
10.27.4.37	event_edit() [1/2]	364
10.27.4.38	event_edit() [2/2]	364
10.27.4.39	stop() [1/2]	365
10.27.4.40	stop() [2/2]	365
10.27.4.41	song_mode() [1/2]	365
10.27.4.42	song_mode() [2/2]	365
10.27.4.43	menu_mode() [1/2]	365
10.27.4.44	menu_mode() [2/2]	365
10.27.4.45	follow_transport() [1/2]	366
10.27.4.46	follow_transport() [2/2]	366
10.27.4.47	fast_forward() [1/2]	366
10.27.4.48	fast_forward() [2/2]	366
10.27.4.49	rewind() [1/2]	366
10.27.4.50	rewind() [2/2]	366
10.27.4.51	pointer_position() [1/2]	366
10.27.4.52	pointer_position() [2/2]	367
10.27.4.53	toggle_mutes() [1/2]	367
10.27.4.54	toggle_mutes() [2/2]	367
10.27.4.55	toggle_jack() [1/2]	367
10.27.4.56	toggle_jack() [2/2]	367
10.27.4.57	tap_bpm() [1/2]	367
10.27.4.58	tap_bpm() [2/2]	367
10.27.4.59	song_record() [1/2]	368
10.27.4.60	song_record() [2/2]	368
10.27.4.61	oneshot_queue() [1/2]	368
10.27.4.62	oneshot_queue() [2/2]	368
10.27.4.63	show_ui_sequence_key() [1/2]	368

10.27.4.64	show_ui_sequence_key() [2/2]	368
10.27.4.65	show_ui_sequence_number() [1/2]	368
10.27.4.66	show_ui_sequence_number() [2/2]	369
10.27.4.67	get_key_events()	369
10.27.4.68	get_key_count()	369
10.27.4.69	get_key_groups()	369
10.27.4.70	get_key_events_rev()	369
10.27.4.71	get_key_groups_rev()	369
10.27.4.72	lookup_keyevent_key()	370
10.27.4.73	lookup_keygroup_key()	370
10.27.4.74	lookup_keyevent_seq()	370
10.27.4.75	lookup_keygroup_group()	370
10.27.4.76	key_name()	371
10.27.4.77	set_all_key_events()	371
10.27.4.78	set_all_key_groups()	371
10.27.4.79	set_key_event()	371
10.27.4.80	set_key_group()	372
10.27.4.81	group_max() [1/2]	372
10.27.4.82	group_max() [2/2]	372
10.27.4.83	at_bpm_up()	372
10.27.4.84	at_bpm_dn()	372
10.27.4.85	at_replace()	373
10.27.4.86	at_queue()	373
10.27.4.87	at_keep_queue()	373
10.27.4.88	at_snapshot_1()	373
10.27.4.89	at_snapshot_2()	373
10.27.4.90	at_screenset_up()	373
10.27.4.91	at_screenset_dn()	373
10.27.4.92	at_set_playing_screenset()	373
10.27.4.93	at_group_on()	374

10.27.4.94	at_group_off()	374
10.27.4.95	at_group_learn()	374
10.27.4.96	at_start()	374
10.27.4.97	at_pause()	374
10.27.4.98	at_song_mode()	374
10.27.4.99	at_toggle_jack()	374
10.27.4.100	at_menu_mode()	374
10.27.4.101	at_follow_transport()	375
10.27.4.102	at_fast_forward()	375
10.27.4.103	at_rewind()	375
10.27.4.104	at_pointer_position()	375
10.27.4.105	at_toggle_mutes()	375
10.27.4.106	at_tap_bpm()	375
10.27.4.107	at_song_record()	375
10.27.4.108	at_oneshot_queue()	375
10.27.4.109	at_pattern_edit()	376
10.27.4.110	at_pattern_shift()	376
10.27.4.111	at_event_edit()	376
10.27.4.112	at_stop()	376
10.27.4.113	at_show_ui_sequence_key()	376
10.27.4.114	at_show_ui_sequence_number()	376
10.27.4.115	set_basic_key_events()	376
10.27.4.116	set_basic_key_groups()	376
10.27.5	Friends And Related Function Documentation	377
10.27.5.1	options	377
10.27.5.2	perform	377
10.27.5.3	optionsfile	377
10.27.6	Field Documentation	377
10.27.6.1	m_key_show_ui_sequence_key	377
10.27.6.2	m_key_show_ui_sequence_number	377

10.27.6.3 m_key_events	377
10.27.6.4 m_key_groups	378
10.27.6.5 m_key_events_rev	378
10.27.6.6 m_key_groups_rev	378
10.27.6.7 m_group_max	378
10.27.6.8 m_key_bpm_up	378
10.27.6.9 m_key_bpm_dn	378
10.27.6.10 m_key_replace	378
10.27.6.11 m_key_queue	379
10.27.6.12 m_key_keep_queue	379
10.27.6.13 m_key_snapshot_1	379
10.27.6.14 m_key_snapshot_2	379
10.27.6.15 m_key_screenset_up	379
10.27.6.16 m_key_screenset_dn	379
10.27.6.17 m_key_set_playing_screenset	379
10.27.6.18 m_key_group_on	379
10.27.6.19 m_key_group_off	380
10.27.6.20 m_key_group_learn	380
10.27.6.21 m_key_start	380
10.27.6.22 m_key_pause	380
10.27.6.23 m_key_song_mode	380
10.27.6.24 m_key_toggle_jack	380
10.27.6.25 m_key_menu_mode	380
10.27.6.26 m_key_follow_transport	380
10.27.6.27 m_key_rewind	381
10.27.6.28 m_key_fast_forward	381
10.27.6.29 m_key_pointer_position	381
10.27.6.30 m_key_toggle_mutes	381
10.27.6.31 m_key_tap_bpm	381
10.27.6.32 m_key_pattern_edit	381

10.27.6.33	m_key_pattern_shift	381
10.27.6.34	m_key_event_edit	381
10.27.6.35	m_key_stop	382
10.27.6.36	m_key_song_record	382
10.27.6.37	m_key_oneshot_queue	382
10.28	seq64::keys_perform_gtk2 Class Reference	382
10.28.1	Detailed Description	384
10.28.2	Constructor & Destructor Documentation	384
10.28.2.1	keys_perform_gtk2()	384
10.28.2.2	~keys_perform_gtk2()	384
10.28.3	Member Function Documentation	384
10.28.3.1	key_name()	384
10.28.3.2	set_all_key_events()	385
10.28.3.3	set_all_key_groups()	385
10.29	seq64::keys_perform_transfer Struct Reference	385
10.29.1	Field Documentation	386
10.29.1.1	kpt_bpm_up	386
10.29.1.2	kpt_bpm_dn	386
10.29.1.3	kpt_screenset_up	386
10.29.1.4	kpt_screenset_dn	386
10.29.1.5	kpt_set_playing_screenset	387
10.29.1.6	kpt_group_on	387
10.29.1.7	kpt_group_off	387
10.29.1.8	kpt_group_learn	387
10.29.1.9	kpt_replace	387
10.29.1.10	kpt_queue	387
10.29.1.11	kpt_keep_queue	387
10.29.1.12	kpt_snapshot_1	387
10.29.1.13	kpt_snapshot_2	388
10.29.1.14	kpt_start	388

10.29.1.15	kpt_stop	388
10.29.1.16	kpt_show_ui_sequence_key	388
10.29.1.17	kpt_show_ui_sequence_number	388
10.29.1.18	kpt_pattern_edit	388
10.29.1.19	kpt_pattern_shift	388
10.29.1.20	kpt_event_edit	388
10.29.1.21	kpt_tap_bpm	389
10.29.1.22	kpt_pause	389
10.29.1.23	kpt_song_mode	389
10.29.1.24	kpt_toggle_jack	389
10.29.1.25	kpt_menu_mode	389
10.29.1.26	kpt_follow_transport	389
10.29.1.27	kpt_fast_forward	389
10.29.1.28	kpt_rewind	389
10.29.1.29	kpt_pointer_position	390
10.29.1.30	kpt_toggle_mutes	390
10.29.1.31	kpt_song_record	390
10.29.1.32	kpt_oneshot_queue	390
10.30	seq64::keystroke Class Reference	390
10.30.1	Detailed Description	391
10.30.2	Constructor & Destructor Documentation	391
10.30.2.1	keystroke() [1/3]	391
10.30.2.2	keystroke() [2/3]	391
10.30.2.3	keystroke() [3/3]	392
10.30.3	Member Function Documentation	392
10.30.3.1	operator=()	392
10.30.3.2	is_press()	392
10.30.3.3	is_letter()	393
10.30.3.4	is() [1/2]	394
10.30.3.5	is() [2/2]	394

10.30.3.6 <code>is_delete()</code>	394
10.30.3.7 <code>key()</code>	395
10.30.3.8 <code>shift_lock()</code>	395
10.30.3.9 <code>modifier()</code>	395
10.30.3.10 <code>mod_control()</code>	395
10.30.3.11 <code>mod_control_shift()</code>	395
10.30.3.12 <code>mod_super()</code>	395
10.30.4 Field Documentation	396
10.30.4.1 <code>m_is_press</code>	396
10.30.4.2 <code>m_key</code>	396
10.30.4.3 <code>m_modifier</code>	396
10.31seq64::lash Class Reference	396
10.31.1 Detailed Description	397
10.31.2 Constructor & Destructor Documentation	397
10.31.2.1 <code>lash()</code>	397
10.31.3 Member Function Documentation	397
10.31.3.1 <code>set_alsa_client_id()</code>	397
10.31.3.2 <code>start()</code>	398
10.31.3.3 <code>process_events()</code>	398
10.31.3.4 <code>init()</code>	398
10.31.3.5 <code>handle_event()</code>	398
10.31.3.6 <code>handle_config()</code>	398
10.31.4 Field Documentation	399
10.31.4.1 <code>m_perform</code>	399
10.31.4.2 <code>m_client</code>	399
10.31.4.3 <code>m_lash_args</code>	399
10.31.4.4 <code>m_is_lash_supported</code>	399
10.32seq64::lfownd Class Reference	400
10.32.1 Detailed Description	401
10.32.2 Constructor & Destructor Documentation	402

10.32.2.1 lfownd()	402
10.32.2.2 ~lfownd()	402
10.32.3 Member Function Documentation	402
10.32.3.1 toggle_visible()	402
10.32.3.2 scale_lfo_change()	402
10.32.3.3 on_focus_out_event()	402
10.32.4 Field Documentation	403
10.32.4.1 m_seq	403
10.32.4.2 m_seqdata	403
10.32.4.3 m_hbox	403
10.32.4.4 m_scale_value	403
10.32.4.5 m_scale_range	403
10.32.4.6 m_scale_speed	403
10.32.4.7 m_scale_phase	403
10.32.4.8 m_scale_wave	404
10.32.4.9 m_wave_name	404
10.32.4.10m_value	404
10.32.4.11m_range	404
10.32.4.12m_speed	404
10.32.4.13m_phase	404
10.32.4.14m_wave	404
10.33seq64::maintime Class Reference	405
10.33.1 Detailed Description	406
10.33.2 Constructor & Destructor Documentation	407
10.33.2.1 maintime() [1/2]	407
10.33.2.2 maintime() [2/2]	407
10.33.2.3 ~maintime()	407
10.33.3 Member Function Documentation	407
10.33.3.1 operator=()	407
10.33.3.2 idle_progress()	407

10.33.3.3 on_expose_event()	408
10.33.4 Friends And Related Function Documentation	408
10.33.4.1 mainwnd	408
10.33.5 Field Documentation	408
10.33.5.1 m_beat_width	408
10.33.5.2 m_bar_width	408
10.33.5.3 m_pill_width	408
10.33.5.4 m_box_width	409
10.33.5.5 m_box_height	409
10.33.5.6 m_flash_width	409
10.33.5.7 m_flash_height	409
10.33.5.8 m_flash_x	409
10.33.5.9 m_box_less_pill	409
10.33.5.10 m_tick	409
10.33.5.11 m_ppqn	409
10.34 seq64::mainwid Class Reference	410
10.34.1 Detailed Description	413
10.34.2 Constructor & Destructor Documentation	413
10.34.2.1 mainwid()	413
10.34.2.2 ~mainwid()	414
10.34.3 Member Function Documentation	414
10.34.3.1 set_screenset()	414
10.34.3.2 log_screenset()	414
10.34.3.3 reset()	415
10.34.3.4 update_sequences_on_window()	415
10.34.3.5 draw_pixmap_on_window()	415
10.34.3.6 fill_background_window()	415
10.34.3.7 nominal_width()	415
10.34.3.8 nominal_height()	415
10.34.3.9 redraw()	415

10.34.3.10	<code>seq_set_and_edit()</code>	416
10.34.3.11	<code>seq_set_and_eventedit()</code>	416
10.34.3.12	<code>draw_marker_on_sequence()</code>	416
10.34.3.13	<code>update_markers()</code>	417
10.34.3.14	<code>valid_sequence()</code>	417
10.34.3.15	<code>draw_sequence_on_pixmap()</code>	417
10.34.3.16	<code>draw_sequences_on_pixmap()</code>	418
10.34.3.17	<code>draw_sequence_pixmap_on_window()</code>	418
10.34.3.18	<code>seq_from_xy()</code>	418
10.34.3.19	<code>timeout()</code>	418
10.34.3.20	<code>calculate_base_sizes()</code>	419
10.34.3.21	<code>select_fg_bg_colors()</code>	419
10.34.3.22	<code>on_realize()</code>	419
10.34.3.23	<code>on_expose_event()</code>	419
10.34.3.24	<code>on_button_press_event()</code>	420
10.34.3.25	<code>on_button_release_event()</code>	420
10.34.3.26	<code>on_motion_notify_event()</code>	421
10.34.3.27	<code>on_focus_in_event()</code>	421
10.34.3.28	<code>on_focus_out_event()</code>	421
10.34.4	Friends And Related Function Documentation	421
10.34.4.1	<code>mainwnd</code>	422
10.34.4.2	<code>update_mainwid_sequences</code>	422
10.34.5	Field Documentation	422
10.34.5.1	<code>m_armed_progress_color</code>	422
10.34.5.2	<code>m_moving_seq</code>	422
10.34.5.3	<code>m_button_down</code>	422
10.34.5.4	<code>m_moving</code>	422
10.34.5.5	<code>m_old_seq</code>	422
10.34.5.6	<code>m_screenset</code>	423
10.34.5.7	<code>m_last_tick_x</code>	423

10.34.5.8 m_mainwnd_rows	423
10.34.5.9 m_mainwnd_cols	423
10.34.5.10m_seqarea_x	423
10.34.5.11m_seqarea_y	423
10.34.5.12m_seqarea_seq_x	423
10.34.5.13m_seqarea_seq_y	423
10.34.5.14m_mainwid_x	424
10.34.5.15m_mainwid_y	424
10.34.5.16m_mainwid_border_x	424
10.34.5.17m_mainwid_border_y	424
10.34.5.18m_mainwid_spacing	424
10.34.5.19m_text_size_x	424
10.34.5.20m_text_size_y	424
10.34.5.21m_max_sets	424
10.34.5.22m_screenset_slots	425
10.34.5.23m_screenset_offset	425
10.34.5.24m_progress_height	425
10.35seq64::mainwnd Class Reference	425
10.35.1 Member Enumeration Documentation	433
10.35.1.1 SaveOption	433
10.35.2 Constructor & Destructor Documentation	433
10.35.2.1 mainwnd()	433
10.35.2.2 ~mainwnd()	434
10.35.3 Member Function Documentation	434
10.35.3.1 open_file()	434
10.35.3.2 rc_error_dialog()	434
10.35.3.3 ppqn() [1/2]	434
10.35.3.4 ppqn() [2/2]	435
10.35.3.5 handle_signal()	435
10.35.3.6 debug_text()	435

10.35.3.7 multi_wid()	435
10.35.3.8 adj_callback_wid()	435
10.35.3.9 independent()	436
10.35.3.10 need_set_spinner()	436
10.35.3.11 adj_callback_ss()	436
10.35.3.12 adj_callback_bpm()	437
10.35.3.13 edit_callback_notepad()	437
10.35.3.14 set_wid_label()	437
10.35.3.15 update_screenset()	437
10.35.3.16 update_markers()	437
10.35.3.17 reset()	438
10.35.3.18 reset_window()	438
10.35.3.19 set_play_image()	438
10.35.3.20 set_songlive_image()	438
10.35.3.21 start_playing()	438
10.35.3.22 pause_playing()	439
10.35.3.23 stop_playing()	439
10.35.3.24 toggle_playing()	439
10.35.3.25 timer_callback()	439
10.35.3.26 set_screenset()	439
10.35.3.27 tempo_log()	440
10.35.3.28 toggle_tempo_record()	440
10.35.3.29 toggle_time_format()	440
10.35.3.30 queue_it()	440
10.35.3.31 set_song_record()	440
10.35.3.32 toggle_song_record()	441
10.35.3.33 toggle_song_snap()	441
10.35.3.34 set_song_playback()	441
10.35.3.35 song_record_snap()	441
10.35.3.36 panic()	441

10.35.3.37	learn_toggle()	441
10.35.3.38	open_performance_edit()	442
10.35.3.39	open_performance_edit_2()	442
10.35.3.40	enregister_perfedits()	442
10.35.3.41	sequence_key()	442
10.35.3.42	spinner_max()	442
10.35.3.43	apply_song_transpose()	442
10.35.3.44	clear_mute_groups()	442
10.35.3.45	reload_mute_groups()	443
10.35.3.46	update_window_title()	443
10.35.3.47	update_recent_files_menu()	443
10.35.3.48	load_recent_file()	443
10.35.3.49	toLower()	443
10.35.3.50	file_new()	443
10.35.3.51	file_open()	444
10.35.3.52	file_save()	444
10.35.3.53	set_song_mute()	444
10.35.3.54	wid_box_to_slot()	444
10.35.3.55	wid_slot_to_box()	444
10.35.3.56	file_import_dialog()	444
10.35.3.57	options_dialog()	445
10.35.3.58	jack_dialog()	445
10.35.3.59	about_dialog()	445
10.35.3.60	build_info_dialog()	445
10.35.3.61	query_save_changes()	445
10.35.3.62	new_open_error_dialog()	445
10.35.3.63	file_save_as()	446
10.35.3.64	file_exit()	446
10.35.3.65	new_file()	447
10.35.3.66	save_file()	447

10.35.3.67	choose_file()	447
10.35.3.68	ss_save()	447
10.35.3.69	install_signal_handlers()	447
10.35.3.70	signal_action()	447
10.35.3.71	edit_field_has_focus()	448
10.35.3.72	populate_menu_file()	448
10.35.3.73	populate_menu_edit()	448
10.35.3.74	populate_menu_help()	448
10.35.3.75	populate_menu_view()	448
10.35.3.76	set_status_text()	448
10.35.3.77	on_delete_event()	449
10.35.3.78	on_key_press_event()	449
10.35.3.79	on_key_release_event()	449
10.35.3.80	on_realize()	450
10.35.3.81	on_grouplearnchange()	450
10.35.4	Field Documentation	450
10.35.4.1	sm_sigpipe	450
10.35.4.2	sm_widmax	450
10.35.4.3	m_menubar	450
10.35.4.4	m_menu_file	450
10.35.4.5	m_menu_recent	451
10.35.4.6	m_menu_edit	451
10.35.4.7	m_menu_view	451
10.35.4.8	m_menu_help	451
10.35.4.9	m_status_label	451
10.35.4.10	m_ppqn	451
10.35.4.11	m_mainwid_grid	451
10.35.4.12	m_mainwid_frames	451
10.35.4.13	m_mainwid_adjustors	452
10.35.4.14	m_mainwid_spinners	452

10.35.4.15m_mainwid_blocks	452
10.35.4.16m_mainwid_rows	452
10.35.4.17m_mainwid_columns	452
10.35.4.18m_mainwid_count	452
10.35.4.19m_mainwid_independent	452
10.35.4.20m_main_wid	453
10.35.4.21m_adjust_ss	453
10.35.4.22m_spinbutton_ss	453
10.35.4.23m_current_screenset	453
10.35.4.24m_main_time	453
10.35.4.25m_perf_edit	453
10.35.4.26m_perf_edit_2	453
10.35.4.27m_options	454
10.35.4.28m_main_cursor	454
10.35.4.29m_image_play	454
10.35.4.30m_button_panic	454
10.35.4.31m_button_learn	454
10.35.4.32m_button_stop	454
10.35.4.33m_button_play	454
10.35.4.34m_button_tempo_log	454
10.35.4.35m_button_tempo_record	455
10.35.4.36m_is_tempo_recording	455
10.35.4.37m_button_perfedit	455
10.35.4.38m_button_jack	455
10.35.4.39m_button_song_record	455
10.35.4.40m_button_song_snap	455
10.35.4.41m_is_song_recording	455
10.35.4.42m_is_snap_recording	456
10.35.4.43m_tick_time	456
10.35.4.44m_button_time_type	456

10.35.4.45m_tick_time_as_bbt	456
10.35.4.46m_adjust_bpm	456
10.35.4.47m_spinbutton_bpm	456
10.35.4.48m_button_queue	456
10.35.4.49m_adjust_load_offset	456
10.35.4.50m_spinbutton_load_offset	457
10.35.4.51m_entry_notes	457
10.35.4.52m_is_running	457
10.35.4.53m_timeout_connect	457
10.35.4.54m_menu_mode	457
10.35.4.55m_call_seq_edit	457
10.35.4.56m_call_seq_shift	457
10.35.4.57m_call_seq_eventedit	458
10.36seq64::mastermidibase Class Reference	458
10.36.1 Constructor & Destructor Documentation	463
10.36.1.1 mastermidibase()	463
10.36.1.2 ~mastermidibase()	463
10.36.2 Member Function Documentation	463
10.36.2.1 init()	463
10.36.2.2 announce_bus_exists()	464
10.36.2.3 get_num_out_buses()	464
10.36.2.4 get_num_in_buses()	464
10.36.2.5 filter_by_channel() [1/2]	464
10.36.2.6 filter_by_channel() [2/2]	464
10.36.2.7 get_beats_per_minute()	465
10.36.2.8 get_bpm()	465
10.36.2.9 get_ppqn()	465
10.36.2.10s_dumping()	465
10.36.2.11get_sequence()	465
10.36.2.12start()	465

10.36.2.13	stop()	465
10.36.2.14	port_start()	465
10.36.2.15	port_exit()	466
10.36.2.16	play()	466
10.36.2.17	continue_from()	467
10.36.2.18	nit_clock()	467
10.36.2.19	emit_clock()	467
10.36.2.20	sysex()	468
10.36.2.21	print()	468
10.36.2.22	flush()	468
10.36.2.23	panic()	468
10.36.2.24	set_sequence_input()	468
10.36.2.25	dump_midi_input()	469
10.36.2.26	get_midi_out_bus_name()	469
10.36.2.27	get_midi_in_bus_name()	470
10.36.2.28	poll_for_midi()	470
10.36.2.29	is_more_input()	470
10.36.2.30	get_midi_event()	470
10.36.2.31	set_clock()	471
10.36.2.32	set_input()	471
10.36.2.33	get_input()	472
10.36.2.34	is_input_system_port()	472
10.36.2.35	get_clock()	472
10.36.2.36	set_ppqn()	473
10.36.2.37	set_beats_per_minute()	473
10.36.2.38	set_port_statuses()	473
10.36.2.39	get_port_statuses()	473
10.36.2.40	clock()	474
10.36.2.41	input()	474
10.36.2.42	activate()	474

10.36.2.43	api_init()	474
10.36.2.44	api_start()	474
10.36.2.45	api_continue_from()	474
10.36.2.46	api_init_clock()	475
10.36.2.47	api_stop()	475
10.36.2.48	api_set_ppqn()	475
10.36.2.49	api_set_beats_per_minute()	475
10.36.2.50	api_flush()	475
10.36.2.51	api_clock()	475
10.36.2.52	api_port_start()	476
10.36.2.53	api_get_midi_event()	476
10.36.2.54	api_poll_for_midi()	476
10.36.2.55	save_clock()	476
10.36.2.56	save_input()	477
10.36.3	Friends And Related Function Documentation	477
10.36.3.1	perform	477
10.36.3.2	midi_alsa_info	477
10.36.4	Field Documentation	477
10.36.4.1	m_max_busses	477
10.36.4.2	m_bus_announce	478
10.36.4.3	m_inbus_array	478
10.36.4.4	m_outbus_array	478
10.36.4.5	m_master_clocks	478
10.36.4.6	m_master_inputs	478
10.36.4.7	m_queue	478
10.36.4.8	m_ppqn	478
10.36.4.9	m_beats_per_minute	479
10.36.4.10	m_dumping_input	479
10.36.4.11	m_vector_sequence	479
10.36.4.12	m_filter_by_channel	479

10.36.4.13m_seq	479
10.36.4.14m_mutex	479
10.37seq64::mastermidibus Class Reference	480
10.37.1 Detailed Description	481
10.37.2 Constructor & Destructor Documentation	481
10.37.2.1 mastermidibus()	481
10.37.2.2 ~mastermidibus()	482
10.37.3 Member Function Documentation	482
10.37.3.1 activate()	482
10.37.3.2 api_get_midi_event()	482
10.37.3.3 api_poll_for_midi()	482
10.37.3.4 api_init()	482
10.37.3.5 api_set_ppqn()	483
10.37.3.6 api_set_beats_per_minute()	483
10.37.3.7 api_flush()	483
10.37.3.8 api_port_start()	483
10.37.3.9 port_list()	483
10.37.4 Friends And Related Function Documentation	483
10.37.4.1 midi_alsa_info	483
10.37.4.2 midi_jack_info	484
10.37.5 Field Documentation	484
10.37.5.1 m_midi_master	484
10.37.5.2 m_use_jack_polling	484
10.38seq64::editable_event::meta_length_t Struct Reference	484
10.38.1 Field Documentation	484
10.38.1.1 event_value	484
10.38.1.2 event_length	484
10.39seq64::midi_alsa Class Reference	485
10.39.1 Constructor & Destructor Documentation	486
10.39.1.1 midi_alsa()	487

10.39.1.2 ~midi_alsa()	487
10.39.2 Member Function Documentation	487
10.39.2.1 get_client()	487
10.39.2.2 get_port()	487
10.39.2.3 api_init_out()	487
10.39.2.4 api_init_in()	487
10.39.2.5 api_init_out_sub()	488
10.39.2.6 api_init_in_sub()	488
10.39.2.7 api_deinit_in()	488
10.39.2.8 api_get_midi_event()	488
10.39.2.9 api_play()	488
10.39.2.10 api_sysex()	488
10.39.2.11 api_flush()	489
10.39.2.12 api_continue_from()	489
10.39.2.13 api_start()	489
10.39.2.14 api_stop()	489
10.39.2.15 api_clock()	489
10.39.2.16 api_set_ppqn()	489
10.39.2.17 api_set_beats_per_minute()	490
10.39.2.18 set_virtual_name()	490
10.39.3 Field Documentation	490
10.39.3.1 m_seq	490
10.39.3.2 m_dest_addr_client	490
10.39.3.3 m_dest_addr_port	490
10.39.3.4 m_local_addr_client	490
10.39.3.5 m_local_addr_port	491
10.39.3.6 m_input_port_name	491
10.40 seq64::midi_alsa_info Class Reference	491
10.40.1 Constructor & Destructor Documentation	493
10.40.1.1 midi_alsa_info()	493

10.40.1.2 ~midi_alsa_info()	493
10.40.2 Member Function Documentation	494
10.40.2.1 seq()	494
10.40.2.2 api_get_midi_event()	494
10.40.2.3 api_poll_for_midi()	494
10.40.2.4 api_set_ppqn()	494
10.40.2.5 api_set_beats_per_minute()	494
10.40.2.6 api_port_start()	495
10.40.2.7 api_flush()	495
10.40.2.8 get_all_port_info()	495
10.40.3 Field Documentation	495
10.40.3.1 sm_input_caps	495
10.40.3.2 sm_output_caps	495
10.40.3.3 m_alsa_seq	495
10.40.3.4 m_num_poll_descriptors	496
10.40.3.5 m_poll_descriptors	496
10.41 seq64::midi_api Class Reference	496
10.41.1 Detailed Description	499
10.41.2 Constructor & Destructor Documentation	499
10.41.2.1 midi_api()	499
10.41.2.2 ~midi_api()	499
10.41.3 Member Function Documentation	499
10.41.3.1 is_input_port()	499
10.41.3.2 is_virtual_port()	500
10.41.3.3 is_system_port()	500
10.41.3.4 api_connect()	500
10.41.3.5 api_poll_for_midi()	500
10.41.3.6 api_init_out()	500
10.41.3.7 api_init_out_sub()	500
10.41.3.8 api_init_in()	501

10.41.3.9	api_init_in_sub()	501
10.41.3.10	api_deinit_in()	501
10.41.3.11	api_get_midi_event()	501
10.41.3.12	api_play()	501
10.41.3.13	api_sysex()	502
10.41.3.14	api_continue_from()	502
10.41.3.15	api_start()	502
10.41.3.16	api_stop()	502
10.41.3.17	api_flush()	502
10.41.3.18	api_clock()	503
10.41.3.19	api_set_ppqn()	503
10.41.3.20	api_set_beats_per_minute()	503
10.41.3.21	api_get_bus_name()	503
10.41.3.22	api_get_port_name()	503
10.41.3.23	s_port_open()	503
10.41.3.24	master_info() [1/2]	504
10.41.3.25	master_info() [2/2]	504
10.41.3.26	parent_bus() [1/2]	504
10.41.3.27	parent_bus() [2/2]	504
10.41.3.28	master_midi_mode()	504
10.41.3.29	error()	504
10.41.3.30	user_callback()	504
10.41.3.31	cancel_callback()	505
10.41.3.32	set_port_open()	505
10.41.3.33	input_data()	505
10.41.4	Field Documentation	505
10.41.4.1	m_master_info	505
10.41.4.2	m_parent_bus	505
10.41.4.3	m_input_data	505
10.41.4.4	m_connected	505

10.41.4.5 m_error_string	506
10.41.4.6 m_error_callback	506
10.41.4.7 m_first_error_occurred	506
10.41.4.8 m_error_callback_user_data	506
10.42seq64::midi_container Class Reference	506
10.42.1 Detailed Description	509
10.42.2 Constructor & Destructor Documentation	509
10.42.2.1 midi_container()	509
10.42.2.2 ~midi_container()	509
10.42.3 Member Function Documentation	509
10.42.3.1 fill()	509
10.42.3.2 size()	510
10.42.3.3 done()	511
10.42.3.4 put()	511
10.42.3.5 get()	511
10.42.3.6 clear()	511
10.42.3.7 position_reset()	511
10.42.3.8 position()	512
10.42.3.9 position_increment()	512
10.42.3.10add_variable()	512
10.42.3.11add_long()	512
10.42.3.12add_short()	512
10.42.3.13add_event()	513
10.42.3.14add_ex_event()	513
10.42.3.15fill_seq_number()	513
10.42.3.16fill_seq_name()	514
10.42.3.17fill_meta_track_end()	514
10.42.3.18fill_proprietary()	514
10.42.3.19song_fill_seq_event()	515
10.42.3.20song_fill_seq_trigger()	515

10.42.4 Friends And Related Function Documentation	515
10.42.4.1 midifile	515
10.42.5 Field Documentation	515
10.42.5.1 m_sequence	516
10.42.5.2 m_position_for_get	516
10.43seq64::midi_control Class Reference	516
10.43.1 Detailed Description	517
10.43.2 Member Enumeration Documentation	517
10.43.2.1 action	517
10.43.3 Constructor & Destructor Documentation	518
10.43.3.1 midi_control()	518
10.43.4 Member Function Documentation	518
10.43.4.1 active()	518
10.43.4.2 inverse_active()	518
10.43.4.3 status()	518
10.43.4.4 data()	519
10.43.4.5 min_value()	519
10.43.4.6 max_value()	519
10.43.4.7 set() [1 / 2]	519
10.43.4.8 set() [2 / 2]	519
10.43.4.9 match()	520
10.43.4.10 n_range()	520
10.43.5 Field Documentation	520
10.43.5.1 m_active	520
10.43.5.2 m_inverse_active	520
10.43.5.3 m_status	520
10.43.5.4 m_data	520
10.43.5.5 m_min_value	521
10.43.5.6 m_max_value	521
10.44seq64::midi_in_alsa Class Reference	521

10.44.1 Constructor & Destructor Documentation	523
10.44.1.1 midi_in_alsa()	523
10.45seq64::midi_in_jack Class Reference	523
10.45.1 Constructor & Destructor Documentation	525
10.45.1.1 midi_in_jack()	525
10.45.1.2 ~midi_in_jack()	525
10.45.2 Member Function Documentation	525
10.45.2.1 api_poll_for_midi()	525
10.45.2.2 api_get_midi_event()	525
10.45.2.3 open_client()	526
10.45.3 Field Documentation	526
10.45.3.1 m_client_name	526
10.46seq64::midi_info Class Reference	526
10.46.1 Constructor & Destructor Documentation	529
10.46.1.1 midi_info()	529
10.46.1.2 ~midi_info()	530
10.46.2 Member Function Documentation	530
10.46.2.1 midi_mode() [1/2]	530
10.46.2.2 midi_mode() [2/2]	530
10.46.2.3 midi_handle() [1/2]	530
10.46.2.4 input_ports()	530
10.46.2.5 output_ports()	530
10.46.2.6 full_port_count()	530
10.46.2.7 clear()	531
10.46.2.8 app_name()	531
10.46.2.9 ppqn()	531
10.46.2.10 bpm()	531
10.46.2.11 api_set_ppqn()	531
10.46.2.12 api_set_beats_per_minute()	531
10.46.2.13 api_port_start()	531

10.46.2.14	api_get_midi_event()	532
10.46.2.15	api_poll_for_midi()	532
10.46.2.16	api_flush()	532
10.46.2.17	api_connect()	532
10.46.2.18	get_port_count()	532
10.46.2.19	get_bus_id()	532
10.46.2.20	get_bus_name()	533
10.46.2.21	get_port_id()	533
10.46.2.22	get_port_name()	533
10.46.2.23	get_input()	533
10.46.2.24	get_virtual()	533
10.46.2.25	get_system()	533
10.46.2.26	queue_number()	533
10.46.2.27	connect_name()	534
10.46.2.28	port_list()	534
10.46.2.29	global_queue() [1/2]	534
10.46.2.30	error()	534
10.46.2.31	get_all_port_info()	534
10.46.2.32	add_bus()	534
10.46.2.33	global_queue() [2/2]	534
10.46.2.34	midi_handle() [2/2]	535
10.46.2.35	bus_container()	535
10.46.2.36	nc_midi_port_info()	535
10.46.2.37	ref_midi_port_info()	535
10.46.3	Friends And Related Function Documentation	535
10.46.3.1	rtmidi_info	535
10.46.4	Field Documentation	535
10.46.4.1	m_midi_mode_input	535
10.46.4.2	m_input	536
10.46.4.3	m_output	536

10.46.4.4 m_bus_container	536
10.46.4.5 m_global_queue	536
10.46.4.6 m_midi_handle	536
10.46.4.7 m_app_name	536
10.46.4.8 m_ppqn	536
10.46.4.9 m_bpm	536
10.46.4.10m_error_string	537
10.47seq64::midi_jack Class Reference	537
10.47.1 Constructor & Destructor Documentation	539
10.47.1.1 midi_jack() [1/2]	539
10.47.1.2 midi_jack() [2/2]	539
10.47.1.3 ~midi_jack()	539
10.47.2 Member Function Documentation	539
10.47.2.1 client_handle() [1/2]	540
10.47.2.2 jack_data()	540
10.47.2.3 remote_port_name() [1/2]	540
10.47.2.4 remote_port_name() [2/2]	540
10.47.2.5 port_handle() [1/2]	540
10.47.2.6 client_handle() [2/2]	540
10.47.2.7 port_handle() [2/2]	540
10.47.2.8 open_client_impl()	541
10.47.2.9 close_client()	541
10.47.2.10close_port()	541
10.47.2.11create_ringbuffer()	541
10.47.2.12connect_port()	541
10.47.2.13register_port()	541
10.47.2.14open_client()	541
10.47.2.15api_connect()	542
10.47.2.16api_init_out()	542
10.47.2.17api_init_in()	542

10.47.2.18	api_init_out_sub()	542
10.47.2.19	api_init_in_sub()	542
10.47.2.20	api_deinit_in()	542
10.47.2.21	api_get_midi_event()	543
10.47.2.22	api_play()	543
10.47.2.23	api_sysex()	543
10.47.2.24	api_flush()	543
10.47.2.25	api_continue_from()	543
10.47.2.26	api_start()	544
10.47.2.27	api_stop()	544
10.47.2.28	api_clock()	544
10.47.2.29	api_set_ppqn()	544
10.47.2.30	api_set_beats_per_minute()	544
10.47.2.31	api_get_port_name()	544
10.47.2.32	send_byte()	545
10.47.2.33	send_message()	545
10.47.2.34	set_virtual_name()	545
10.47.3	Friends And Related Function Documentation	545
10.47.3.1	midi_jack_info	545
10.47.4	Field Documentation	545
10.47.4.1	m_remote_port_name	545
10.47.4.2	m_jack_info	545
10.47.4.3	m_jack_data	546
10.48	seq64::midi_jack_data Struct Reference	546
10.48.1	Detailed Description	546
10.48.2	Constructor & Destructor Documentation	546
10.48.2.1	midi_jack_data()	547
10.48.2.2	~midi_jack_data()	547
10.48.3	Member Function Documentation	547
10.48.3.1	valid_buffer()	547

10.48.4 Field Documentation	547
10.48.4.1 m_jack_client	547
10.48.4.2 m_jack_port	547
10.48.4.3 m_jack_buffsize	547
10.48.4.4 m_jack_buffmessage	548
10.48.4.5 m_jack_lasttime	548
10.48.4.6 m_jack_rtmidiin	548
10.49seq64::midi_jack_info Class Reference	548
10.49.1 Constructor & Destructor Documentation	550
10.49.1.1 midi_jack_info()	550
10.49.1.2 ~midi_jack_info()	551
10.49.2 Member Function Documentation	551
10.49.2.1 client_handle() [1/2]	551
10.49.2.2 api_get_midi_event()	551
10.49.2.3 api_connect()	551
10.49.2.4 api_poll_for_midi()	551
10.49.2.5 api_set_ppqn()	551
10.49.2.6 api_set_beats_per_minute()	552
10.49.2.7 api_port_start()	552
10.49.2.8 api_flush()	552
10.49.2.9 get_all_port_info()	552
10.49.2.10client_handle() [2/2]	552
10.49.2.11connect()	552
10.49.2.12disconnect()	553
10.49.2.13extract_names()	553
10.49.2.14add()	553
10.49.3 Friends And Related Function Documentation	553
10.49.3.1 midi_jack	553
10.49.3.2 jack_process_io	553
10.49.4 Field Documentation	553

10.49.4.1 m_jack_ports	553
10.49.4.2 m_jack_client	554
10.49.4.3 m_jack_client_2	554
10.50seq64::midi_list Class Reference	554
10.50.1 Member Typedef Documentation	556
10.50.1.1 CharList	556
10.50.2 Constructor & Destructor Documentation	556
10.50.2.1 midi_list()	556
10.50.2.2 ~midi_list()	557
10.50.3 Member Function Documentation	557
10.50.3.1 size()	557
10.50.3.2 done()	557
10.50.3.3 put()	557
10.50.3.4 get()	557
10.50.3.5 clear()	558
10.50.4 Field Documentation	558
10.50.4.1 m_char_list	558
10.51seq64::midi_measures Class Reference	558
10.51.1 Detailed Description	559
10.51.2 Constructor & Destructor Documentation	559
10.51.2.1 midi_measures() [1/2]	559
10.51.2.2 midi_measures() [2/2]	559
10.51.3 Member Function Documentation	559
10.51.3.1 measures() [1/2]	559
10.51.3.2 measures() [2/2]	559
10.51.3.3 beats() [1/2]	560
10.51.3.4 beats() [2/2]	560
10.51.3.5 divisions() [1/2]	560
10.51.3.6 divisions() [2/2]	560
10.51.4 Field Documentation	560

10.51.4.1 m_measures	560
10.51.4.2 m_beats	561
10.51.4.3 m_divisions	561
10.52seq64::midi_message Class Reference	561
10.52.1 Detailed Description	562
10.52.2 Member Typedef Documentation	562
10.52.2.1 container	562
10.52.3 Constructor & Destructor Documentation	562
10.52.3.1 midi_message()	562
10.52.4 Member Function Documentation	562
10.52.4.1 operator[]()	562
10.52.4.2 array()	562
10.52.4.3 count()	563
10.52.4.4 empty()	563
10.52.4.5 push()	563
10.52.4.6 timestamp() [1/2]	563
10.52.4.7 timestamp() [2/2]	563
10.52.4.8 is_sysex()	563
10.52.4.9 show()	563
10.52.5 Field Documentation	563
10.52.5.1 m_bytes	564
10.52.5.2 m_timestamp	564
10.53seq64::midi_out_alsa Class Reference	564
10.53.1 Constructor & Destructor Documentation	566
10.53.1.1 midi_out_alsa()	566
10.54seq64::midi_out_jack Class Reference	566
10.54.1 Constructor & Destructor Documentation	568
10.54.1.1 midi_out_jack()	568
10.54.1.2 ~midi_out_jack()	568
10.54.2 Member Function Documentation	568

10.54.2.1 open_client()	568
10.55seq64::midi_port_info Class Reference	568
10.55.1 Constructor & Destructor Documentation	569
10.55.1.1 midi_port_info()	569
10.55.2 Member Function Documentation	569
10.55.2.1 add() [1/2]	569
10.55.2.2 add() [2/2]	570
10.55.2.3 clear()	570
10.55.2.4 get_port_count()	570
10.55.2.5 get_bus_id()	570
10.55.2.6 get_bus_name()	570
10.55.2.7 get_port_id()	570
10.55.2.8 get_port_name()	570
10.55.2.9 get_input()	571
10.55.2.10get_virtual()	571
10.55.2.11get_system()	571
10.55.2.12get_queue_number()	571
10.55.2.13connect_name()	571
10.55.3 Field Documentation	571
10.55.3.1 m_port_count	571
10.55.3.2 m_port_container	572
10.56seq64::midi_queue Class Reference	572
10.56.1 Detailed Description	572
10.56.2 Constructor & Destructor Documentation	572
10.56.2.1 midi_queue()	572
10.56.2.2 ~midi_queue()	573
10.56.3 Member Function Documentation	573
10.56.3.1 empty()	573
10.56.3.2 count()	573
10.56.3.3 full()	573

10.56.3.4 add()	573
10.56.3.5 pop()	573
10.56.3.6 pop_front()	574
10.56.3.7 allocate()	574
10.56.3.8 deallocate()	574
10.56.3.9 front()	574
10.56.4 Field Documentation	574
10.56.4.1 m_front	574
10.56.4.2 m_back	574
10.56.4.3 m_size	574
10.56.4.4 m_ring_size	575
10.56.4.5 m_ring	575
10.57seq64::midi_splitter Class Reference	575
10.57.1 Detailed Description	576
10.57.2 Constructor & Destructor Documentation	576
10.57.2.1 midi_splitter()	576
10.57.2.2 ~midi_splitter()	577
10.57.3 Member Function Documentation	577
10.57.3.1 log_main_sequence()	577
10.57.3.2 initialize()	577
10.57.3.3 increment()	577
10.57.3.4 split()	577
10.57.3.5 ppqn()	578
10.57.3.6 count()	578
10.57.3.7 split_channel()	578
10.57.4 Field Documentation	579
10.57.4.1 m_ppqn	579
10.57.4.2 m_use_default_ppqn	579
10.57.4.3 m_smf0_channels_count	579
10.57.4.4 m_smf0_channels	579

10.57.4.5 m_smf0_main_sequence	579
10.57.4.6 m_smf0_seq_number	580
10.58seq64::midi_timing Class Reference	580
10.58.1 Detailed Description	581
10.58.2 Constructor & Destructor Documentation	581
10.58.2.1 midi_timing() [1/2]	581
10.58.2.2 midi_timing() [2/2]	581
10.58.3 Member Function Documentation	581
10.58.3.1 beats_per_minute() [1/2]	581
10.58.3.2 beats_per_minute() [2/2]	581
10.58.3.3 beats_per_measure() [1/2]	582
10.58.3.4 beats_per_measure() [2/2]	582
10.58.3.5 beat_width() [1/2]	582
10.58.3.6 beat_width() [2/2]	582
10.58.3.7 ppqn() [1/2]	582
10.58.3.8 ppqn() [2/2]	583
10.58.4 Field Documentation	583
10.58.4.1 m_beats_per_minute	583
10.58.4.2 m_beats_per_measure	583
10.58.4.3 m_beat_width	583
10.58.4.4 m_ppqn	583
10.59seq64::midi_vector Class Reference	584
10.59.1 Member Typedef Documentation	585
10.59.1.1 CharVector	585
10.59.2 Constructor & Destructor Documentation	585
10.59.2.1 midi_vector()	585
10.59.2.2 ~midi_vector()	586
10.59.3 Member Function Documentation	586
10.59.3.1 size()	586
10.59.3.2 done()	586

10.59.3.3 put()	586
10.59.3.4 get()	587
10.59.3.5 clear()	587
10.59.4 Field Documentation	587
10.59.4.1 m_char_vector	587
10.60seq64::midibase Class Reference	588
10.60.1 Constructor & Destructor Documentation	592
10.60.1.1 midibase()	593
10.60.1.2 ~midibase()	594
10.60.2 Member Function Documentation	594
10.60.2.1 show_bus_values()	594
10.60.2.2 show_clock()	594
10.60.2.3 display_name() [1/2]	594
10.60.2.4 bus_name() [1/2]	594
10.60.2.5 port_name() [1/2]	594
10.60.2.6 connect_name()	595
10.60.2.7 get_bus_index()	595
10.60.2.8 get_bus_id()	595
10.60.2.9 get_port_id()	595
10.60.2.10ppqn()	595
10.60.2.11bpm()	595
10.60.2.12match()	595
10.60.2.13s_virtual_port() [1/2]	596
10.60.2.14s_virtual_port() [2/2]	596
10.60.2.15s_input_port() [1/2]	596
10.60.2.16s_output_port()	596
10.60.2.17s_input_port() [2/2]	596
10.60.2.18s_system_port()	596
10.60.2.19set_system_port_flag()	596
10.60.2.20set_clock()	596

10.60.2.21	<code>get_clock()</code>	597
10.60.2.22	<code>port_disabled()</code>	597
10.60.2.23	<code>clock_enabled()</code>	597
10.60.2.24	<code>get_input()</code>	597
10.60.2.25	<code>set_input_status()</code>	597
10.60.2.26	<code>queue_number()</code>	597
10.60.2.27	<code>set_bus_id()</code>	598
10.60.2.28	<code>set_name()</code>	598
10.60.2.29	<code>set_alt_name()</code>	598
10.60.2.30	<code>set_multi_name()</code>	599
10.60.2.31	<code>set_clock_mod()</code>	599
10.60.2.32	<code>get_clock_mod()</code>	600
10.60.2.33	<code>poll_for_midi()</code>	600
10.60.2.34	<code>get_midi_event()</code>	600
10.60.2.35	<code>nit_out()</code>	600
10.60.2.36	<code>nit_in()</code>	601
10.60.2.37	<code>deinit_in()</code>	601
10.60.2.38	<code>nit_out_sub()</code>	601
10.60.2.39	<code>nit_in_sub()</code>	601
10.60.2.40	<code>play()</code>	601
10.60.2.41	<code>sysex()</code>	602
10.60.2.42	<code>flush()</code>	602
10.60.2.43	<code>start()</code>	602
10.60.2.44	<code>stop()</code>	602
10.60.2.45	<code>clock()</code>	602
10.60.2.46	<code>continue_from()</code>	603
10.60.2.47	<code>nit_clock()</code>	603
10.60.2.48	<code>print()</code>	603
10.60.2.49	<code>set_input()</code>	603
10.60.2.50	<code>display_name()</code> [2/2]	604

10.60.2.51	bus_name() [2/2]	604
10.60.2.52	port_name() [2/2]	604
10.60.2.53	set_port_id()	604
10.60.2.54	api_poll_for_midi()	604
10.60.2.55	api_get_midi_event()	605
10.60.2.56	api_init_in_sub()	605
10.60.2.57	api_init_out_sub()	605
10.60.2.58	api_deinit_in()	605
10.60.2.59	api_play()	605
10.60.2.60	api_sysex()	605
10.60.2.61	api_flush()	606
10.60.2.62	api_init_in()	606
10.60.2.63	api_init_out()	606
10.60.2.64	api_continue_from()	606
10.60.2.65	api_start()	606
10.60.2.66	api_stop()	606
10.60.2.67	api_clock()	607
10.60.3	Friends And Related Function Documentation	607
10.60.3.1	mastermidibus	607
10.60.4	Field Documentation	607
10.60.4.1	m_clock_mod	607
10.60.4.2	m_bus_index	607
10.60.4.3	m_bus_id	607
10.60.4.4	m_port_id	607
10.60.4.5	m_clock_type	608
10.60.4.6	m_inputing	608
10.60.4.7	m_ppqn	608
10.60.4.8	m_bpm	608
10.60.4.9	m_queue	608
10.60.4.10	m_display_name	608

10.60.4.11m_bus_name	608
10.60.4.12m_port_name	609
10.60.4.13m_lasttick	609
10.60.4.14m_is_virtual_port	609
10.60.4.15m_is_input_port	609
10.60.4.16m_is_system_port	609
10.60.4.17m_mutex	609
10.61seq64::midibus Class Reference	610
10.61.1 Constructor & Destructor Documentation	611
10.61.1.1 midibus()	611
10.61.1.2 ~midibus()	612
10.61.2 Member Function Documentation	612
10.61.2.1 api_connect()	612
10.61.2.2 api_init_in()	612
10.61.2.3 api_init_in_sub()	612
10.61.2.4 api_init_out()	612
10.61.2.5 api_init_out_sub()	612
10.61.2.6 api_deinit_in()	613
10.61.2.7 api_get_midi_event()	613
10.61.2.8 api_poll_for_midi()	613
10.61.2.9 api_continue_from()	613
10.61.2.10api_start()	613
10.61.2.11api_stop()	613
10.61.2.12api_clock()	614
10.61.2.13api_play()	614
10.61.3 Friends And Related Function Documentation	614
10.61.3.1 mastermidibus	614
10.61.4 Field Documentation	614
10.61.4.1 m_rt_midi	614
10.61.4.2 m_master_info	614

10.62seq64::midifile Class Reference	614
10.62.1 Detailed Description	617
10.62.2 Constructor & Destructor Documentation	618
10.62.2.1 midifile()	618
10.62.2.2 ~midifile()	618
10.62.3 Member Function Documentation	618
10.62.3.1 parse()	619
10.62.3.2 write()	620
10.62.3.3 write_song()	620
10.62.3.4 error_message()	620
10.62.3.5 error_is_fatal()	620
10.62.3.6 ppqn() [1/2]	621
10.62.3.7 get_file_pos()	621
10.62.3.8 initialize_sequence()	621
10.62.3.9 finalize_sequence()	621
10.62.3.10clear_errors()	621
10.62.3.11ppqn() [2/2]	621
10.62.3.12at_end()	621
10.62.3.13grab_input_stream()	621
10.62.3.14parse_smf_0()	622
10.62.3.15parse_smf_1()	622
10.62.3.16parse_prop_header()	624
10.62.3.17parse_proprietary_track()	625
10.62.3.18checklen()	626
10.62.3.19add_trigger()	626
10.62.3.20read_seek()	626
10.62.3.21read_long()	627
10.62.3.22read_short()	627
10.62.3.23read_byte()	627
10.62.3.24read_varinum()	627

10.62.3.25	read_byte_array() [1/2]	627
10.62.3.26	read_byte_array() [2/2]	628
10.62.3.27	read_gap()	628
10.62.3.28	write_long()	628
10.62.3.29	write_triple()	629
10.62.3.30	write_short()	629
10.62.3.31	write_byte()	629
10.62.3.32	write_varinum()	630
10.62.3.33	write_track_name()	630
10.62.3.34	read_track_name()	631
10.62.3.35	write_seq_number()	631
10.62.3.36	read_seq_number()	631
10.62.3.37	write_track_end()	632
10.62.3.38	write_header()	632
10.62.3.39	write_prop_header()	632
10.62.3.40	write_proprietary_track()	633
10.62.3.41	varinum_size()	633
10.62.3.42	prop_item_size()	635
10.62.3.43	track_name_size()	635
10.62.3.44	set_error()	635
10.62.3.45	set_error_dump() [1/2]	636
10.62.3.46	set_error_dump() [2/2]	636
10.62.3.47	write_track()	637
10.62.3.48	seq_number_size()	637
10.62.3.49	track_end_size()	637
10.62.3.50	s_sysex_special_id()	637
10.62.4	Field Documentation	637
10.62.4.1	m_mutex	637
10.62.4.2	m_file_size	637
10.62.4.3	m_error_message	638

10.62.4.4 m_error_is_fatal	638
10.62.4.5 m_disable_reported	638
10.62.4.6 m_pos	638
10.62.4.7 m_name	638
10.62.4.8 m_data	638
10.62.4.9 m_char_list	638
10.62.4.10m_new_format	639
10.62.4.11m_global_bgsequence	639
10.62.4.12m_ppqn	639
10.62.4.13m_use_default_ppqn	639
10.62.4.14m_smf0_splitter	639
10.63seq64::mutex Class Reference	640
10.63.1 Constructor & Destructor Documentation	641
10.63.1.1 mutex()	641
10.63.2 Member Function Documentation	641
10.63.2.1 lock()	641
10.63.2.2 unlock()	641
10.63.3 Field Documentation	641
10.63.3.1 sm_recursive_mutex	641
10.63.3.2 m_mutex_lock	641
10.64seq64::editable_event::name_value_t Struct Reference	641
10.64.1 Field Documentation	642
10.64.1.1 event_value	642
10.64.1.2 event_name	642
10.65seq64::options Class Reference	642
10.65.1 Member Enumeration Documentation	643
10.65.1.1 button	644
10.65.2 Constructor & Destructor Documentation	644
10.65.2.1 options()	644
10.65.3 Member Function Documentation	644

10.65.3.1 perf()	644
10.65.3.2 clock_callback_off()	645
10.65.3.3 clock_callback_on()	645
10.65.3.4 clock_callback_mod()	645
10.65.3.5 clock_callback_disable()	645
10.65.3.6 clock_mod_callback()	645
10.65.3.7 edit_tempo_track_number()	645
10.65.3.8 log_tempo_track_number()	646
10.65.3.9 input_callback()	646
10.65.3.10 filter_callback()	646
10.65.3.11 transport_callback()	646
10.65.3.12 mouse_seq24_callback()	646
10.65.3.13 mouse_fruity_callback()	646
10.65.3.14 mouse_mod4_callback()	646
10.65.3.15 mouse_snap_split_callback()	647
10.65.3.16 mouse_click_edit_callback()	647
10.65.3.17 ash_support_callback()	647
10.65.3.18 add_midi_clock_page()	647
10.65.3.19 add_midi_input_page()	647
10.65.3.20 add_keyboard_page()	647
10.65.3.21 add_extended_keys_page()	647
10.65.3.22 add_mouse_page()	648
10.65.3.23 add_jack_sync_page()	648
10.65.4 Field Documentation	648
10.65.4.1 m_tooltips	648
10.65.4.2 m_mainperf	648
10.65.4.3 m_button_ok	648
10.65.4.4 m_button_jack_transport	648
10.65.4.5 m_button_jack_master	648
10.65.4.6 m_button_jack_master_cond	649

10.65.4.7 m_button_jack_connect	649
10.65.4.8 m_button_jack_disconnect	649
10.65.4.9 m_notebook	649
10.66seq64::optionsfile Class Reference	649
10.66.1 Detailed Description	651
10.66.2 Constructor & Destructor Documentation	651
10.66.2.1 optionsfile()	651
10.66.2.2 ~optionsfile()	651
10.66.3 Member Function Documentation	651
10.66.3.1 parse()	651
10.66.3.2 parse_mute_group_section()	653
10.66.3.3 write()	653
10.66.3.4 error_message()	654
10.67seq64::perfedit Class Reference	654
10.67.1 Detailed Description	659
10.67.2 Constructor & Destructor Documentation	659
10.67.2.1 perfedit()	659
10.67.2.2 ~perfedit()	660
10.67.3 Member Function Documentation	660
10.67.3.1 init_before_show()	660
10.67.3.2 enqueue_draw()	660
10.67.3.3 zoom_check()	660
10.67.3.4 enregister_peer()	661
10.67.3.5 set_zoom()	661
10.67.3.6 get_toggle_jack()	661
10.67.3.7 toggle_jack()	661
10.67.3.8 rewind()	662
10.67.3.9 fast_forward()	662
10.67.3.10set_follow_transport()	662
10.67.3.11toggle_follow_transport()	662

10.67.3.12	set_jack_mode()	662
10.67.3.13	set_transpose()	662
10.67.3.14	transpose_button_callback()	663
10.67.3.15	set_beats_per_bar()	663
10.67.3.16	set_beat_width()	663
10.67.3.17	set_snap()	664
10.67.3.18	set_guides()	664
10.67.3.19	grow()	664
10.67.3.20	set_looped()	664
10.67.3.21	expand()	664
10.67.3.22	collapse()	664
10.67.3.23	copy()	665
10.67.3.24	undo()	665
10.67.3.25	redo()	665
10.67.3.26	popup_menu()	665
10.67.3.27	draw_sequences()	665
10.67.3.28	timeout()	665
10.67.3.29	set_image()	665
10.67.3.30	start_playing()	666
10.67.3.31	pause_playing()	666
10.67.3.32	stop_playing()	666
10.67.3.33	toggle_playing()	666
10.67.3.34	on_realize()	666
10.67.3.35	on_key_press_event()	666
10.67.3.36	on_key_release_event()	667
10.67.3.37	on_delete_event()	667
10.67.4	Friends And Related Function Documentation	667
10.67.4.1	update_perfeddit_sequences	667
10.67.5	Field Documentation	667
10.67.5.1	m_peer_perfeddit	667

10.67.5.2 m_table	668
10.67.5.3 m_vadjust	668
10.67.5.4 m_hadjust	668
10.67.5.5 m_vscroll	668
10.67.5.6 m_hscroll	668
10.67.5.7 m_perfnames	668
10.67.5.8 m_perfroll	668
10.67.5.9 m_perftime	668
10.67.5.10m_menu_snap	669
10.67.5.11m_menu_xpose	669
10.67.5.12m_button_xpose	669
10.67.5.13m_entry_xpose	669
10.67.5.14m_image_play	669
10.67.5.15m_button_snap	669
10.67.5.16m_entry_snap	669
10.67.5.17m_button_stop	669
10.67.5.18m_button_play	670
10.67.5.19m_button_loop	670
10.67.5.20m_button_expand	670
10.67.5.21m_button_collapse	670
10.67.5.22m_button_copy	670
10.67.5.23m_button_grow	670
10.67.5.24m_button_undo	670
10.67.5.25m_button_redo	670
10.67.5.26m_button_jack	671
10.67.5.27m_button_follow	671
10.67.5.28m_button_bpm	671
10.67.5.29m_entry_bpm	671
10.67.5.30m_button_bw	671
10.67.5.31m_entry_bw	671

10.67.5.32	m_hbox	671
10.67.5.33	m_hlbox	671
10.67.5.34	m_menu_bpm	672
10.67.5.35	m_menu_bw	672
10.67.5.36	m_snap	672
10.67.5.37	m_bpm	672
10.67.5.38	m_bw	672
10.67.5.39	m_ppqn	672
10.67.5.40	m_is_running	672
10.67.5.41	m_standard_bpm	672
10.68	seq64::perfnames Class Reference	673
10.68.1	Detailed Description	675
10.68.2	Constructor & Destructor Documentation	675
10.68.2.1	perfnames()	675
10.68.2.2	~perfnames()	676
10.68.3	Member Function Documentation	676
10.68.3.1	redraw_dirty_sequences()	676
10.68.3.2	enqueue_draw()	676
10.68.3.3	convert_y()	676
10.68.3.4	draw_sequences()	676
10.68.3.5	draw_sequence()	677
10.68.3.6	change_vert()	677
10.68.3.7	redraw()	677
10.68.3.8	on_realize()	677
10.68.3.9	on_expose_event()	678
10.68.3.10	on_button_press_event()	678
10.68.3.11	on_button_release_event()	678
10.68.3.12	on_size_allocate()	679
10.68.3.13	on_scroll_event()	679
10.68.4	Friends And Related Function Documentation	679

10.68.4.1	perfedir	679
10.68.5	Field Documentation	679
10.68.5.1	m_parent	680
10.68.5.2	m_names_chars	680
10.68.5.3	m_char_w	680
10.68.5.4	m_setbox_w	680
10.68.5.5	m_namebox_w	680
10.68.5.6	m_names_x	680
10.68.5.7	m_names_y	680
10.68.5.8	m_xy_offset	681
10.68.5.9	m_seqs_in_set	681
10.68.5.10	m_sequence_max	681
10.68.5.11	m_sequence_offset	681
10.68.5.12	m_sequence_active	681
10.69	seq64::perform Class Reference	681
10.69.1	Detailed Description	698
10.69.2	Member Enumeration Documentation	699
10.69.2.1	action_t	699
10.69.2.2	record_tempo_op_t	699
10.69.2.3	mute_op_t	699
10.69.2.4	ff_rw_button_t	700
10.69.3	Constructor & Destructor Documentation	700
10.69.3.1	perform()	700
10.69.3.2	~perform()	700
10.69.4	Member Function Documentation	701
10.69.4.1	is_modified() [1/2]	701
10.69.4.2	modify()	701
10.69.4.3	ppqn() [1/2]	701
10.69.4.4	bpm()	701
10.69.4.5	sequence_count()	701

10.69.4.6 sequence_high()	701
10.69.4.7 sequence_max()	702
10.69.4.8 group_max()	702
10.69.4.9 is_control_status()	702
10.69.4.10midi_mute_group_present() [1/2]	702
10.69.4.11set_edit_sequence()	702
10.69.4.12inset_edit_sequence()	702
10.69.4.13s_edit_sequence()	703
10.69.4.14get_beats_per_bar()	703
10.69.4.15set_beats_per_bar()	703
10.69.4.16get_beat_width()	703
10.69.4.17set_beat_width()	703
10.69.4.18get_tempo_track_number()	704
10.69.4.19set_tempo_track_number()	704
10.69.4.20clocks_per_metronome() [1/2]	704
10.69.4.21clocks_per_metronome() [2/2]	704
10.69.4.22set_32nds_per_quarter()	704
10.69.4.23get_32nds_per_quarter()	705
10.69.4.24us_per_quarter_note() [1/2]	705
10.69.4.25us_per_quarter_note() [2/2]	705
10.69.4.26gui() [1/2]	705
10.69.4.27gui() [2/2]	705
10.69.4.28keys() [1/2]	705
10.69.4.29keys() [2/2]	705
10.69.4.30master_bus()	706
10.69.4.31filter_by_channel()	706
10.69.4.32s_running() [1/2]	706
10.69.4.33s_pattern_playing() [1/2]	706
10.69.4.34s_pattern_playing() [2/2]	706
10.69.4.35toggle_song_start_mode()	706

10.69.4.36song_start_mode() [1/2]	706
10.69.4.37song_start_mode() [2/2]	707
10.69.4.38s_jack_running()	707
10.69.4.39s_jack_master()	707
10.69.4.40enregister()	707
10.69.4.41toggle_jack_mode()	707
10.69.4.42set_jack_mode()	707
10.69.4.43get_toggle_jack()	708
10.69.4.44set_jack_stop_tick()	708
10.69.4.45combine_bytes()	708
10.69.4.46FF_rewind()	708
10.69.4.47FF_RW_timeout()	709
10.69.4.48start_from_perfed() [1/2]	709
10.69.4.49start_from_perfed() [2/2]	709
10.69.4.50set_follow_transport()	709
10.69.4.51get_follow_transport()	709
10.69.4.52toggle_follow_transport()	709
10.69.4.53follow_progress()	710
10.69.4.54set_reposition()	710
10.69.4.55ff_rw_type() [1/2]	710
10.69.4.56ff_rw_type() [2/2]	710
10.69.4.57rewind()	710
10.69.4.58fast_forward()	710
10.69.4.59reposition()	711
10.69.4.60set_sequence_input()	711
10.69.4.61set_recording() [1/2]	711
10.69.4.62set_recording() [2/2]	711
10.69.4.63set_quantized_recording() [1/2]	712
10.69.4.64set_quantized_recording() [2/2]	712
10.69.4.65set_overwrite_recording()	712

10.69.4.66set_thru() [1/2]	713
10.69.4.67set_thru() [2/2]	713
10.69.4.68selected_trigger()	714
10.69.4.69clear_all()	714
10.69.4.70launch()	714
10.69.4.71finish()	715
10.69.4.72new_sequence()	715
10.69.4.73add_sequence()	715
10.69.4.74delete_sequence()	716
10.69.4.75s_sequence_in_edit()	716
10.69.4.76print_busses()	717
10.69.4.77get_tick()	717
10.69.4.78set_tick()	717
10.69.4.79get_jack_tick()	717
10.69.4.80set_jack_tick()	717
10.69.4.81set_left_tick()	718
10.69.4.82get_left_tick()	718
10.69.4.83set_start_tick()	718
10.69.4.84get_start_tick()	718
10.69.4.85set_right_tick()	718
10.69.4.86get_right_tick()	720
10.69.4.87left_right_size()	720
10.69.4.88s_active()	720
10.69.4.89apply_song_transpose()	720
10.69.4.90set_transpose()	721
10.69.4.91get_transpose()	721
10.69.4.92get_beats_per_minute()	721
10.69.4.93reload_mute_groups()	721
10.69.4.94clear_mute_groups()	721
10.69.4.95set_sequence_control_status()	722

10.69.4.96	<code>unset_sequence_control_status()</code>	722
10.69.4.97	<code>unset_queued_replace()</code>	722
10.69.4.98	<code>sequence_playing_toggle()</code>	723
10.69.4.99	<code>sequence_playing_change()</code>	723
10.69.4.100	<code>set_keep_queue()</code>	724
10.69.4.101	<code>get_keep_queue()</code>	724
10.69.4.102	<code>sequence_playing_on()</code>	724
10.69.4.103	<code>sequence_playing_off()</code>	724
10.69.4.104	<code>mute_all_tracks()</code>	724
10.69.4.105	<code>toggle_all_tracks()</code>	725
10.69.4.106	<code>armed_saved()</code>	725
10.69.4.107	<code>toggle_playing_tracks()</code>	725
10.69.4.108	<code>mute_screenset()</code>	725
10.69.4.109	<code>output_func()</code>	725
10.69.4.110	<code>input_func()</code>	726
10.69.4.111	<code>set_group_mute_state()</code>	728
10.69.4.112	<code>get_group_mute_state()</code>	728
10.69.4.113	<code>mute_group_offset()</code>	728
10.69.4.114	<code>screenset_offset()</code> [1/2]	730
10.69.4.115	<code>slot_number()</code>	730
10.69.4.116	<code>save_playing_state()</code>	730
10.69.4.117	<code>restore_playing_state()</code>	730
10.69.4.118	<code>save_current_screenset()</code>	731
10.69.4.119	<code>clear_current_screenset()</code>	732
10.69.4.120	<code>key_name()</code>	732
10.69.4.121	<code>get_key_events()</code>	732
10.69.4.122	<code>get_key_count()</code>	732
10.69.4.123	<code>get_key_groups()</code>	732
10.69.4.124	<code>get_key_events_rev()</code>	733
10.69.4.125	<code>get_key_groups_rev()</code>	733

10.69.4.125	show_ui_sequence_key() [1/2]	733
10.69.4.127	show_ui_sequence_key() [2/2]	733
10.69.4.128	show_ui_sequence_number() [1/2]	733
10.69.4.129	show_ui_sequence_number() [2/2]	733
10.69.4.130	lookup_keyevent_key()	734
10.69.4.131	lookup_slot_key()	734
10.69.4.132	lookup_keyevent_seq()	734
10.69.4.133	lookup_keygroup_key()	735
10.69.4.134	lookup_keygroup_group()	735
10.69.4.135	start_playing()	735
10.69.4.136	pause_playing()	736
10.69.4.137	stop_playing()	737
10.69.4.138	start_key()	737
10.69.4.139	pause_key()	737
10.69.4.140	stop_key()	738
10.69.4.141	learn_toggle()	738
10.69.4.142	decrement_beats_per_minute()	738
10.69.4.143	increment_beats_per_minute()	738
10.69.4.144	page_decrement_beats_per_minute()	738
10.69.4.145	page_increment_beats_per_minute()	739
10.69.4.146	decrement_screenset()	739
10.69.4.147	increment_screenset()	739
10.69.4.148	highlight()	740
10.69.4.149	smf_0()	740
10.69.4.150	get_sequence() [1/2]	740
10.69.4.151	get_sequence() [2/2]	740
10.69.4.152	sequence_key()	741
10.69.4.153	sequence_label() [1/2]	741
10.69.4.154	sequence_label() [2/2]	742
10.69.4.155	sequence_title()	742

10.69.4.156	main_window_title()	743
10.69.4.157	sequence_window_title()	743
10.69.4.158	set_input_bus()	743
10.69.4.159	set_clock_bus()	744
10.69.4.160	mainwnd_key_event()	744
10.69.4.161	keyboard_control_press()	745
10.69.4.162	keyboard_group_c_status_press()	745
10.69.4.163	keyboard_group_c_status_release()	746
10.69.4.164	keyboard_group_press()	746
10.69.4.165	keyboard_group_release()	746
10.69.4.166	keyboard_group_action()	746
10.69.4.167	perfroll_key_event()	746
10.69.4.168	playback_key_event()	747
10.69.4.169	clear_sequence_triggers()	747
10.69.4.170	print_triggers()	747
10.69.4.171	move_triggers()	748
10.69.4.172	copy_triggers()	748
10.69.4.173	push_trigger_undo()	748
10.69.4.174	pop_trigger_undo()	748
10.69.4.175	pop_trigger_redo()	749
10.69.4.176	get_trigger_state()	749
10.69.4.177	add_trigger()	749
10.69.4.178	delete_trigger()	749
10.69.4.179	add_or_delete_trigger()	750
10.69.4.180	split_trigger()	750
10.69.4.181	paste_trigger()	750
10.69.4.182	paste_or_split_trigger()	750
10.69.4.183	intersect_triggers()	751
10.69.4.184	get_max_trigger()	751
10.69.4.185	dirty_main()	751

10.69.4.186_dirty_edit()	752
10.69.4.187_dirty_perf()	752
10.69.4.188_dirty_names()	752
10.69.4.189_exportable()	753
10.69.4.190_get_screenset()	753
10.69.4.191_screenset()	753
10.69.4.192_get_playing_screenset()	754
10.69.4.193_toggle_other_seqs()	754
10.69.4.194_toggle_other_names()	754
10.69.4.195_are_any_armed()	754
10.69.4.196_max_sets() [1/2]	755
10.69.4.197_seqs_in_set()	755
10.69.4.198_is_long_recording() [1/2]	755
10.69.4.199_long_record_snap() [1/2]	755
10.69.4.200_resume_note_ons() [1/2]	755
10.69.4.201_resume_note_ons() [2/2]	755
10.69.4.202_select_trigger()	755
10.69.4.203_deselect_all_triggers()	756
10.69.4.204_get_bank_name()	756
10.69.4.205_set_looping()	756
10.69.4.206_get_sequence_color()	756
10.69.4.207_set_sequence_color()	756
10.69.4.208_save_undo()	757
10.69.4.209_get_have_undo()	757
10.69.4.210_save_redo()	757
10.69.4.211_set_have_redo()	757
10.69.4.212_seq_edit_mode() [1/2]	757
10.69.4.213_seq_edit_mode() [2/2]	757
10.69.4.214_current_screenset_notepad()	758
10.69.4.215_set_screenset_notepad() [1/2]	758

10.69.4.216	set_screenset_notepad() [2/2]	758
10.69.4.217	start()	758
10.69.4.218	stop()	759
10.69.4.219	start_jack()	759
10.69.4.220	stop_jack()	759
10.69.4.221	long_recording_stop()	759
10.69.4.222	long_recording() [2/2]	760
10.69.4.223	long_record_snap() [2/2]	760
10.69.4.224	playback_mode() [1/2]	760
10.69.4.225	playback_mode() [2/2]	760
10.69.4.226	group_learning()	760
10.69.4.227	set_beats_per_minute()	760
10.69.4.228	panic()	761
10.69.4.229	popn() [2/2]	761
10.69.4.230	collapse()	761
10.69.4.231	copy()	761
10.69.4.232	expand()	761
10.69.4.233	midi_control_toggle()	761
10.69.4.234	midi_control_on()	762
10.69.4.235	midi_control_off()	762
10.69.4.236	midi_control_event()	762
10.69.4.237	midi_control_record()	763
10.69.4.238	handle_midi_control()	763
10.69.4.239	handle_midi_control_ex()	764
10.69.4.240	handle_midi_control_event()	765
10.69.4.241	get_screenset_notepad()	765
10.69.4.242	any_group_unmutes()	765
10.69.4.243	print_group_unmutes()	766
10.69.4.244	mute_group_tracks()	766
10.69.4.245	select_and_mute_group()	766

10.69.4.246	set_song_mute()	766
10.69.4.247	set_playing_screensset()	767
10.69.4.248	set_mode_group_mute()	767
10.69.4.249	unset_mode_group_mute()	767
10.69.4.250	select_group_mute()	767
10.69.4.251	set_mode_group_learn()	768
10.69.4.252	unset_mode_group_learn()	768
10.69.4.253	load_mute_group()	768
10.69.4.254	save_mute_group()	769
10.69.4.255	set_and_copy_mute_group()	769
10.69.4.256	activate()	769
10.69.4.257	position_jack()	770
10.69.4.258	off_sequences()	770
10.69.4.259	enqueue_sequences()	770
10.69.4.260	all_notes_off()	770
10.69.4.261	set_active()	771
10.69.4.262	set_was_active()	771
10.69.4.263	set_sequences()	771
10.69.4.264	play()	771
10.69.4.265	set_orig_ticks()	772
10.69.4.266	max_active_set()	772
10.69.4.267	launch_input_thread()	772
10.69.4.268	launch_output_thread()	773
10.69.4.269	init_jack_transport()	773
10.69.4.270	reinit_jack_transport()	773
10.69.4.271	seq_in_playing_screen()	773
10.69.4.272	is_modified() [2/2]	774
10.69.4.273	valid_midi_control_seq()	774
10.69.4.274	max_sets() [2/2]	774
10.69.4.275	screensset_valid()	774

10.69.4.276	is_running() [2/2]	775
10.69.4.277	screenset_offset() [2/2]	775
10.69.4.278	seq_valid()	775
10.69.4.279	mseq_valid()	776
10.69.4.280	stall_sequence()	776
10.69.4.281	inner_start()	777
10.69.4.282	inner_stop()	777
10.69.4.283	amp_track()	777
10.69.4.284	amp_group()	778
10.69.4.285	set_key_event()	778
10.69.4.286	set_key_group()	779
10.69.4.287	get_current_tempo()	779
10.69.4.288	create_master_bus()	779
10.69.4.289	reallocate_clocks()	780
10.69.4.290	add_clock()	780
10.69.4.291	set_clock()	780
10.69.4.292	get_clock()	780
10.69.4.293	add_input()	780
10.69.4.294	set_input()	781
10.69.4.295	get_input()	781
10.69.4.296	input_system_port()	781
10.69.4.297	midi_mute_group_present() [2/2]	781
10.69.5	Friends And Related Function Documentation	781
10.69.5.1	jack_assistant	781
10.69.5.2	keybindentry	781
10.69.5.3	mainwnd	782
10.69.5.4	midifile	782
10.69.5.5	wrkfile	782
10.69.5.6	optionsfile	782
10.69.5.7	options	782

10.69.5.8	perfedit	782
10.69.5.9	perfroll	782
10.69.5.10	sequence	782
10.69.5.11	input_thread_func	782
10.69.5.12	output_thread_func	783
10.69.5.13	jack_sync_callback	783
10.69.5.14	jack_transport_callback	783
10.69.5.15	jack_shutdown	784
10.69.5.16	jack_timebase_callback	784
10.69.5.17	get_current_jack_position	785
10.69.6	Field Documentation	786
10.69.6.1	sm_mc_dummy	786
10.69.6.2	m_song_start_mode	786
10.69.6.3	m_start_from_perfedit	786
10.69.6.4	m_reposition	786
10.69.6.5	m_excell_FF_RW	787
10.69.6.6	m_FF_RW_button_type	787
10.69.6.7	m_mute_group	787
10.69.6.8	m_mute_group_rc	787
10.69.6.9	m_armed_saved	787
10.69.6.10	m_armed_statuses	787
10.69.6.11	m_seqs_in_set	787
10.69.6.12	m_max_groups	788
10.69.6.13	m_tracks_mute_state	788
10.69.6.14	m_mode_group	788
10.69.6.15	m_mode_group_learn	788
10.69.6.16	m_mute_group_selected	788
10.69.6.17	m_midi_mute_group_present	788
10.69.6.18	m_seqs	789
10.69.6.19	m_seqs_active	789

10.69.6.20m_was_active_main	789
10.69.6.21m_was_active_edit	789
10.69.6.22m_was_active_perf	789
10.69.6.23m_was_active_names	789
10.69.6.24m_sequence_state	790
10.69.6.25m_screenset_state	790
10.69.6.26m_queued_replace_slot	790
10.69.6.27m_transpose	790
10.69.6.28m_out_thread	790
10.69.6.29m_in_thread	790
10.69.6.30m_out_thread_launched	790
10.69.6.31m_in_thread_launched	791
10.69.6.32m_is_running	791
10.69.6.33m_is_pattern_playing	791
10.69.6.34m_inputting	791
10.69.6.35m_outputting	791
10.69.6.36m_looping	791
10.69.6.37m_song_recording	791
10.69.6.38m_song_record_snap	792
10.69.6.39m_resume_note_ons	792
10.69.6.40m_current_tick	792
10.69.6.41m_playback_mode	792
10.69.6.42m_ppqn	792
10.69.6.43m_bpm	792
10.69.6.44m_beats_per_bar	792
10.69.6.45m_beat_width	793
10.69.6.46m_tempo_track_number	793
10.69.6.47m_clocks_per_metronome	793
10.69.6.48m_32nds_per_quarter	793
10.69.6.49m_us_per_quarter_note	793

10.69.6.50m_master_bus	793
10.69.6.51m_filter_by_channel	793
10.69.6.52m_master_clocks	794
10.69.6.53m_master_inputs	794
10.69.6.54m_one_measure	794
10.69.6.55m_left_tick	794
10.69.6.56m_right_tick	794
10.69.6.57m_starting_tick	794
10.69.6.58m_tick	794
10.69.6.59m_jack_tick	795
10.69.6.60m_usemidiclock	795
10.69.6.61m_midiclockrunning	795
10.69.6.62m_midiclocktick	795
10.69.6.63m_midiclockpos	795
10.69.6.64m_dont_reset_ticks	795
10.69.6.65m_screenset_notepad	795
10.69.6.66m_midi_cc_toggle	795
10.69.6.67m_midi_cc_on	796
10.69.6.68m_midi_cc_off	796
10.69.6.69m_control_status	796
10.69.6.70m_screenset	796
10.69.6.71m_screenset_offset	796
10.69.6.72m_playscreen	796
10.69.6.73m_playscreen_offset	796
10.69.6.74m_max_sets	797
10.69.6.75m_sequence_count	797
10.69.6.76m_sequence_max	797
10.69.6.77m_sequence_high	797
10.69.6.78m_edit_sequence	797
10.69.6.79m_is_modified	797

10.69.6.80	m_condition_var	798
10.69.6.81	m_jack_asst	798
10.69.6.82	m_have_undo	798
10.69.6.83	m_undo_vect	798
10.69.6.84	m_have_redo	798
10.69.6.85	m_redo_vect	798
10.69.6.86	m_notify	798
10.69.6.87	m_gui_support	798
10.70	seq64::performcallback Struct Reference	799
10.70.1	Detailed Description	800
10.70.2	Member Function Documentation	800
10.70.2.1	on_grouplearnchange()	800
10.71	seq64::perffroll Class Reference	800
10.71.1	Constructor & Destructor Documentation	805
10.71.1.1	perffroll()	805
10.71.1.2	~perffroll()	806
10.71.2	Member Function Documentation	806
10.71.2.1	set_guides()	806
10.71.2.2	update_sizes()	806
10.71.2.3	init_before_show()	807
10.71.2.4	fill_background_pixmap()	807
10.71.2.5	increment_size()	807
10.71.2.6	draw_all()	807
10.71.2.7	follow_progress()	808
10.71.2.8	redraw_progress()	808
10.71.2.9	draw_progress()	808
10.71.2.10	redraw_dirty_sequences()	808
10.71.2.11	set_ppqn()	808
10.71.2.12	convert_xy()	808
10.71.2.13	convert_x()	809

10.71.2.14	snap_x()	809
10.71.2.15	snap_y()	809
10.71.2.16	draw_sequence_on()	810
10.71.2.17	draw_background_on()	810
10.71.2.18	draw_drawable_row()	810
10.71.2.19	draw_sequence()	810
10.71.2.20	offset_sequence()	811
10.71.2.21	change_horz()	811
10.71.2.22	change_vert()	811
10.71.2.23	split_trigger()	811
10.71.2.24	enqueue_draw()	811
10.71.2.25	set_zoom()	811
10.71.2.26	convert_drop_xy()	812
10.71.2.27	horizontal_adjust()	812
10.71.2.28	vertical_adjust()	812
10.71.2.29	horizontal_set()	812
10.71.2.30	vertical_set()	813
10.71.2.31	activate_adding()	813
10.71.2.32	handle_motion_key()	813
10.71.2.33	s_adding()	813
10.71.2.34	set_adding()	813
10.71.2.35	s_adding_pressed()	813
10.71.2.36	set_adding_pressed()	814
10.71.2.37	growing()	814
10.71.2.38	moving()	814
10.71.2.39	drop_action()	814
10.71.2.40	on_realize()	814
10.71.2.41	on_expose_event()	814
10.71.2.42	on_button_press_event()	815
10.71.2.43	on_button_release_event()	815

10.71.2.44	on_motion_notify_event()	815
10.71.2.45	on_scroll_event()	815
10.71.2.46	on_focus_in_event()	816
10.71.2.47	on_focus_out_event()	816
10.71.2.48	on_size_allocate()	816
10.71.2.49	on_key_press_event()	816
10.71.2.50	on_size_request()	817
10.71.3	Friends And Related Function Documentation	817
10.71.3.1	perfdit	817
10.71.4	Field Documentation	817
10.71.4.1	sm_perfroll_size_box_w	817
10.71.4.2	sm_perfroll_background_x	817
10.71.4.3	sm_perfroll_size_box_click_w	817
10.71.4.4	m_parent	818
10.71.4.5	m_adding	818
10.71.4.6	m_adding_pressed	818
10.71.4.7	m_h_page_increment	818
10.71.4.8	m_v_page_increment	818
10.71.4.9	m_snap_x	818
10.71.4.10	m_snap_y	818
10.71.4.11	m_ppqn	819
10.71.4.12	m_page_factor	819
10.71.4.13	m_divs_per_beat	819
10.71.4.14	m_ticks_per_bar	819
10.71.4.15	m_perf_scale_x	819
10.71.4.16	m_w_scale_x	819
10.71.4.17	m_zoom	819
10.71.4.18	m_names_y	819
10.71.4.19	m_background_x	820
10.71.4.20	m_size_box_w	820

10.71.4.21m_measure_length	820
10.71.4.22m_beat_length	820
10.71.4.23m_old_progress_ticks	820
10.71.4.24m_scroll_page	820
10.71.4.25m_have_button_press	820
10.71.4.26m_4bar_offset	821
10.71.4.27m_sequence_offset	821
10.71.4.28m_roll_length_ticks	821
10.71.4.29m_drop_tick	821
10.71.4.30m_drop_tick_offset	821
10.71.4.31m_drop_sequence	821
10.71.4.32m_sequence_max	821
10.71.4.33m_sequence_active	822
10.71.4.34m_moving	822
10.71.4.35m_growing	822
10.71.4.36m_grow_direction	822
10.72seq64::perftime Class Reference	822
10.72.1 Constructor & Destructor Documentation	825
10.72.1.1 perftime()	825
10.72.1.2 ~perftime()	826
10.72.2 Member Function Documentation	826
10.72.2.1 reset()	826
10.72.2.2 set_scale()	826
10.72.2.3 set_guides()	826
10.72.2.4 increment_size()	827
10.72.2.5 enqueue_draw()	827
10.72.2.6 set_zoom()	827
10.72.2.7 draw_background()	827
10.72.2.8 draw_progress_on_window()	827
10.72.2.9 change_horz()	827

10.72.2.10	set_ppqn()	828
10.72.2.11	tick_to_pixel()	828
10.72.2.12	pixel_to_tick()	828
10.72.2.13	tick_offset()	829
10.72.2.14	update_sizes()	829
10.72.2.15	idle_progress()	829
10.72.2.16	update_pixmap()	829
10.72.2.17	draw_pixmap_on_window()	829
10.72.2.18	on_realize()	829
10.72.2.19	on_expose_event()	829
10.72.2.20	on_button_press_event()	830
10.72.2.21	on_size_allocate()	830
10.72.2.22	on_button_release_event()	830
10.72.2.23	key_press_event()	831
10.72.3	Friends And Related Function Documentation	831
10.72.3.1	perfedit	831
10.72.4	Field Documentation	831
10.72.4.1	m_parent	831
10.72.4.2	m_4bar_offset	831
10.72.4.3	m_tick_offset	831
10.72.4.4	m_ppqn	832
10.72.4.5	m_snap	832
10.72.4.6	m_measure_length	832
10.72.4.7	m_left_marker_tick	832
10.72.4.8	m_right_marker_tick	832
10.72.4.9	m_perf_scale_x	832
10.72.4.10	m_timearea_y	832
10.73	seq64::midi_port_info::port_info_t Struct Reference	832
10.73.1	Detailed Description	833
10.73.2	Field Documentation	833

10.73.2.1 m_client_number	833
10.73.2.2 m_client_name	833
10.73.2.3 m_port_number	833
10.73.2.4 m_port_name	834
10.73.2.5 m_queue_number	834
10.73.2.6 m_is_input	834
10.73.2.7 m_is_virtual	834
10.73.2.8 m_is_system	834
10.74seq64::rc_settings Class Reference	834
10.74.1 Detailed Description	840
10.74.2 Constructor & Destructor Documentation	840
10.74.2.1 rc_settings() [1/2]	840
10.74.2.2 rc_settings() [2/2]	840
10.74.3 Member Function Documentation	840
10.74.3.1 operator=()	840
10.74.3.2 config_filespec()	841
10.74.3.3 user_filespec()	841
10.74.3.4 set_defaults()	841
10.74.3.5 comments_block()	841
10.74.3.6 clear_comments()	842
10.74.3.7 append_comment_line()	842
10.74.3.8 verbose_option() [1/2]	842
10.74.3.9 auto_option_save() [1/2]	842
10.74.3.10 legacy_format() [1/2]	842
10.74.3.11 lash_support() [1/2]	842
10.74.3.12 allow_mod4_mode() [1/2]	842
10.74.3.13 allow_snap_split() [1/2]	843
10.74.3.14 allow_click_edit() [1/2]	843
10.74.3.15 show_midi() [1/2]	843
10.74.3.16 priority() [1/2]	843

10.74.3.17	stats() [1/2]	843
10.74.3.18	pass_sysex() [1/2]	843
10.74.3.19	with_jack_transport() [1/2]	843
10.74.3.20	with_jack_master() [1/2]	843
10.74.3.21	with_jack_master_cond() [1/2]	844
10.74.3.22	with_jack_midi() [1/2]	844
10.74.3.23	with_jack_transport() [2/2]	844
10.74.3.24	with_jack_master() [2/2]	844
10.74.3.25	with_jack_master_cond() [2/2]	844
10.74.3.26	with_jack()	844
10.74.3.27	filter_by_channel() [1/2]	844
10.74.3.28	manual_alsa_ports() [1/2]	845
10.74.3.29	reveal_alsa_ports() [1/2]	845
10.74.3.30	print_keys() [1/2]	845
10.74.3.31	device_ignore() [1/2]	845
10.74.3.32	device_ignore_num() [1/2]	845
10.74.3.33	interaction_method() [1/2]	845
10.74.3.34	mute_group_saving() [1/2]	845
10.74.3.35	filename() [1/2]	845
10.74.3.36	filename() [2/2]	845
10.74.3.37	jack_session_uuid() [1/2]	846
10.74.3.38	last_used_dir() [1/2]	846
10.74.3.39	last_used_dir() [2/2]	846
10.74.3.40	add_recent_file()	846
10.74.3.41	append_recent_file()	847
10.74.3.42	remove_recent_file()	847
10.74.3.43	config_directory() [1/2]	847
10.74.3.44	home_config_directory()	847
10.74.3.45	set_config_files()	847
10.74.3.46	config_filename() [1/2]	848

10.74.3.47	user_filename() [1/2]	848
10.74.3.48	config_filename_alt() [1/2]	848
10.74.3.49	user_filename_alt() [1/2]	848
10.74.3.50	application_name()	848
10.74.3.51	app_client_name()	848
10.74.3.52	tempo_track_number() [1/2]	848
10.74.3.53	recent_file()	848
10.74.3.54	recent_file_count()	849
10.74.3.55	verbose_option() [2/2]	849
10.74.3.56	auto_option_save() [2/2]	849
10.74.3.57	legacy_format() [2/2]	849
10.74.3.58	ash_support() [2/2]	849
10.74.3.59	allow_mod4_mode() [2/2]	850
10.74.3.60	allow_snap_split() [2/2]	850
10.74.3.61	allow_click_edit() [2/2]	850
10.74.3.62	show_midi() [2/2]	850
10.74.3.63	priority() [2/2]	850
10.74.3.64	stats() [2/2]	850
10.74.3.65	pass_sysex() [2/2]	850
10.74.3.66	with_jack_midi() [2/2]	851
10.74.3.67	filter_by_channel() [2/2]	851
10.74.3.68	manual_alsa_ports() [2/2]	851
10.74.3.69	reveal_alsa_ports() [2/2]	851
10.74.3.70	print_keys() [2/2]	851
10.74.3.71	device_ignore() [2/2]	851
10.74.3.72	tempo_track_number() [2/2]	851
10.74.3.73	device_ignore_num() [2/2]	851
10.74.3.74	interaction_method() [2/2]	852
10.74.3.75	mute_group_saving() [2/2]	852
10.74.3.76	jack_session_uuid() [2/2]	852

10.74.3.77	config_directory() [2/2]	852
10.74.3.78	config_filename() [2/2]	853
10.74.3.79	user_filename() [2/2]	853
10.74.3.80	config_filename_alt() [2/2]	853
10.74.3.81	user_filename_alt() [2/2]	853
10.74.4	Friends And Related Function Documentation	854
10.74.4.1	optionsfile	854
10.74.4.2	options	854
10.74.4.3	mainwnd	854
10.74.4.4	rtmidi_info	854
10.74.4.5	parse_command_line_options	854
10.74.4.6	help_check	855
10.74.5	Field Documentation	855
10.74.5.1	m_comments_block	855
10.74.5.2	m_verbose_option	855
10.74.5.3	m_auto_option_save	856
10.74.5.4	m_legacy_format	856
10.74.5.5	m_lash_support	856
10.74.5.6	m_allow_mod4_mode	856
10.74.5.7	m_allow_snap_split	856
10.74.5.8	m_allow_click_edit	856
10.74.5.9	m_show_midi	856
10.74.5.10	m_priority	856
10.74.5.11	m_stats	857
10.74.5.12	m_pass_sysex	857
10.74.5.13	m_with_jack_transport	857
10.74.5.14	m_with_jack_master	857
10.74.5.15	m_with_jack_master_cond	857
10.74.5.16	m_with_jack_midi	857
10.74.5.17	m_filter_by_channel	857

10.74.5.18m_manual_alsa_ports	857
10.74.5.19m_reveal_alsa_ports	858
10.74.5.20m_print_keys	858
10.74.5.21m_device_ignore	858
10.74.5.22m_device_ignore_num	858
10.74.5.23m_interaction_method	858
10.74.5.24m_mute_group_saving	858
10.74.5.25m_filename	858
10.74.5.26m_jack_session_uuid	858
10.74.5.27m_last_used_dir	859
10.74.5.28m_config_directory	859
10.74.5.29m_config_filename	859
10.74.5.30m_user_filename	859
10.74.5.31m_config_filename_alt	859
10.74.5.32m_user_filename_alt	859
10.74.5.33m_application_name	859
10.74.5.34m_app_client_name	860
10.74.5.35m_tempo_track_number	860
10.74.5.36m_recent_files	860
10.75seq64::rect Class Reference	860
10.75.1 Detailed Description	861
10.75.2 Constructor & Destructor Documentation	862
10.75.2.1 rect() [1/2]	862
10.75.2.2 rect() [2/2]	862
10.75.3 Member Function Documentation	862
10.75.3.1 get()	862
10.75.3.2 set()	863
10.75.3.3 clear()	863
10.75.3.4 xy_to_rect() [1/2]	863
10.75.3.5 xy_to_rect_get()	863

10.75.3.6 xy_to_rect() [2/2]	864
10.75.3.7 x() [1/2]	864
10.75.3.8 x() [2/2]	864
10.75.3.9 x_incr()	864
10.75.3.10y() [1/2]	865
10.75.3.11y() [2/2]	865
10.75.3.12y_incr()	865
10.75.3.13width() [1/2]	865
10.75.3.14width() [2/2]	865
10.75.3.15width_incr()	865
10.75.3.16height() [1/2]	865
10.75.3.17height() [2/2]	866
10.75.3.18height_incr()	866
10.75.3.19xy_incr()	866
10.75.3.20calculated_width()	866
10.75.3.21calculated_height()	866
10.75.4 Field Documentation	866
10.75.4.1 m_x	866
10.75.4.2 m_y	867
10.75.4.3 m_width	867
10.75.4.4 m_height	867
10.76seq64::rterror Class Reference	867
10.76.1 Detailed Description	868
10.76.2 Member Enumeration Documentation	868
10.76.2.1 Type	868
10.76.3 Constructor & Destructor Documentation	868
10.76.3.1 rterror()	868
10.76.3.2 ~rterror()	869
10.76.4 Member Function Documentation	869
10.76.4.1 print_message()	869

10.76.4.2	getType()	869
10.76.4.3	get_message()	869
10.76.4.4	what()	869
10.76.5	Field Documentation	869
10.76.5.1	m_message	869
10.76.5.2	m_type	870
10.77	seq64::rtmidi Class Reference	870
10.77.1	Detailed Description	872
10.77.2	Constructor & Destructor Documentation	872
10.77.2.1	rtmidi()	872
10.77.2.2	~rtmidi()	872
10.77.3	Member Function Documentation	872
10.77.3.1	api_connect()	872
10.77.3.2	api_play()	872
10.77.3.3	api_continue_from()	873
10.77.3.4	api_start()	873
10.77.3.5	api_stop()	873
10.77.3.6	api_clock()	873
10.77.3.7	api_set_ppqn()	873
10.77.3.8	api_set_beats_per_minute()	873
10.77.3.9	api_init_out()	874
10.77.3.10	api_init_out_sub()	874
10.77.3.11	api_init_in()	874
10.77.3.12	api_init_in_sub()	874
10.77.3.13	api_deinit_in()	874
10.77.3.14	api_get_midi_event()	874
10.77.3.15	api_poll_for_midi()	875
10.77.3.16	api_sysex()	875
10.77.3.17	api_flush()	875
10.77.3.18	s_port_open()	875

10.77.3.19	get_bus_id()	875
10.77.3.20	get_bus_name()	876
10.77.3.21	get_port_id()	876
10.77.3.22	get_port_name()	876
10.77.3.23	get_port_count()	876
10.77.3.24	full_port_count()	876
10.77.3.25	get_api() [1/2]	877
10.77.3.26	get_api() [2/2]	877
10.77.3.27	set_api()	877
10.77.3.28	delete_api()	877
10.77.4	Friends And Related Function Documentation	877
10.77.4.1	midibus	877
10.77.5	Field Documentation	877
10.77.5.1	m_midi_info	877
10.77.5.2	m_midi_api	878
10.78	seq64::rtmidi_in Class Reference	878
10.78.1	Detailed Description	880
10.78.2	Constructor & Destructor Documentation	880
10.78.2.1	rtmidi_in()	880
10.78.2.2	~rtmidi_in()	880
10.78.3	Member Function Documentation	880
10.78.3.1	user_callback()	880
10.78.3.2	cancel_callback()	881
10.78.3.3	openmidi_api()	881
10.79	seq64::rtmidi_in_data Class Reference	881
10.79.1	Detailed Description	882
10.79.2	Constructor & Destructor Documentation	882
10.79.2.1	rtmidi_in_data()	882
10.79.3	Member Function Documentation	882
10.79.3.1	queue() [1/2]	883

10.79.3.2 queue() [2/2]	883
10.79.3.3 message() [1/2]	883
10.79.3.4 message() [2/2]	883
10.79.3.5 ignore_flags() [1/2]	883
10.79.3.6 test_ignore_flags()	883
10.79.3.7 ignore_flags() [2/2]	883
10.79.3.8 do_input() [1/2]	884
10.79.3.9 do_input() [2/2]	884
10.79.3.10first_message() [1/2]	884
10.79.3.11first_message() [2/2]	884
10.79.3.12continue_sysex() [1/2]	884
10.79.3.13continue_sysex() [2/2]	884
10.79.3.14using_callback() [1/2]	884
10.79.3.15using_callback() [2/2]	885
10.79.3.16api_data() [1/3]	885
10.79.3.17api_data() [2/3]	885
10.79.3.18api_data() [3/3]	885
10.79.3.19user_data() [1/3]	885
10.79.3.20user_data() [2/3]	885
10.79.3.21user_data() [3/3]	885
10.79.3.22user_callback() [1/2]	886
10.79.3.23user_callback() [2/2]	886
10.79.4 Field Documentation	886
10.79.4.1 m_queue	886
10.79.4.2 m_message	886
10.79.4.3 m_ignore_flags	886
10.79.4.4 m_do_input	886
10.79.4.5 m_first_message	886
10.79.4.6 m_api_data	887
10.79.4.7 m_using_callback	887

10.79.4.8 <code>m_user_callback</code>	887
10.79.4.9 <code>m_user_data</code>	887
10.79.4.10 <code>m_continue_sysex</code>	887
10.80 <code>seq64::rtmidi_info</code> Class Reference	887
10.80.1 Detailed Description	889
10.80.2 Constructor & Destructor Documentation	889
10.80.2.1 <code>rtmidi_info()</code>	889
10.80.2.2 <code>~rtmidi_info()</code>	890
10.80.3 Member Function Documentation	890
10.80.3.1 <code>get_version()</code>	890
10.80.3.2 <code>get_compiled_api()</code>	890
10.80.3.3 <code>midi_mode()</code> [1/2]	890
10.80.3.4 <code>midi_mode()</code> [2/2]	890
10.80.3.5 <code>clear()</code>	890
10.80.3.6 <code>add_input()</code>	890
10.80.3.7 <code>add_output()</code>	891
10.80.3.8 <code>add_bus()</code>	891
10.80.3.9 <code>get_bus_id()</code>	891
10.80.3.10 <code>get_bus_name()</code>	891
10.80.3.11 <code>get_port_count()</code>	891
10.80.3.12 <code>full_port_count()</code>	892
10.80.3.13 <code>get_port_id()</code>	892
10.80.3.14 <code>get_port_name()</code>	892
10.80.3.15 <code>get_input()</code>	892
10.80.3.16 <code>get_virtual()</code>	892
10.80.3.17 <code>get_system()</code>	892
10.80.3.18 <code>get_all_port_info()</code>	892
10.80.3.19 <code>queue_number()</code>	893
10.80.3.20 <code>app_name()</code>	893
10.80.3.21 <code>global_queue()</code>	893

10.80.3.22	ppqn()	893
10.80.3.23	api_set_ppqn()	893
10.80.3.24	bpm()	893
10.80.3.25	api_set_beats_per_minute()	893
10.80.3.26	api_port_start()	894
10.80.3.27	api_get_midi_event()	894
10.80.3.28	api_flush()	894
10.80.3.29	api_poll_for_midi()	894
10.80.3.30	port_list()	894
10.80.3.31	selected_api() [1/2]	894
10.80.3.32	get_api_info() [1/2]	894
10.80.3.33	get_api_info() [2/2]	895
10.80.3.34	api_connect()	895
10.80.3.35	selected_api() [2/2]	895
10.80.3.36	set_api_info()	895
10.80.3.37	delete_api()	895
10.80.3.38	openmidi_api()	895
10.80.4	Friends And Related Function Documentation	895
10.80.4.1	mastermidibus	896
10.80.4.2	midibus	896
10.80.4.3	rtmidi_in	896
10.80.4.4	rtmidi_out	896
10.80.5	Field Documentation	896
10.80.5.1	m_info_api	896
10.80.5.2	sm_selected_api	896
10.81	seq64::rtmidi_out Class Reference	897
10.81.1	Detailed Description	898
10.81.2	Constructor & Destructor Documentation	898
10.81.2.1	rtmidi_out()	898
10.81.2.2	~rtmidi_out()	898

10.81.3 Member Function Documentation	898
10.81.3.1 openmidi_api()	898
10.82seq64::Seq24PerfInput Class Reference	899
10.82.1 Constructor & Destructor Documentation	900
10.82.1.1 Seq24PerfInput()	900
10.82.1.2 ~Seq24PerfInput()	901
10.82.2 Member Function Documentation	901
10.82.2.1 activate_adding()	901
10.82.2.2 handle_motion_key()	901
10.82.2.3 check_handles()	902
10.82.2.4 on_button_press_event()	902
10.82.2.5 on_button_release_event()	903
10.82.2.6 on_motion_notify_event()	904
10.82.3 Field Documentation	904
10.82.3.1 m_effective_tick	904
10.83seq64::Seq24SeqEventInput Class Reference	905
10.83.1 Constructor & Destructor Documentation	906
10.83.1.1 Seq24SeqEventInput()	906
10.83.2 Member Function Documentation	906
10.83.2.1 on_button_press_event()	906
10.83.2.2 on_button_release_event()	907
10.83.2.3 on_motion_notify_event()	907
10.84seq64::seqdata Class Reference	907
10.84.1 Constructor & Destructor Documentation	910
10.84.1.1 seqdata()	910
10.84.1.2 ~seqdata()	911
10.84.2 Member Function Documentation	911
10.84.2.1 reset()	911
10.84.2.2 redraw()	911
10.84.2.3 set_zoom()	911

10.84.2.4 set_data_type()	912
10.84.2.5 idle_redraw()	912
10.84.2.6 update_sizes()	912
10.84.2.7 update_pixmap()	912
10.84.2.8 draw_line_on_window()	912
10.84.2.9 xy_to_rect()	913
10.84.2.10 draw_events_on()	914
10.84.2.11 change_horz()	914
10.84.2.12 convert_x()	915
10.84.2.13 render_number()	915
10.84.2.14 render_digits()	915
10.84.2.15 draw_events_on_pixmap()	915
10.84.2.16 draw_pixmap_on_window()	916
10.84.2.17 on_realize()	916
10.84.2.18 on_expose_event()	916
10.84.2.19 on_button_press_event()	916
10.84.2.20 on_button_release_event()	917
10.84.2.21 on_motion_notify_event()	917
10.84.2.22 on_leave_notify_event()	917
10.84.2.23 on_scroll_event()	917
10.84.2.24 on_size_allocate()	918
10.84.3 Friends And Related Function Documentation	918
10.84.3.1 lfownd	918
10.84.3.2 sequevent	918
10.84.3.3 seqroll	918
10.84.4 Field Documentation	918
10.84.4.1 m_seq	919
10.84.4.2 m_zoom	919
10.84.4.3 m_scroll_offset_ticks	919
10.84.4.4 m_scroll_offset_x	919

10.84.4.5 m_number_w	919
10.84.4.6 m_number_h	919
10.84.4.7 m_number_offset_y	919
10.84.4.8 m_status	919
10.84.4.9 m_cc	920
10.84.4.10m_old	920
10.84.4.11m_drag_handle	920
10.84.4.12m_dragging	920
10.85seq64::seqedit Class Reference	920
10.85.1 Detailed Description	928
10.85.2 Constructor & Destructor Documentation	928
10.85.2.1 seqedit()	929
10.85.2.2 ~seqedit()	929
10.85.3 Member Function Documentation	929
10.85.3.1 set_zoom()	929
10.85.3.2 set_snap()	930
10.85.3.3 set_note_length()	930
10.85.3.4 set_beats_per_bar()	930
10.85.3.5 set_beats_per_bar_manual()	931
10.85.3.6 set_beat_width()	931
10.85.3.7 set_transpose_image()	931
10.85.3.8 set_mousemode_image()	932
10.85.3.9 set_rec_vol()	932
10.85.3.10set_rec_type()	932
10.85.3.11horizontal_adjust()	932
10.85.3.12vertical_adjust()	933
10.85.3.13horizontal_set()	933
10.85.3.14vertical_set()	933
10.85.3.15get_measures()	933
10.85.3.16set_measures()	934

10.85.3.17set_measures_manual()	934
10.85.3.18apply_length()	934
10.85.3.19set_midi_channel()	934
10.85.3.20set_midi_bus()	935
10.85.3.21set_scale()	935
10.85.3.22set_chord()	935
10.85.3.23set_key()	936
10.85.3.24set_background_sequence()	936
10.85.3.25transpose_change_callback()	936
10.85.3.26name_change_callback()	936
10.85.3.27follow_change_callback()	936
10.85.3.28play_change_callback()	936
10.85.3.29record_change_callback()	937
10.85.3.30q_rec_change_callback()	937
10.85.3.31thru_change_callback()	937
10.85.3.32undo_callback()	937
10.85.3.33redo_callback()	937
10.85.3.34update_all_windows()	938
10.85.3.35fill_top_bar()	938
10.85.3.36create_menus()	938
10.85.3.37set_data_type()	938
10.85.3.38set_event_entry()	939
10.85.3.39popup_menu()	939
10.85.3.40popup_event_menu()	939
10.85.3.41repopulate_event_menu()	939
10.85.3.42popup_mini_event_menu()	940
10.85.3.43repopulate_mini_event_menu()	940
10.85.3.44popup_record_menu()	940
10.85.3.45popup_midibus_menu()	940
10.85.3.46popup_sequence_menu()	941

10.85.3.47	popup_tool_menu()	941
10.85.3.48	popup_midich_menu()	941
10.85.3.49	repopulate_midich_menu()	941
10.85.3.50	create_menu_image()	941
10.85.3.51	timeout()	942
10.85.3.52	do_action()	942
10.85.3.53	mouse_action()	942
10.85.3.54	start_playing()	942
10.85.3.55	stop_playing()	942
10.85.3.56	change_focus()	942
10.85.3.57	handle_close()	943
10.85.3.58	on_realize()	943
10.85.3.59	on_set_focus()	943
10.85.3.60	on_focus_in_event()	943
10.85.3.61	on_focus_out_event()	943
10.85.3.62	on_delete_event()	944
10.85.3.63	on_scroll_event()	944
10.85.3.64	on_key_press_event()	944
10.85.4	Field Documentation	945
10.85.4.1	seqmenu	945
10.85.4.2	m_initial_snap	945
10.85.4.3	m_initial_note_length	946
10.85.4.4	m_initial_chord	946
10.85.4.5	m_initial_zoom	946
10.85.4.6	m_zoom	946
10.85.4.7	m_snap	946
10.85.4.8	m_note_length	946
10.85.4.9	m_scale	946
10.85.4.10	m_chord	946
10.85.4.11	m_key	947

10.85.4.12m_bgsequence	947
10.85.4.13m_measures	947
10.85.4.14m_ppqn	947
10.85.4.15m_pp_whole	947
10.85.4.16m_pp_eighth	947
10.85.4.17m_pp_sixteenth	947
10.85.4.18m_seq	947
10.85.4.19m_menubar	948
10.85.4.20m_menu_tools	948
10.85.4.21m_menu_zoom	948
10.85.4.22m_menu_snap	948
10.85.4.23m_menu_note_length	948
10.85.4.24m_menu_length	948
10.85.4.25m_toggle_transpose	948
10.85.4.26m_image_transpose	949
10.85.4.27m_menu_midich	949
10.85.4.28m_menu_midibus	949
10.85.4.29m_menu_data	949
10.85.4.30m_menu_minidata	949
10.85.4.31m_menu_key	949
10.85.4.32m_menu_scale	949
10.85.4.33m_menu_chords	949
10.85.4.34m_menu_sequences	950
10.85.4.35m_menu_bpm	950
10.85.4.36m_menu_bw	950
10.85.4.37m_menu_rec_vol	950
10.85.4.38m_menu_rec_type	950
10.85.4.39m_vadjust	950
10.85.4.40m_hadjust	950
10.85.4.41m_vscroll_new	950

10.85.4.42m_hscroll_new	951
10.85.4.43m_seqkeys_wid	951
10.85.4.44m_seqtime_wid	951
10.85.4.45m_seqdata_wid	951
10.85.4.46m_segevent_wid	951
10.85.4.47m_seqroll_wid	951
10.85.4.48m_button_lfo	951
10.85.4.49m_lfo_wnd	951
10.85.4.50m_table	952
10.85.4.51m_vbox	952
10.85.4.52m_hbox	952
10.85.4.53m_hbox2	952
10.85.4.54m_button_undo	952
10.85.4.55m_button_redo	952
10.85.4.56m_button_quantize	952
10.85.4.57m_button_tools	952
10.85.4.58m_button_sequence	953
10.85.4.59m_entry_sequence	953
10.85.4.60m_button_bus	953
10.85.4.61m_entry_bus	953
10.85.4.62m_button_channel	953
10.85.4.63m_entry_channel	953
10.85.4.64m_button_snap	953
10.85.4.65m_entry_snap	953
10.85.4.66m_button_note_length	954
10.85.4.67m_entry_note_length	954
10.85.4.68m_button_zoom	954
10.85.4.69m_entry_zoom	954
10.85.4.70m_button_length	954
10.85.4.71m_entry_length	954

10.85.4.72m_button_key	954
10.85.4.73m_entry_key	954
10.85.4.74m_button_scale	955
10.85.4.75m_entry_scale	955
10.85.4.76m_button_chord	955
10.85.4.77m_entry_chord	955
10.85.4.78m_tooltips	955
10.85.4.79m_button_data	955
10.85.4.80m_button_minidata	955
10.85.4.81m_entry_data	955
10.85.4.82m_button_bpm	956
10.85.4.83m_entry_bpm	956
10.85.4.84m_button_bw	956
10.85.4.85m_entry_bw	956
10.85.4.86m_button_rec_vol	956
10.85.4.87m_button_rec_type	956
10.85.4.88m_toggle_follow	956
10.85.4.89m_toggle_play	956
10.85.4.90m_toggle_record	957
10.85.4.91m_toggle_q_rec	957
10.85.4.92m_toggle_thru	957
10.85.4.93m_entry_seqnumber	957
10.85.4.94m_entry_name	957
10.85.4.95m_image_mousemode	957
10.85.4.96m_editing_status	957
10.85.4.97m_editing_cc	957
10.85.4.98m_first_event	958
10.85.4.99m_first_event_name	958
10.85.4.100m_have_focus	958
10.86seq64::seqevent Class Reference	958

10.86.1 Constructor & Destructor Documentation	962
10.86.1.1 sequevent()	962
10.86.1.2 ~sequevent()	962
10.86.2 Member Function Documentation	962
10.86.2.1 reset()	963
10.86.2.2 redraw()	963
10.86.2.3 set_zoom()	963
10.86.2.4 set_adding()	963
10.86.2.5 set_snap()	963
10.86.2.6 set_data_type()	964
10.86.2.7 update_sizes()	964
10.86.2.8 draw_background()	964
10.86.2.9 draw_events_on_pixmap()	964
10.86.2.10 draw_pixmap_on_window()	964
10.86.2.11 draw_selection_on_window()	965
10.86.2.12 update_pixmap()	965
10.86.2.13 force_draw()	965
10.86.2.14 idle_redraw()	965
10.86.2.15 x_to_w()	965
10.86.2.16 drop_event()	966
10.86.2.17 draw_events_on()	966
10.86.2.18 start_paste()	966
10.86.2.19 change_horz()	966
10.86.2.20 convert_x()	967
10.86.2.21 convert_t()	967
10.86.2.22 snap_y()	967
10.86.2.23 snap_x()	967
10.86.2.24 on_realize()	968
10.86.2.25 on_expose_event()	968
10.86.2.26 on_button_press_event()	968

10.86.2.27	on_button_release_event()	969
10.86.2.28	on_motion_notify_event()	969
10.86.2.29	on_focus_in_event()	970
10.86.2.30	on_focus_out_event()	970
10.86.2.31	on_key_press_event()	970
10.86.2.32	on_size_allocate()	971
10.86.3	Field Documentation	971
10.86.3.1	m_seq	971
10.86.3.2	m_zoom	971
10.86.3.3	m_snap	971
10.86.3.4	m_old	971
10.86.3.5	m_selected	971
10.86.3.6	m_scroll_offset_ticks	972
10.86.3.7	m_scroll_offset_x	972
10.86.3.8	m_seqdata_wid	972
10.86.3.9	m_adding	972
10.86.3.10	m_selecting	972
10.86.3.11	m_moving_init	972
10.86.3.12	m_moving	972
10.86.3.13	m_growing	972
10.86.3.14	m_painting	973
10.86.3.15	m_paste	973
10.86.3.16	m_move_snap_offset_x	973
10.86.3.17	m_status	973
10.86.3.18	m_cc	973
10.87	seq64::seqkeys Class Reference	973
10.87.1	Detailed Description	976
10.87.2	Constructor & Destructor Documentation	976
10.87.2.1	seqkeys()	976
10.87.2.2	~seqkeys()	977

10.87.3 Member Function Documentation	977
10.87.3.1 set_scale()	977
10.87.3.2 set_key()	977
10.87.3.3 set_hint_key()	977
10.87.3.4 set_hint_state()	978
10.87.3.5 force_draw()	978
10.87.3.6 set_listen_button_press()	978
10.87.3.7 set_listen_button_release()	978
10.87.3.8 set_listen_motion_notify()	979
10.87.3.9 draw_area()	979
10.87.3.10 update_pixmap()	979
10.87.3.11 convert_y()	979
10.87.3.12 draw_key()	979
10.87.3.13 change_vert()	980
10.87.3.14 update_sizes()	980
10.87.3.15 reset()	980
10.87.3.16 s_black_key()	980
10.87.3.17 on_realize()	981
10.87.3.18 on_expose_event()	981
10.87.3.19 on_button_press_event()	981
10.87.3.20 on_button_release_event()	981
10.87.3.21 on_motion_notify_event()	982
10.87.3.22 on_enter_notify_event()	982
10.87.3.23 on_leave_notify_event()	982
10.87.3.24 on_scroll_event()	982
10.87.3.25 on_size_allocate()	983
10.87.4 Friends And Related Function Documentation	983
10.87.4.1 seqroll	983
10.87.4.2 FruitySeqRollInput	983
10.87.5 Field Documentation	983

10.87.5.1 m_seq	983
10.87.5.2 m_scroll_offset_key	984
10.87.5.3 m_scroll_offset_y	984
10.87.5.4 m_hint_state	984
10.87.5.5 m_hint_key	984
10.87.5.6 m_keying	984
10.87.5.7 m_keying_note	984
10.87.5.8 m_scale	984
10.87.5.9 m_key	984
10.87.5.10 m_show_octave_letters	985
10.88 seq64::seqmenu Class Reference	985
10.88.1 Detailed Description	989
10.88.2 Member Typedef Documentation	989
10.88.2.1 SeqeditMap	989
10.88.2.2 SeqeditPair	989
10.88.2.3 iterator	989
10.88.2.4 const_iterator	989
10.88.3 Constructor & Destructor Documentation	989
10.88.3.1 seqmenu()	989
10.88.3.2 ~seqmenu()	990
10.88.4 Member Function Documentation	990
10.88.4.1 current_seq() [1/2]	990
10.88.4.2 is_modified() [1/2]	990
10.88.4.3 current_seq() [2/2]	990
10.88.4.4 set_edit_sequence()	990
10.88.4.5 unset_edit_sequence()	991
10.88.4.6 is_edit_sequence()	991
10.88.4.7 is_modified() [2/2]	991
10.88.4.8 get_current_sequence()	991
10.88.4.9 get_sequence()	991

10.88.4.10	<code>s_current_seq_active()</code>	991
10.88.4.11	<code>is_current_seq_in_edit()</code>	991
10.88.4.12	<code>new_current_sequence()</code>	992
10.88.4.13	<code>new_sequence()</code>	992
10.88.4.14	<code>delete_current_sequence()</code>	992
10.88.4.15	<code>toggle_current_sequence()</code>	992
10.88.4.16	<code>popup_menu()</code>	992
10.88.4.17	<code>seq_edit()</code>	992
10.88.4.18	<code>seq_event_edit()</code>	993
10.88.4.19	<code>create_seqedit()</code>	993
10.88.4.20	<code>remove_seqedit()</code>	993
10.88.4.21	<code>seq_set_and_edit()</code>	993
10.88.4.22	<code>seq_set_and_eventedit()</code>	994
10.88.4.23	<code>redraw()</code>	994
10.88.4.24	<code>seq_new()</code>	994
10.88.4.25	<code>seq_copy()</code>	994
10.88.4.26	<code>seq_cut()</code>	995
10.88.4.27	<code>seq_paste()</code>	995
10.88.4.28	<code>seq_clear_perf()</code>	995
10.88.4.29	<code>set_bus_and_midi_channel()</code>	995
10.88.4.30	<code>set_transposable()</code>	995
10.88.4.31	<code>mute_all_tracks()</code>	996
10.88.4.32	<code>unmute_all_tracks()</code>	996
10.88.4.33	<code>toggle_all_tracks()</code>	996
10.88.4.34	<code>toggle_playing_tracks()</code>	996
10.88.4.35	<code>on_realize()</code>	996
10.88.5	Friends And Related Function Documentation	996
10.88.5.1	<code>mainwnd</code>	996
10.88.5.2	<code>seqedit</code>	997
10.88.6	Field Documentation	997

10.88.6.1 sm_segedit_list	997
10.88.6.2 sm_clipboard	997
10.88.6.3 sm_clipboard_empty	997
10.88.6.4 m_menu	997
10.88.6.5 m_mainperf	997
10.88.6.6 m_segedit	998
10.88.6.7 m_eventedit	998
10.88.6.8 m_current_seq	998
10.88.6.9 m_modified	998
10.89seq64::seqroll Class Reference	998
10.89.1 Constructor & Destructor Documentation	1005
10.89.1.1 seqroll()	1005
10.89.1.2 ~seqroll()	1005
10.89.2 Member Function Documentation	1005
10.89.2.1 set_snap()	1005
10.89.2.2 set_zoom()	1006
10.89.2.3 set_note_length()	1006
10.89.2.4 note_off_length()	1006
10.89.2.5 add_note()	1006
10.89.2.6 set_key()	1007
10.89.2.7 set_scale()	1007
10.89.2.8 set_chord()	1007
10.89.2.9 set_data_type()	1007
10.89.2.10set_background_sequence()	1008
10.89.2.11update_pixmap()	1009
10.89.2.12update_sizes()	1009
10.89.2.13update_background()	1009
10.89.2.14draw_background_on_pixmap()	1009
10.89.2.15draw_events_on_pixmap()	1009
10.89.2.16draw_selection_on_window()	1010

10.89.2.17	<code>draw_progress_on_window()</code>	1010
10.89.2.18	<code>reset()</code>	1010
10.89.2.19	<code>update_and_draw()</code>	1010
10.89.2.20	<code>redraw()</code>	1010
10.89.2.21	<code>redraw_events()</code>	1011
10.89.2.22	<code>start_paste()</code>	1011
10.89.2.23	<code>complete_paste()</code> [1/2]	1011
10.89.2.24	<code>complete_paste()</code> [2/2]	1011
10.89.2.25	<code>follow_progress()</code>	1011
10.89.2.26	<code>set_expanded_recording()</code>	1011
10.89.2.27	<code>get_expanded_record()</code>	1012
10.89.2.28	<code>set_progress_follow()</code>	1012
10.89.2.29	<code>get_progress_follow()</code>	1012
10.89.2.30	<code>force_draw()</code>	1012
10.89.2.31	<code>horizontal_adjust()</code>	1012
10.89.2.32	<code>vertical_adjust()</code>	1012
10.89.2.33	<code>snap_y()</code>	1013
10.89.2.34	<code>snap_x()</code>	1013
10.89.2.35	<code>convert_xy()</code>	1013
10.89.2.36	<code>convert_drop_xy()</code>	1014
10.89.2.37	<code>convert_tn()</code>	1014
10.89.2.38	<code>xy_to_rect()</code>	1014
10.89.2.39	<code>convert_tn_box_to_rect()</code> [1/2]	1014
10.89.2.40	<code>convert_tn_box_to_rect()</code> [2/2]	1015
10.89.2.41	<code>convert_sel_box_to_rect()</code>	1015
10.89.2.42	<code>get_selected_box()</code>	1016
10.89.2.43	<code>draw_events_on()</code>	1016
10.89.2.44	<code>idle_redraw()</code>	1017
10.89.2.45	<code>idle_progress()</code>	1017
10.89.2.46	<code>change_horz()</code>	1017

10.89.2.47	change_vert()	1017
10.89.2.48	move_selection_box()	1017
10.89.2.49	move_selected_notes()	1018
10.89.2.50	grow_selected_notes()	1018
10.89.2.51	set_adding()	1018
10.89.2.52	update_mouse_pointer()	1019
10.89.2.53	button_press_initial()	1019
10.89.2.54	align_selection()	1019
10.89.2.55	button_press()	1019
10.89.2.56	button_release()	1020
10.89.2.57	motion_notify()	1020
10.89.2.58	clear_flags()	1021
10.89.2.59	set_scroll_x()	1021
10.89.2.60	set_scroll_y()	1021
10.89.2.61	scroll_offset_x()	1021
10.89.2.62	scroll_offset_y()	1021
10.89.2.63	set_current_offset_x_y()	1021
10.89.2.64	adding()	1022
10.89.2.65	selecting()	1022
10.89.2.66	growing()	1022
10.89.2.67	normal_action()	1022
10.89.2.68	select_action()	1022
10.89.2.69	drop_action()	1023
10.89.2.70	moving()	1023
10.89.2.71	on_realize()	1023
10.89.2.72	on_expose_event()	1023
10.89.2.73	on_button_press_event()	1023
10.89.2.74	on_button_release_event()	1024
10.89.2.75	on_motion_notify_event()	1024
10.89.2.76	on_focus_in_event()	1025

10.89.2.77	on_focus_out_event()	1025
10.89.2.78	on_key_press_event()	1025
10.89.2.79	on_scroll_event()	1026
10.89.2.80	on_size_allocate()	1026
10.89.2.81	on_leave_notify_event()	1026
10.89.2.82	on_enter_notify_event()	1027
10.89.3	Field Documentation	1027
10.89.3.1	m_horizontal_adjust	1027
10.89.3.2	m_vertical_adjust	1027
10.89.3.3	m_old	1027
10.89.3.4	m_selected	1027
10.89.3.5	m_seq	1027
10.89.3.6	m_seqkeys_wid	1028
10.89.3.7	m_pos	1028
10.89.3.8	m_zoom	1028
10.89.3.9	m_snap	1028
10.89.3.10	m_note_length	1028
10.89.3.11	m_scale	1028
10.89.3.12	m_chord	1028
10.89.3.13	m_key	1028
10.89.3.14	m_adding	1029
10.89.3.15	m_selecting	1029
10.89.3.16	m_moving	1029
10.89.3.17	m_moving_init	1029
10.89.3.18	m_growing	1029
10.89.3.19	m_painting	1029
10.89.3.20	m_paste	1029
10.89.3.21	m_is_drag_pasting	1029
10.89.3.22	m_is_drag_pasting_start	1030
10.89.3.23	m_justselected_one	1030

10.89.3.24m_move_delta_x	1030
10.89.3.25m_move_delta_y	1030
10.89.3.26m_move_snap_offset_x	1030
10.89.3.27m_progress_x	1030
10.89.3.28m_scroll_offset_ticks	1030
10.89.3.29m_scroll_offset_key	1030
10.89.3.30m_scroll_offset_x	1031
10.89.3.31m_scroll_offset_y	1031
10.89.3.32m_scroll_page	1031
10.89.3.33m_progress_follow	1031
10.89.3.34m_transport_follow	1031
10.89.3.35m_trans_button_press	1031
10.89.3.36m_background_sequence	1031
10.89.3.37m_drawing_background_seq	1031
10.89.3.38m_expanded_recording	1032
10.89.3.39m_status	1032
10.89.3.40m_cc	1032
10.90seq64::seqtime Class Reference	1032
10.90.1 Constructor & Destructor Documentation	1034
10.90.1.1 seqtime()	1035
10.90.1.2 ~seqtime()	1035
10.90.2 Member Function Documentation	1035
10.90.2.1 reset()	1035
10.90.2.2 redraw()	1035
10.90.2.3 set_zoom()	1035
10.90.2.4 draw_pixmap_on_window()	1035
10.90.2.5 draw_progress_on_window()	1036
10.90.2.6 update_pixmap()	1036
10.90.2.7 change_horz()	1036
10.90.2.8 update_sizes()	1037

10.90.2.9 <code>idle_progress()</code>	1037
10.90.2.10 <code>on_realize()</code>	1037
10.90.2.11 <code>on_expose_event()</code>	1037
10.90.2.12 <code>on_size_allocate()</code>	1037
10.90.2.13 <code>on_button_press_event()</code>	1037
10.90.2.14 <code>on_button_release_event()</code>	1037
10.90.3 Field Documentation	1038
10.90.3.1 <code>m_seq</code>	1038
10.90.3.2 <code>m_scroll_offset_ticks</code>	1038
10.90.3.3 <code>m_scroll_offset_x</code>	1038
10.90.3.4 <code>m_zoom</code>	1038
10.91 <code>seq64::sequence</code> Class Reference	1038
10.91.1 Detailed Description	1052
10.91.2 Member Typedef Documentation	1052
10.91.2.1 <code>EventStack</code>	1052
10.91.3 Member Enumeration Documentation	1052
10.91.3.1 <code>select_action_t</code>	1052
10.91.4 Constructor & Destructor Documentation	1052
10.91.4.1 <code>sequence()</code>	1052
10.91.4.2 <code>~sequence()</code>	1053
10.91.5 Member Function Documentation	1053
10.91.5.1 <code>operator=()</code>	1053
10.91.5.2 <code>partial_assign()</code>	1053
10.91.5.3 <code>events()</code> [1/2]	1053
10.91.5.4 <code>events()</code> [2/2]	1054
10.91.5.5 <code>any_selected_notes()</code>	1054
10.91.5.6 <code>triggerlist()</code> [1/2]	1054
10.91.5.7 <code>triggerlist()</code> [2/2]	1054
10.91.5.8 <code>trigger_count()</code>	1054
10.91.5.9 <code>selected_trigger_count()</code>	1054

10.91.5.10	set_trigger_paste_tick()	1054
10.91.5.11	get_trigger_paste_tick()	1055
10.91.5.12	seq_number()	1055
10.91.5.13	number() [1/2]	1055
10.91.5.14	number() [2/2]	1055
10.91.5.15	color() [1/2]	1055
10.91.5.16	color() [2/2]	1055
10.91.5.17	edit_mode() [1/2]	1056
10.91.5.18	edit_mode() [2/2]	1056
10.91.5.19	modify()	1056
10.91.5.20	event_count()	1056
10.91.5.21	set_hold_undo()	1056
10.91.5.22	get_hold_undo()	1057
10.91.5.23	set_have_undo()	1057
10.91.5.24	have_undo()	1057
10.91.5.25	set_have_redo()	1057
10.91.5.26	have_redo()	1057
10.91.5.27	push_undo()	1057
10.91.5.28	pop_undo()	1058
10.91.5.29	pop_redo()	1058
10.91.5.30	push_trigger_undo()	1058
10.91.5.31	pop_trigger_undo()	1058
10.91.5.32	pop_trigger_redo()	1058
10.91.5.33	set_name()	1058
10.91.5.34	calculate_measures()	1059
10.91.5.35	get_ppqn()	1059
10.91.5.36	set_beats_per_bar()	1059
10.91.5.37	get_beats_per_bar()	1059
10.91.5.38	set_beat_width()	1059
10.91.5.39	get_beat_width()	1060

10.91.5.40	measures_to_ticks()	1060
10.91.5.41	clocks_per_metronome() [1/2]	1060
10.91.5.42	clocks_per_metronome() [2/2]	1060
10.91.5.43	set_32nds_per_quarter()	1060
10.91.5.44	get_32nds_per_quarter()	1060
10.91.5.45	us_per_quarter_note() [1/2]	1061
10.91.5.46	us_per_quarter_note() [2/2]	1061
10.91.5.47	set_rec_vol()	1061
10.91.5.48	set_song_mute()	1061
10.91.5.49	toggle_song_mute()	1061
10.91.5.50	get_song_mute()	1061
10.91.5.51	apply_song_transpose()	1062
10.91.5.52	set_transposable()	1062
10.91.5.53	get_transposable()	1062
10.91.5.54	title()	1062
10.91.5.55	name()	1062
10.91.5.56	default_name()	1062
10.91.5.57	default_name()	1063
10.91.5.58	set_editing()	1063
10.91.5.59	get_editing()	1063
10.91.5.60	set_raise()	1063
10.91.5.61	get_raise()	1063
10.91.5.62	set_length()	1063
10.91.5.63	set_num_measures()	1064
10.91.5.64	get_num_measures()	1064
10.91.5.65	apply_length() [1/2]	1064
10.91.5.66	extend()	1065
10.91.5.67	apply_length() [2/2]	1065
10.91.5.68	get_length()	1065
10.91.5.69	get_last_tick()	1066

10.91.5.70	set_last_tick()	1066
10.91.5.71	mod_last_tick()	1066
10.91.5.72	set_playing()	1066
10.91.5.73	get_playing()	1066
10.91.5.74	toggle_playing() [1/2]	1066
10.91.5.75	toggle_playing() [2/2]	1067
10.91.5.76	toggle_queued()	1067
10.91.5.77	get_queued()	1067
10.91.5.78	get_queued_tick()	1067
10.91.5.79	check_queued_tick()	1067
10.91.5.80	set_recording()	1067
10.91.5.81	set_quantized_recording()	1068
10.91.5.82	set_input_recording()	1068
10.91.5.83	get_recording()	1068
10.91.5.84	recording_next_measure()	1068
10.91.5.85	get_snap_tick()	1068
10.91.5.86	set_snap_tick()	1069
10.91.5.87	get_quantized_rec()	1069
10.91.5.88	set_thru()	1069
10.91.5.89	set_input_thru()	1069
10.91.5.90	get_thru()	1069
10.91.5.91	off_one_shot()	1069
10.91.5.92	song_recording_start()	1070
10.91.5.93	song_recording_stop()	1070
10.91.5.94	one_shot_tick()	1070
10.91.5.95	song_recording() [1/2]	1070
10.91.5.96	one_shot() [1/2]	1071
10.91.5.97	off_from_snap() [1/2]	1071
10.91.5.98	song_playback_block() [1/2]	1071
10.91.5.99	song_recording_snap() [1/2]	1071

10.91.5.100ong_record_tick() [1/2]	1071
10.91.5.101resume_note_ons()	1071
10.91.5.102oggle_one_shot()	1071
10.91.5.103dirty_main()	1072
10.91.5.104dirty_edit()	1072
10.91.5.105dirty_perf()	1072
10.91.5.106dirty_names()	1072
10.91.5.107set_dirty_mp()	1073
10.91.5.108set_dirty()	1073
10.91.5.109get_midi_channel()	1073
10.91.5.110smf_0()	1073
10.91.5.111set_midi_channel()	1073
10.91.5.112print()	1074
10.91.5.113print_triggers()	1074
10.91.5.114play()	1074
10.91.5.115play_queue()	1074
10.91.5.116add_note()	1075
10.91.5.117add_chord()	1076
10.91.5.118add_event() [1/2]	1076
10.91.5.119add_event() [2/2]	1077
10.91.5.120append_event()	1077
10.91.5.121sort_events()	1078
10.91.5.122add_trigger()	1078
10.91.5.123split_trigger()	1078
10.91.5.124half_split_trigger()	1079
10.91.5.125exact_split_trigger()	1079
10.91.5.126grow_trigger()	1079
10.91.5.127delete_trigger()	1079
10.91.5.128get_trigger_state()	1080
10.91.5.129select_trigger()	1080

10.91.5.130	get_triggers()	1080
10.91.5.131	inselect_trigger()	1081
10.91.5.132	inselect_triggers()	1081
10.91.5.133	intersect_triggers() [1/2]	1081
10.91.5.134	intersect_triggers() [2/2]	1081
10.91.5.135	intersect_notes()	1082
10.91.5.136	intersect_events()	1082
10.91.5.137	delete_selected_triggers()	1083
10.91.5.138	cut_selected_trigger()	1083
10.91.5.139	copy_selected_trigger()	1083
10.91.5.140	paste_trigger()	1083
10.91.5.141	move_triggers() [1/2]	1083
10.91.5.142	selected_trigger_start()	1084
10.91.5.143	selected_trigger_end()	1084
10.91.5.144	get_max_trigger()	1085
10.91.5.145	move_triggers() [2/2]	1085
10.91.5.146	copy_triggers()	1085
10.91.5.147	clear_triggers()	1085
10.91.5.148	get_trigger_offset()	1086
10.91.5.149	get_midi_bus()	1086
10.91.5.150	get_midi_bus()	1086
10.91.5.151	set_master_midi_bus()	1086
10.91.5.152	select_note_events()	1087
10.91.5.153	select_events() [1/3]	1087
10.91.5.154	select_events() [2/3]	1088
10.91.5.155	select_events() [3/3]	1088
10.91.5.156	select_event_handle()	1089
10.91.5.157	select_linked()	1089
10.91.5.158	select_even_or_odd_notes()	1090
10.91.5.159	select_all_notes()	1090

10.91.5.160	get_num_selected_notes()	1090
10.91.5.161	get_num_selected_events()	1090
10.91.5.162	select_all()	1091
10.91.5.163	copy_selected()	1091
10.91.5.164	cut_selected()	1091
10.91.5.165	paste_selected()	1092
10.91.5.166	get_selected_box()	1093
10.91.5.167	get_onsets_selected_box()	1093
10.91.5.168	get_clipboard_box()	1093
10.91.5.169	adjust_timestamp()	1094
10.91.5.170	dim_timestamp()	1094
10.91.5.171	flip_timestamp()	1095
10.91.5.172	move_selected_notes()	1095
10.91.5.173	stream_event()	1096
10.91.5.174	change_event_data_range()	1097
10.91.5.175	change_event_data_lfo()	1098
10.91.5.176	increment_selected()	1098
10.91.5.177	decrement_selected()	1099
10.91.5.178	grow_selected()	1099
10.91.5.179	stretch_selected()	1100
10.91.5.180	mark_selected()	1100
10.91.5.181	remove_selected()	1100
10.91.5.182	remove_marked()	1101
10.91.5.183	repaint_all()	1101
10.91.5.184	unselect()	1101
10.91.5.185	verify_and_link()	1101
10.91.5.186	link_new()	1101
10.91.5.187	link_tempos()	1101
10.91.5.188	zero_markers()	1101
10.91.5.189	play_note_on()	1101

10.91.5.190	lay_note_off()	1102
10.91.5.191	off_playing_notes()	1102
10.91.5.192	stop()	1102
10.91.5.193	pause()	1102
10.91.5.194	ac_draw_marker()	1103
10.91.5.195	reset_draw_marker()	1103
10.91.5.196	reset_draw_trigger_marker()	1103
10.91.5.197	reset_ex_iterator()	1103
10.91.5.198	get_next_note_event()	1103
10.91.5.199	get_minmax_note_events()	1104
10.91.5.200	get_next_event()	1104
10.91.5.201	get_next_event_ex()	1105
10.91.5.202	get_next_trigger()	1105
10.91.5.203	quantize_events()	1106
10.91.5.204	flush_quantize()	1106
10.91.5.205	transpose_notes()	1107
10.91.5.206	shift_notes()	1107
10.91.5.207	multiply_pattern()	1107
10.91.5.208	musical_key() [1/2]	1107
10.91.5.209	musical_key() [2/2]	1108
10.91.5.210	musical_scale() [1/2]	1108
10.91.5.211	musical_scale() [2/2]	1108
10.91.5.212	background_sequence() [1/2]	1108
10.91.5.213	background_sequence() [2/2]	1108
10.91.5.214	show_events()	1108
10.91.5.215	copy_events()	1108
10.91.5.216	note_off_margin()	1109
10.91.5.217	set_unit_measure()	1109
10.91.5.218	get_unit_measure()	1109
10.91.5.219	channel_match()	1109

10.91.5.226	set_overwrite_rec()	1109
10.91.5.227	get_overwrite_rec()	1109
10.91.5.228	set_loop_reset()	1110
10.91.5.229	get_loop_reset()	1110
10.91.5.230	handle_size()	1110
10.91.5.231	event_in_range()	1110
10.91.5.232	set_parent()	1111
10.91.5.233	put_event_on_bus()	1111
10.91.5.234	reset_loop()	1111
10.91.5.235	set_trigger_offset()	1111
10.91.5.236	adjust_trigger_offsets_to_length()	1112
10.91.5.237	adjust_offset()	1112
10.91.5.238	remove() [1/2]	1112
10.91.5.239	remove() [2/2]	1112
10.91.5.240	remove_all()	1113
10.91.5.241	channels_match()	1113
10.91.5.242	one_shot() [2/2]	1113
10.91.5.243	off_from_snap() [2/2]	1113
10.91.5.244	long_playback_block() [2/2]	1114
10.91.5.245	long_recording() [2/2]	1114
10.91.5.246	long_recording_snap() [2/2]	1114
10.91.5.247	long_record_tick() [2/2]	1114
10.91.6	Friends And Related Function Documentation	1114
10.91.6.1	perform	1114
10.91.6.2	triggers	1114
10.91.7	Field Documentation	1114
10.91.7.1	m_events_clipboard	1115
10.91.7.2	m_parent	1115
10.91.7.3	m_events	1115
10.91.7.4	m_triggers	1115

10.91.7.5 m_events_undo_hold	1115
10.91.7.6 m_have_undo	1115
10.91.7.7 m_have_redo	1115
10.91.7.8 m_events_undo	1116
10.91.7.9 m_events_redo	1116
10.91.7.10m_iterator_draw	1116
10.91.7.11m_channel_match	1116
10.91.7.12m_midi_channel	1116
10.91.7.13m_bus	1116
10.91.7.14m_song_mute	1116
10.91.7.15m_transposable	1117
10.91.7.16m_notes_on	1117
10.91.7.17m_master_bus	1117
10.91.7.18m_playing_notes	1117
10.91.7.19m_was_playing	1117
10.91.7.20m_playing	1117
10.91.7.21m_recording	1117
10.91.7.22m_quantized_rec	1118
10.91.7.23m_thru	1118
10.91.7.24m_queued	1118
10.91.7.25m_one_shot	1118
10.91.7.26m_one_shot_tick	1118
10.91.7.27m_off_from_snap	1118
10.91.7.28m_song_playback_block	1118
10.91.7.29m_song_recording	1119
10.91.7.30m_song_recording_snap	1119
10.91.7.31m_song_record_tick	1119
10.91.7.32m_overwrite_recording	1119
10.91.7.33m_loop_reset	1119
10.91.7.34m_unit_measure	1119

10.91.7.35m_dirty_main	1119
10.91.7.36m_dirty_edit	1119
10.91.7.37m_dirty_perf	1120
10.91.7.38m_dirty_names	1120
10.91.7.39m_editing	1120
10.91.7.40m_raise	1120
10.91.7.41m_name	1120
10.91.7.42m_default_name	1120
10.91.7.43m_last_tick	1120
10.91.7.44m_queued_tick	1120
10.91.7.45m_trigger_offset	1121
10.91.7.46m_maxbeats	1121
10.91.7.47m_ppqn	1121
10.91.7.48m_seq_number	1121
10.91.7.49m_seq_color	1121
10.91.7.50m_seq_edit_mode	1121
10.91.7.51m_length	1121
10.91.7.52m_snap_tick	1122
10.91.7.53m_time_beats_per_measure	1122
10.91.7.54m_time_beat_width	1122
10.91.7.55m_clocks_per_metronome	1122
10.91.7.56m_32nds_per_quarter	1122
10.91.7.57m_us_per_quarter_note	1122
10.91.7.58m_rec_vol	1122
10.91.7.59m_note_on_velocity	1123
10.91.7.60m_note_off_velocity	1123
10.91.7.61m_musical_key	1123
10.91.7.62m_musical_scale	1123
10.91.7.63m_background_sequence	1123
10.91.7.64m_mutex	1123

10.91.7.65m_note_off_margin	1123
10.92seq64::trigger Class Reference	1123
10.92.1 Detailed Description	1125
10.92.2 Constructor & Destructor Documentation	1125
10.92.2.1 trigger()	1125
10.92.3 Member Function Documentation	1125
10.92.3.1 operator<()	1125
10.92.3.2 length()	1125
10.92.3.3 tick_start() [1/2]	1125
10.92.3.4 tick_start() [2/2]	1126
10.92.3.5 increment_tick_start()	1126
10.92.3.6 decrement_tick_start()	1126
10.92.3.7 at_trigger_transition()	1126
10.92.3.8 tick_end() [1/2]	1126
10.92.3.9 tick_end() [2/2]	1126
10.92.3.10increment_tick_end()	1127
10.92.3.11decrement_tick_end()	1127
10.92.3.12offset() [1/2]	1127
10.92.3.13offset() [2/2]	1127
10.92.3.14increment_offset()	1127
10.92.3.15decrement_offset()	1127
10.92.3.16selected() [1/2]	1127
10.92.3.17selected() [2/2]	1128
10.92.4 Field Documentation	1128
10.92.4.1 m_tick_start	1128
10.92.4.2 m_tick_end	1128
10.92.4.3 m_offset	1128
10.92.4.4 m_selected	1128
10.93seq64::triggers Class Reference	1128
10.93.1 Member Typedef Documentation	1131

10.93.1.1 List	1131
10.93.1.2 Stack	1131
10.93.2 Member Enumeration Documentation	1132
10.93.2.1 grow_edit_t	1132
10.93.3 Constructor & Destructor Documentation	1132
10.93.3.1 triggers()	1132
10.93.3.2 ~triggers()	1132
10.93.4 Member Function Documentation	1132
10.93.4.1 operator=()	1132
10.93.4.2 set_ppqn()	1133
10.93.4.3 set_length()	1133
10.93.4.4 triggerlist() [1/2]	1133
10.93.4.5 triggerlist() [2/2]	1133
10.93.4.6 count()	1133
10.93.4.7 number_selected()	1133
10.93.4.8 push_undo()	1134
10.93.4.9 pop_undo()	1134
10.93.4.10pop_redo()	1134
10.93.4.11print()	1134
10.93.4.12play()	1134
10.93.4.13add()	1136
10.93.4.14adjust_offsets_to_length()	1136
10.93.4.15split() [1/2]	1137
10.93.4.16half_split()	1137
10.93.4.17exact_split()	1137
10.93.4.18grow()	1137
10.93.4.19remove()	1138
10.93.4.20get_state()	1138
10.93.4.21select() [1/2]	1138
10.93.4.22unselect() [1/3]	1139

10.93.4.23	unselect() [2/3]	1139
10.93.4.24	intersect() [1/2]	1139
10.93.4.25	intersect() [2/2]	1140
10.93.4.26	remove_selected()	1140
10.93.4.27	copy_selected()	1140
10.93.4.28	paste()	1140
10.93.4.29	move_selected()	1140
10.93.4.30	get_selected_start()	1141
10.93.4.31	get_selected_end()	1141
10.93.4.32	get_maximum()	1142
10.93.4.33	move()	1142
10.93.4.34	copy()	1142
10.93.4.35	clear()	1143
10.93.4.36	next()	1143
10.93.4.37	next_trigger()	1144
10.93.4.38	reset_draw_trigger_marker()	1144
10.93.4.39	set_trigger_paste_tick()	1144
10.93.4.40	get_trigger_paste_tick()	1144
10.93.4.41	adjust_offset()	1144
10.93.4.42	split() [2/2]	1145
10.93.4.43	select() [2/2]	1145
10.93.4.44	unselect() [3/3]	1145
10.93.5	Friends And Related Function Documentation	1146
10.93.5.1	midi_container	1146
10.93.5.2	midifile	1146
10.93.5.3	sequence	1146
10.93.5.4	Seq24PerfInput	1146
10.93.5.5	FruityPerfInput	1146
10.93.6	Field Documentation	1146
10.93.6.1	m_parent	1147

10.93.6.2 m_triggers	1147
10.93.6.3 m_number_selected	1147
10.93.6.4 m_clipboard	1147
10.93.6.5 m_undo_stack	1147
10.93.6.6 m_redo_stack	1147
10.93.6.7 m_iterator_play_trigger	1147
10.93.6.8 m_iterator_draw_trigger	1147
10.93.6.9 m_trigger_copied	1148
10.93.6.10m_paste_tick	1148
10.93.6.11m_ppqn	1148
10.93.6.12m_length	1148
10.94seq64::user_instrument Class Reference	1148
10.94.1 Detailed Description	1149
10.94.2 Constructor & Destructor Documentation	1149
10.94.2.1 user_instrument() [1/2]	1149
10.94.2.2 user_instrument() [2/2]	1149
10.94.3 Member Function Documentation	1150
10.94.3.1 operator=()	1150
10.94.3.2 is_valid()	1150
10.94.3.3 set_defaults()	1150
10.94.3.4 name()	1150
10.94.3.5 controller_count()	1150
10.94.3.6 controller_max()	1150
10.94.3.7 controller_name()	1150
10.94.3.8 controller_active()	1151
10.94.3.9 set_controller()	1151
10.94.3.10set_name()	1151
10.94.3.11copy_definitions()	1152
10.94.4 Field Documentation	1152
10.94.4.1 m_is_valid	1152

10.94.4.2 m_controller_count	1152
10.94.4.3 m_instrument_def	1152
10.95seq64::user_instrument_t Struct Reference	1152
10.95.1 Field Documentation	1153
10.95.1.1 instrument	1153
10.95.1.2 controllers	1153
10.95.1.3 controllers_active	1153
10.96seq64::user_midi_bus Class Reference	1153
10.96.1 Detailed Description	1154
10.96.2 Constructor & Destructor Documentation	1154
10.96.2.1 user_midi_bus() [1/2]	1154
10.96.2.2 user_midi_bus() [2/2]	1155
10.96.3 Member Function Documentation	1155
10.96.3.1 operator=()	1155
10.96.3.2 is_valid()	1155
10.96.3.3 set_defaults()	1155
10.96.3.4 name()	1156
10.96.3.5 channel_count() [1/2]	1156
10.96.3.6 channel_count() [2/2]	1156
10.96.3.7 channel_max()	1156
10.96.3.8 instrument()	1156
10.96.3.9 set_instrument()	1157
10.96.3.10set_name()	1157
10.96.3.11copy_definitions()	1157
10.96.4 Field Documentation	1157
10.96.4.1 m_is_valid	1157
10.96.4.2 m_channel_count	1158
10.96.4.3 m_midi_bus_def	1158
10.97seq64::user_midi_bus_t Struct Reference	1158
10.97.1 Field Documentation	1158

10.97.1.1 alias	1158
10.97.1.2 instrument	1158
10.98seq64::user_settings Class Reference	1158
10.98.1 Detailed Description	1168
10.98.2 Member Typedef Documentation	1169
10.98.2.1 Busses	1169
10.98.2.2 Bussliterator	1169
10.98.2.3 BussConstliterator	1169
10.98.2.4 Instruments	1169
10.98.2.5 Instrumentliterator	1169
10.98.2.6 InstrumentConstliterator	1169
10.98.3 Member Enumeration Documentation	1169
10.98.3.1 mainwid_grid_style_t	1169
10.98.4 Constructor & Destructor Documentation	1170
10.98.4.1 user_settings() [1/2]	1170
10.98.4.2 user_settings() [2/2]	1170
10.98.5 Member Function Documentation	1170
10.98.5.1 operator=()	1170
10.98.5.2 set_defaults()	1170
10.98.5.3 normalize()	1171
10.98.5.4 add_bus()	1171
10.98.5.5 add_instrument()	1171
10.98.5.6 bus()	1171
10.98.5.7 instrument()	1171
10.98.5.8 bus_count()	1171
10.98.5.9 set_bus_instrument()	1172
10.98.5.10bus_instrument()	1172
10.98.5.11bus_name()	1172
10.98.5.12nstrument_count()	1172
10.98.5.13set_instrument_controllers()	1172

10.98.5.14	instrument_name() [1/2]	1172
10.98.5.15	instrument_name() [2/2]	1173
10.98.5.16	instrument_controller_active()	1173
10.98.5.17	controller_active()	1173
10.98.5.18	instrument_controller_name()	1173
10.98.5.19	controller_name()	1173
10.98.5.20	comments_block()	1173
10.98.5.21	clear_comments()	1174
10.98.5.22	append_comment_line()	1174
10.98.5.23	window_scale() [1/2]	1174
10.98.5.24	window_scaled_down()	1174
10.98.5.25	scale_size()	1174
10.98.5.26	grid_style() [1/2]	1174
10.98.5.27	grid_is_normal()	1174
10.98.5.28	grid_is_white()	1175
10.98.5.29	grid_is_black()	1175
10.98.5.30	grid_brackets() [1/2]	1175
10.98.5.31	mainwnd_rows() [1/2]	1175
10.98.5.32	mainwnd_cols() [1/2]	1175
10.98.5.33	s_variset()	1175
10.98.5.34	s_default_mainwid_size()	1175
10.98.5.35	seqs_in_set()	1175
10.98.5.36	mute_tracks()	1176
10.98.5.37	max_sets() [1/2]	1176
10.98.5.38	max_sequence()	1176
10.98.5.39	text_x() [1/2]	1176
10.98.5.40	text_y() [1/2]	1176
10.98.5.41	seqchars_x() [1/2]	1176
10.98.5.42	seqchars_y() [1/2]	1176
10.98.5.43	seqarea_x() [1/2]	1176

10.98.5.44seqarea_y() [1/2]	1177
10.98.5.45seqarea_seq_x() [1/2]	1177
10.98.5.46seqarea_seq_y() [1/2]	1177
10.98.5.47mainwid_border() [1/2]	1177
10.98.5.48mainwid_spacing() [1/2]	1177
10.98.5.49mainwid_x()	1177
10.98.5.50mainwid_y()	1177
10.98.5.51mainwnd_x()	1177
10.98.5.52mainwnd_y()	1178
10.98.5.53mainwid_border_x()	1178
10.98.5.54mainwid_border_y()	1178
10.98.5.55control_height() [1/2]	1178
10.98.5.56zoom() [1/2]	1178
10.98.5.57zoom() [2/2]	1178
10.98.5.58global_seq_feature() [1/2]	1178
10.98.5.59global_seq_feature() [2/2]	1179
10.98.5.60seqedit_scale() [1/2]	1179
10.98.5.61seqedit_scale() [2/2]	1179
10.98.5.62seqedit_key() [1/2]	1179
10.98.5.63seqedit_key() [2/2]	1179
10.98.5.64seqedit_bgsequence() [1/2]	1179
10.98.5.65seqedit_bgsequence() [2/2]	1179
10.98.5.66use_new_font() [1/2]	1180
10.98.5.67allow_two_perfedits() [1/2]	1180
10.98.5.68perf_h_page_increment() [1/2]	1180
10.98.5.69perf_v_page_increment() [1/2]	1180
10.98.5.70progress_bar_colored() [1/2]	1180
10.98.5.71progress_bar_thick() [1/2]	1180
10.98.5.72inverse_colors() [1/2]	1180
10.98.5.73window_redraw_rate() [1/2]	1180

10.98.5.74	use_more_icons() [1/2]	1181
10.98.5.75	block_rows() [1/2]	1181
10.98.5.76	block_columns() [1/2]	1181
10.98.5.77	block_independent() [1/2]	1181
10.98.5.78	save_user_config() [1/2]	1181
10.98.5.79	save_user_config() [2/2]	1181
10.98.5.80	grid_brackets() [2/2]	1181
10.98.5.81	window_scale() [2/2]	1182
10.98.5.82	grid_style() [2/2]	1182
10.98.5.83	mainwnd_rows() [2/2]	1182
10.98.5.84	mainwnd_cols() [2/2]	1182
10.98.5.85	max_sets() [2/2]	1182
10.98.5.86	text_x() [2/2]	1182
10.98.5.87	text_y() [2/2]	1183
10.98.5.88	seqchars_x() [2/2]	1183
10.98.5.89	seqchars_y() [2/2]	1183
10.98.5.90	seqarea_x() [2/2]	1183
10.98.5.91	seqarea_y() [2/2]	1183
10.98.5.92	seqarea_seq_x() [2/2]	1183
10.98.5.93	seqarea_seq_y() [2/2]	1183
10.98.5.94	mainwid_border() [2/2]	1184
10.98.5.95	mainwid_spacing() [2/2]	1184
10.98.5.96	control_height() [2/2]	1184
10.98.5.97	dump_summary()	1184
10.98.5.98	midi_ppqn() [1/2]	1184
10.98.5.99	midi_beats_per_bar() [1/2]	1184
10.98.5.100	midi_bpm_minimum() [1/2]	1184
10.98.5.101	midi_beats_per_minute() [1/2]	1185
10.98.5.102	midi_bpm_maximum() [1/2]	1185
10.98.5.103	midi_beat_width() [1/2]	1185

10.98.5.104	midi_buss_override() [1/2]	1185
10.98.5.105	velocity_override() [1/2]	1185
10.98.5.106	ppm_precision() [1/2]	1185
10.98.5.107	ppm_step_increment() [1/2]	1185
10.98.5.108	ppm_page_increment() [1/2]	1185
10.98.5.109	min_zoom()	1186
10.98.5.110	max_zoom()	1186
10.98.5.111	baseline_ppqn()	1186
10.98.5.112	option_daemonize() [1/2]	1186
10.98.5.113	option_use_logfile() [1/2]	1186
10.98.5.114	option_logfile() [1/2]	1186
10.98.5.115	work_around_play_image() [1/2]	1186
10.98.5.116	work_around_transpose_image() [1/2]	1187
10.98.5.117	key_height() [1/2]	1187
10.98.5.118	use_new_font() [2/2]	1187
10.98.5.119	allow_two_perfedits() [2/2]	1187
10.98.5.120	perf_h_page_increment() [2/2]	1187
10.98.5.121	perf_v_page_increment() [2/2]	1187
10.98.5.122	progress_bar_colored() [2/2]	1187
10.98.5.123	progress_bar_thick() [2/2]	1188
10.98.5.124	inverse_colors() [2/2]	1188
10.98.5.125	window_redraw_rate() [2/2]	1188
10.98.5.126	use_more_icons() [2/2]	1188
10.98.5.127	lock_rows() [2/2]	1188
10.98.5.128	lock_columns() [2/2]	1188
10.98.5.129	lock_independent() [2/2]	1188
10.98.5.130	option_daemonize() [2/2]	1189
10.98.5.131	option_use_logfile() [2/2]	1189
10.98.5.132	option_logfile() [2/2]	1189
10.98.5.133	work_around_play_image() [2/2]	1189

10.98.5.134	work_around_transpose_image() [2/2]	1189
10.98.5.135	key_height() [2/2]	1189
10.98.5.136	midi_ppqn() [2/2]	1189
10.98.5.137	midi_buss_override() [2/2]	1190
10.98.5.138	velocity_override() [2/2]	1190
10.98.5.139	bpm_precision() [2/2]	1190
10.98.5.140	bpm_step_increment() [2/2]	1190
10.98.5.141	bpm_page_increment() [2/2]	1190
10.98.5.142	mainwid_width()	1190
10.98.5.143	mainwid_height()	1191
10.98.5.144	mainwid_width_fudge()	1191
10.98.5.145	mainwid_height_fudge()	1191
10.98.5.146	midi_beats_per_bar() [2/2]	1191
10.98.5.147	midi_bpm_minimum() [2/2]	1191
10.98.5.148	midi_beats_per_minute() [2/2]	1191
10.98.5.149	midi_bpm_maximum() [2/2]	1192
10.98.5.150	midi_beat_width() [2/2]	1192
10.98.5.151	private_bus()	1192
10.98.5.152	private_instrument()	1192
10.98.6	Friends And Related Function Documentation	1192
10.98.6.1	midifile	1192
10.98.6.2	userfile	1192
10.98.6.3	parse_o_options	1192
10.98.7	Field Documentation	1193
10.98.7.1	m_midi_buses	1193
10.98.7.2	m_instruments	1193
10.98.7.3	m_comments_block	1193
10.98.7.4	m_grid_style	1193
10.98.7.5	m_grid_brackets	1194
10.98.7.6	m_mainwnd_rows	1194

10.98.7.7 m_mainwnd_cols	1194
10.98.7.8 m_max_sets	1194
10.98.7.9 m_window_scale	1194
10.98.7.10m_mainwid_border	1194
10.98.7.11m_mainwid_spacing	1194
10.98.7.12m_control_height	1195
10.98.7.13m_current_zoom	1195
10.98.7.14m_global_seq_feature_save	1195
10.98.7.15m_seqedit_scale	1195
10.98.7.16m_seqedit_key	1195
10.98.7.17m_seqedit_bgsequence	1196
10.98.7.18m_use_new_font	1196
10.98.7.19m_allow_two_perfedits	1196
10.98.7.20m_h_perf_page_increment	1196
10.98.7.21m_v_perf_page_increment	1196
10.98.7.22m_progress_bar_colored	1196
10.98.7.23m_progress_bar_thick	1196
10.98.7.24m_inverse_colors	1197
10.98.7.25m_window_redraw_rate_ms	1197
10.98.7.26m_use_more_icons	1197
10.98.7.27m_mainwid_block_rows	1197
10.98.7.28m_mainwid_block_cols	1197
10.98.7.29m_mainwid_block_independent	1197
10.98.7.30m_text_x	1198
10.98.7.31m_text_y	1198
10.98.7.32m_seqchars_x	1198
10.98.7.33m_seqchars_y	1198
10.98.7.34m_midi_ppqn	1198
10.98.7.35m_midi_beats_per_measure	1199
10.98.7.36m_midi_bpm_minimum	1199

10.98.7.37m_midi_beats_per_minute	1199
10.98.7.38m_midi_bpm_maximum	1199
10.98.7.39m_midi_beat_width	1199
10.98.7.40m_midi_buss_override	1199
10.98.7.41m_velocity_override	1200
10.98.7.42m_bpm_precision	1200
10.98.7.43m_bpm_step_increment	1200
10.98.7.44m_bpm_page_increment	1200
10.98.7.45m_total_seqs	1200
10.98.7.46m_seqs_in_set	1201
10.98.7.47m_gmute_tracks	1201
10.98.7.48m_max_sequence	1201
10.98.7.49m_seqarea_x	1201
10.98.7.50m_seqarea_y	1201
10.98.7.51m_seqarea_seq_x	1201
10.98.7.52m_seqarea_seq_y	1202
10.98.7.53m_mainwid_x	1202
10.98.7.54m_mainwid_y	1202
10.98.7.55m_mainwnd_x	1202
10.98.7.56m_mainwnd_y	1202
10.98.7.57m_save_user_config	1202
10.98.7.58mc_min_zoom	1203
10.98.7.59mc_max_zoom	1203
10.98.7.60mc_baseline_ppqn	1203
10.98.7.61m_user_option_daemonize	1203
10.98.7.62m_user_use_logfile	1203
10.98.7.63m_user_option_logfile	1203
10.98.7.64m_work_around_play_image	1203
10.98.7.65m_work_around_transpose_image	1204
10.98.7.66m_user_ui_key_height	1204
10.99seq64::userfile Class Reference	1204
10.99.1 Constructor & Destructor Documentation	1205
10.99.1.1 userfile()	1205
10.99.1.2 ~userfile()	1205
10.99.2 Member Function Documentation	1205
10.99.2.1 parse()	1205
10.99.2.2 write()	1206
10.99.2.3 dump_setting_summary()	1206

Chapter 1

Sequencer64 Classes and Functions

Author(s) Chris Ahlstrom 2018-01-14

1.1 Introduction

Sequencer64 is a major cleanup, refactoring, and documentation of the *Seq24* live-play MIDI sequencer.

The current document, generated by Doxygen, describes the functions, classes, modules, and other entities used in this project.

Also read the ROADMAP, README, and contrib/bugs_to_investigate files to understand the genesis of this project and the things that still need to be done with Sequencer64.

Also, we have pretty deeply documented *Seq24* and *Sequencer64* with PDF files that can be generated by git-cloning the following projects, installing a number of tools related to PDF and LaTeX, and running "make":

- <https://github.com/ahlstromcj/seq24-doc.git>
- <https://github.com/ahlstromcj/sequencer64-doc.git>

These project also have prebuilt PDFs should one not want to bother building them.

In the present document, we've left out a some side-code to cut down on the size of the document. Still, the resulting PDF is over 1000 pages long.

Some useful references:

- <http://acad.carleton.edu/courses/musc108-00-f14/pages/04/04StandardMIDIFiles.html>
- <http://www.midimusicadventures.com/qs/midi-zips/soundtracks/kq6gm.zip>

Chapter 2

Licenses

Library This application and its libraries, sub-applications, and documents.

Author(s) Chris Ahlstrom 2015-09-10

2.1 License Terms for the This Project.

Wherever the tag `$XPC_SUITE_GPL_LICENSE$` appears, or wherever reference to the GPL licensing scheme (any version) is mentioned, substitute the appropriate license text, depending on whether the project is a library, application, documentation, or server software. We're not going to include paragraphs of licensing information in every module; you are responsible for coming here to read the licensing information.

These licenses apply to each sub-project and file artifact in the project with which this license description was packaged.

Wherever the term **XPC** is encountered in this project, it refers to my projects, which go beyond the package that contains this document.

2.2 XPC Application License

The **XPC** application license is either the **GNU GPLv2.** or the **GNU GPLv3.** Generally, projects that originate with me use the latter language, while projects I have extended may specify the former license.

Copyright (C) 2015-2015 by Chris Ahlstrom

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the

Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor
Boston, MA 02110-1301, USA.

The text of the GNU GPL version 3 license can also be found here:

<http://www.gnu.org/licenses/gpl-3.0.txt>

2.3 XPC Library License

The **XPC** library license is the **GNU LGPLv3**.

Copyright (C) 2015-2015 by Chris Ahlstrom

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Lesser Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the

```
Free Software Foundation, Inc.  
51 Franklin Street, Fifth Floor  
Boston, MA 02110-1301, USA.
```

The text of the GNU LGPL version 3 license can also be found here:

<http://www.gnu.org/licenses/lgpl-3.0.txt>

2.4 XPC Documentation License

The **XPC** documentation license is the **GNU FDLv1.3**.

Copyright (C) 2015-2015 by Chris Ahlstrom

This documentation is free documentation; you can redistribute it and/or modify it under the terms of the GNU Free Documentation License as published by the Free Software Foundation; either version 1.3 of the License, or (at your option) any later version.

This documentation is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Free Documentation License for more details.

You should have received a copy of the GNU Free Documentation License along with this documentation; if not, write to the

```
Free Software Foundation, Inc.  
51 Franklin Street, Fifth Floor  
Boston, MA 02110-1301, USA.
```

The text of the GNU FDL version 1.3 license can also be found here:

<http://www.gnu.org/licenses/fdl.txt>

2.5 XPC Affero License

The **XPC** "Affero" license is the **GNU AGPLv3**.

Copyright (C) 2015-2015 by Chris Ahlstrom

This server software is free server software; you can redistribute it and/or modify it under the terms of the GNU Affero General Public License as published by the Free Software Foundation; either version 1.3 of the License, or (at your option) any later version.

This documentation is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Free Documentation License for more details.

You should have received a copy of the GNU Affero General Public License along with this server software; if not, write to the

```
Free Software Foundation, Inc.  
51 Franklin Street, Fifth Floor  
Boston, MA 02110-1301, USA.
```

The text of the GNU AGPL version 3 license can also be found here:

<http://www.gnu.org/licenses/agpl-3.0.txt>

At the present time, no **XPC** project uses the "Affero" license.

2.6 XPC License Summary

Include one of these licenses in your Doxygen documentation with one of the following Doxygen tags specified above:

```
\ref gpl_license_subproject  
\ref gpl_license_application  
\ref gpl_license_library  
\ref gpl_license_documentation  
\ref gpl_license_affero
```

For more information on navigating GNU licensing, see this page:

<http://www.gnu.org/licenses/>

Copies of these licenses (and some logos) are provided in the `licenses` directory of the main project (or you can search for them at [gnu.org](http://www.gnu.org)).

Chapter 3

Todo List

File `calculations.cpp`

There are additional user-interface and MIDI scaling variables in the perfroll module that we need to move here.

File `daemonize.cpp`

There is a service wrapper available under Win32. It is called "srvhost.exe". At this time, we *still* don't know how to use it, but it is available, and Windows XP seems to use it quite a bit.

File `perfnames.cpp`

When bringing up this dialog, and starting play from it, some extra horizontal lines are drawn for some of the sequences. This happens even in seq24, so this is long standing behavior. Is it useful, and how? Where is it done? In perfroll?

File `rc_settings.cpp`

Kepler34 has two more settings values: [midi-clock-mod-ticks], [note-resume] and [key-height]. The latter sounds more like a "usr" setting.

File `rc_settings.hpp`

Consolidate the usr and rc settings classes, or at least have a base class for common elements like "[comments]".

Global `seq64::editable_events::save_events ()`

Consider what to do about the `sequence::m_is_modified` flag.

Global `seq64::event::set_status_keep_channel (midibyte eventcode)`

THIS function WILL set a BOGUS CHANNEL on events \geq SYSEX !!!!!

Global `seq64::eventedit::handle_save ()`

Could also support writing the events to a new sequence, for added flexibility.

Global `seq64::font::init (Glib::RefPtr< Gdk::Window > windo)`

Can we scale these images via `scale_simple(newwidth, newhight, Gtk::INTERP_BILINEAR)`

Global `seq64::gui_palette_gtk2::gui_palette_gtk2 ()`

Use an array of colors instead of this switch.

Global `seq64::jack_assistant::deinit ()`

Note that we still need a way to call `jack_release_timebase()` when the user turns off the "JACK Master" status of Sequencer64.

Global `seq64::jack_timebase_callback (jack_transport_state_t state, jack_nframes_t nframes, jack_position_t *pos, int new_pos, void *arg)`

Shouldn't we process the first clause ONLY if `new_pos` is true?

Global `seq64::mainwnd::edit_field_has_focus () const`

We may have to revisit this one to add the Adjustment objects and the extra objects created in multi-wid mode.

Global `seq64::mainwnd::mainwnd` (perform &p, bool allowperf2=true, int ppqn=SEQ64_USE_DEFAULT_PPQN, int mainwid_rows=1, int mainwid_cols=1, bool mainwid_indep=false)

Offload most of the work into an initialization function like options does; make the perform parameter a reference.

Global `seq64::mainwnd::on_key_release_event` (GdkEventKey *ev)

Test this functionality in old and new application.

Global `seq64::mainwnd::tempo_log` ()

Upgrade this so that a Ctrl-click calls `toggle_tempo_record`, so that we can eliminate a tempo button; there are too many buttons.

Global `seq64::perfedit::perfedit` (perform &p, bool second_perfeddit=false, int ppqn=SEQ64_USE_DEFAULT_PPQN)

Offload most of the work into an initialization function like options does.

Global `seq64::perform::add_sequence` (sequence *seq, int perf)

Shouldn't we wrap around the sequence list if we can't find an empty sequence slot after `prefnum`?

This function needs some deeper analysis against the original, in my opinion.

Global `seq64::perform::launch` (int ppqn)

We probably need a bpm parameter for consistency at some point.

Global `seq64::perform::lookup_keyevent_key` (int seqnum)

In the context of pattern keys, we should replace `c_seqs_in_set` with a better-named value; if sets are actually larger than that, due to the "sets" option, then we simply repeat the pattern here using a modifier key ["shifted", see `lookup_slot_key()`]; in other words, we're stuck on using 32 pattern hot-keys.

Global `seq64::perform::m_seqs` [c_max_sequence]

First, make the sequence array a vector, and second, put all of these flags into a structure and access those members indirectly.

Global `seq64::perform::pop_trigger_undo` ()

Look at `seq32/src/perform.cpp` and the `perform :: push_trigger_undo(track)` function, which has a track parameter that has a -1 values the supports all tracks. It requires two new vectors (one for undo, one for redo), two new flags (likewise). We've put this code in place, no longer macroed out, now permanent.

Global `seq64::perform::set_left_tick` (midipulse tick, bool setstart=true)

The `perform::m_one_measure` member is currently hardwired to `m_ppqn*4`.

Global `seq64::perform::set_overwrite_recording` (bool overwrite_active, int seq, bool toggle=false)

Might probably as well create `(bool rec_active, bool thru_active, sequence * s)`.

Global `seq64::perform::set_tick` (midipulse tick)

Do we really need `m_current_tick`???

Global `seq64::perftroll::set_ppqn` (int ppqn)

Resolve the issue of `c_perf_scale_x` versus `m_perf_scale_x` in `perftroll`.

Global `seq64::perftime::set_ppqn` (int ppqn)

We need make the 4 constant variable per the number of beats (quarter-notes) per bar, and also at least make 16 (4x4) a meaningful manifest constant.

Global `seq64::pulses_to_string` (midipulse p)

Still needs to be unit tested.

Global `seq64::pulses_to_timestring` (midipulse p, const `midi_timing` &timinginfo)

Still needs to be unit tested.

Global `seq64::reroute_stdio` (const `std::string` &logfile, bool close)

Implement "daemonizing" for Windows, including redirection to the Windows Event Log.

Global `seq64::seqdata::on_scroll_event` (GdkEventScroll *ev)

DOCUMENT the `seqdata` scrolling behavior in the documentation projects.

Global `seq64::seqedit::get_measures ()`

Create a `sequence::set_units()` function or a `sequence::get_measures()` function to forward to.

Global `seq64::seqedit::seqedit (perform &perf, sequence &seq, int pos, int ppqn=SEQ64_USE_DEFAULT↵_PPQN)`

Offload most of the work into an initialization function like `options` does.

Global `seq64::seqedit::set_beat_width (int bw)`

Check if verification is needed at this point.

Global `seq64::seqedit::set_beats_per_bar (int bpm)`

Check if verification is needed at this point.

Global `seq64::seqedit::set_measures (int lim)`

Check if verification is needed at this point.

Global `seq64::seqmenu::m_modified`

We need to make sure that the `perform` object is in control of the modification flag.

Global `seq64::seqmenu::seq_copy ()`

Can be offloaded to a `perform` member function that accepts a sequence clipboard non-const reference parameter.

Global `seq64::seqmenu::seq_cut ()`

A lot of `seq_cut()` can be offloaded to a (new) `perform` member function that takes a sequence clipboard non-const reference parameter.

Global `seq64::seqmenu::seq_paste ()`

All of `seq_paste()` can be offloaded to a (new) `perform` member function with a const clipboard reference parameter.

Global `seq64::seqroll::follow_progress ()`

If playback is disabled (such as by a trigger), then do not update the page;

- When it comes back, make sure we're on the correct page;
- When it stops, put the window back to the beginning, even if the beginning is not defined as "0".

Global `seq64::seqtime::update_pixmap ()`

Sizing needs to be controlled by font parameters. Instead of 19 or 20, estimate the width of 3 letters. Instead of 9 pixels down, use the height of the `seqtime` and the height of a character.

Global `seq64::sequence::add_chord (int chord, midipulse tick, midipulse len, int note)`

Add the ability to preserve the incoming velocity.

Global `seq64::sequence::get_minmax_note_events (int &lowest, int &highest)`

For efficiency, we should calculate this only when the event set changes, and save the results and return them if good.

Global `seq64::sequence::stream_event (event &ev)`

When we feel like debugging, we will replace the global is-playing call with the parent `perform`'s is-running call.

If the last event was a Note Off, we should clear it here, and how?

Global `seq64::triggers::next (midipulse &tick_on, midipulse &tick_off, bool &selected, midipulse &tick_↵offset)`

It would be a bit simpler to simply return a trigger object, wouldn't it?

Chapter 4

Deprecated List

Global [seq64::clock_tick_duration_bogus](#) (midibpm bpm, int ppqn)

This is a somewhat bogus calculation used only for "statistical" output in the old perform module. Name changed to reflect this unfortunate fact. Use [pulse_length_us\(\)](#) instead.

Chapter 5

Bug List

Global `seq64::perform::get_group_mute_state` (int gtrack)

We were not using the group track value if it was zero, but that is a legitimate value.

Global `seq64::perform::set_group_mute_state` (int gtrack, bool muted)

We were not using the group track value if it was zero, but that is a legitimate value.

Chapter 6

Namespace Index

6.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Gtk	27
seq64 Define this macro to use the new seq24 v	27

Chapter 7

Hierarchical Index

7.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

seq64::automutex	125
seq64::busarray	127
seq64::businfo	136
seq64::click	143
seq64::configfile	150
seq64::optionsfile	649
seq64::userfile	1204
Dialog	
seq64::options	642
DrawingArea	
seq64::gui_palette_gtk2	300
seq64::gui_drawingarea_gtk2	281
seq64::eventslots	237
seq64::maintime	405
seq64::mainwid	410
seq64::perfnames	673
seq64::perfroll	800
seq64::Seq24PerfInput	899
seq64::FruityPerfInput	263
seq64::perftime	822
seq64::seqdata	907
seq64::seqevent	958
seq64::FruitySeqEventInput	267
seq64::Seq24SeqEventInput	905
seq64::seqkeys	973
seq64::seqroll	998
seq64::FruitySeqRollInput	272
seq64::seqtime	1032
seq64::editable_events	169
Entry	
seq64::keybindentry	346
seq64::event	180
seq64::editable_event	154
seq64::event_list::event_key	205

seq64::event_list	207
exception	
seq64::rterror	867
seq64::font	257
seq64::gui_assistant	276
seq64::gui_assistant_gtk2	279
seq64::jack_assistant	320
seq64::jack_scratchpad	343
seq64::jack_status_pair_t	346
seq64::keys_perform	350
seq64::keys_perform_gtk2	382
seq64::keys_perform_transfer	385
seq64::keystroke	390
seq64::lash	396
seq64::mastermidibase	458
seq64::mastermidibus	480
seq64::editable_event::meta_length_t	484
seq64::midi_container	506
seq64::midi_list	554
seq64::midi_vector	584
seq64::midi_control	516
seq64::midi_info	526
seq64::midi_alsa_info	491
seq64::midi_jack_info	548
seq64::midi_jack_data	546
seq64::midi_measures	558
seq64::midi_message	561
seq64::midi_port_info	568
seq64::midi_queue	572
seq64::midi_splitter	575
seq64::midi_timing	580
seq64::midibase	588
seq64::midi_api	496
seq64::midi_alsa	485
seq64::midi_in_alsa	521
seq64::midi_out_alsa	564
seq64::midi_jack	537
seq64::midi_in_jack	523
seq64::midi_out_jack	566
seq64::rtmidi	870
seq64::rtmidi_in	878
seq64::rtmidi_out	897
seq64::midibus	610
seq64::midifile	614
seq64::mutex	640
seq64::condition_var	147
seq64::editable_event::name_value_t	641
ObjectBase	
seq64::seqmenu	985
seq64::mainwid	410
seq64::perfnames	673
seq64::perform	681
seq64::performcallback	799
seq64::mainwnd	425
seq64::midi_port_info::port_info_t	832

seq64::rc_settings	834
seq64::rect	860
seq64::rtmidi_in_data	881
seq64::rtmidi_info	887
seq64::sequence	1038
seq64::trigger	1123
seq64::triggers	1128
seq64::user_instrument	1148
seq64::user_instrument_t	1152
seq64::user_midi_bus	1153
seq64::user_midi_bus_t	1158
seq64::user_settings	1158
Window	
seq64::gui_window_gtk2	316
seq64::eventedit	221
seq64::lfownd	400
seq64::mainwnd	425
seq64::perfedit	654
seq64::seqedit	920

Chapter 8

Data Structure Index

8.1 Data Structures

Here are the data structures with brief descriptions:

seq64::automutex	Provides a mutex that locks automatically when created, and unlocks when destroyed	125
seq64::busarray	Holds a number of businfo objects	127
seq64::businfo	A new class to consolidate a number of bus-related arrays into one array	136
seq64::click	Encapsulates any possible mouse click	143
seq64::condition_var	A mutex works best in conjunction with a condition variable	147
seq64::configfile	This class is the abstract base class for optionsfile and userfile	150
seq64::editable_event	Provides for the management of MIDI editable events	154
seq64::editable_events	Provides for the management of an ordered collection MIDI editable events	169
seq64::event	Provides events for management of MIDI events	180
seq64::event_list::event_key	Provides a key value for an event map	205
seq64::event_list	Receptable for MIDI events	207
seq64::eventedit	This class supports an Event Editor that is used to tweak the details of events and get a better idea of the mix of events in a sequence	221
seq64::eventslots	This class implements the left-side list of events in the pattern event-edit window	237
seq64::font	This class provides a wrapper for rendering fonts that are encoded as a 16 x 16 pixmap file in XPM format	257
seq64::FruityPerfInput	Implements the performance input of that certain fruity sequencer that people seem to like . . .	263
seq64::FruitySeqEventInput	This class implements the interaction methods for the "fruity" mode of operation in the event panel of the seqroll	267

seq64::FruitySeqRollInput	
Implements the fruity mouse interaction paradigm for the seqroll	272
seq64::gui_assistant	
This class provides an interface for some of the GUI support needed in Sequencer64	276
seq64::gui_assistant_gtk2	
This class provides an interface for some of the Gtk/Gdk/Glib support needed in Sequencer64	279
seq64::gui_drawingarea_gtk2	
Implements the basic drawing areas of the application	281
seq64::gui_palette_gtk2	
Implements a stock palette of Gdk::Color elements	300
seq64::gui_window_gtk2	
This class supports a basic interface for Gtk::Window-derived objects	316
seq64::jack_assistant	
This class provides the performance mode JACK support	320
seq64::jack_scratchpad	
Provide a temporary structure for passing data and results between a perform and jack_assistant object	343
seq64::jack_status_pair_t	
Provides an internal type to make it easier to display a specific and accurate human-readable message when a JACK operation fails	346
seq64::keybindentry	
Class for management of application key-bindings	346
seq64::keys_perform	
This class supports the performance mode	350
seq64::keys_perform_gtk2	
This class supports the performance mode	382
seq64::keys_perform_transfer	
Provides a data-transfer structure to make it easier to fill in a keys_perform object's members using sscanf()	385
seq64::keystroke	
Encapsulates any practical keystroke	390
seq64::lash	
This class supports LASH operations, if compiled with LASH support (i.e	396
seq64::lfownd	
One LFO window class	400
seq64::maintime	
This class provides the drawing of the progress bar at the top of the main window, along with two "pills" that move in time with the beat and measure	405
seq64::mainwid	
This class implements the piano roll area of the application	410
seq64::mainwnd	
This class implements the functionality of the main window of the application, except for the Patterns Panel functionality, which is implemented in the mainwid class	425
seq64::mastermidibase	
The class that "supervises" all of the midibus objects?	458
seq64::mastermidibus	
The class that "supervises" all of the midibus objects	480
seq64::editable_event::meta_length_t	
Provides a type that contains the pair of values needed to get the Meta event's data length	484
seq64::midi_alsa	
This class implements the ALSA version of the midi_api	485
seq64::midi_alsa_info	
The class for handling ALSA MIDI input	491
seq64::midi_api	
Subclasses of midi_in_api and midi_out_api contain all API- and OS-specific code necessary to fully implement the rtmidi API	496
seq64::midi_container	
This class is the abstract base class for a container of MIDI track information	506

seq64::midi_control	This class (formerly a struct) contains the control information for sequences that make up a live set	516
seq64::midi_in_alsa	This class implements the ALSA version of a MIDI input object	521
seq64::midi_in_jack	The class for handling JACK MIDI input	523
seq64::midi_info	The class for holding basic information on the MIDI input and output ports currently present in the system	526
seq64::midi_jack	This class implements with JACK version of the midi_alsa object	537
seq64::midi_jack_data	Contains the JACK MIDI API data as a kind of scratchpad for this object	546
seq64::midi_jack_info	The class for handling JACK MIDI port enumeration	548
seq64::midi_list	This class is the std::list implementation of the midi_container	554
seq64::midi_measures	Provides a data structure to hold the numeric equivalent of the measures string "measures↔:beats:divisions" ("m:b:d")	558
seq64::midi_message	Provides a handy capsule for a MIDI message, based on the std::vector<unsigned char> data type from the RtMidi project	561
seq64::midi_out_alsa	This class implements the ALSA version of a MIDI output object	564
seq64::midi_out_jack	The JACK MIDI output API class	566
seq64::midi_port_info	A class for holding port information	568
seq64::midi_queue	Provides a queue of midi_message structures	572
seq64::midi_splitter	This class handles the parsing and writing of MIDI files	575
seq64::midi_timing	We anticipate the need to have a small structure holding the parameters needed to calculate MIDI times within an arbitrary song	580
seq64::midi_vector	This class is the std::vector implementation of the midi_container	584
seq64::midibase	This class implements with ALSA version of the midibase object	588
seq64::midibus	This class implements with rtmidi version of the midibus object	610
seq64::midifile	This class handles the parsing and writing of MIDI files	614
seq64::mutex	Simple wrapper for the pthread_mutex_t type used as a recursive mutex	640
seq64::editable_event::name_value_t	Provides a type that contains the pair of values needed for the various lookup maps that are needed to manage editable events	641
seq64::options	This class supports a full tabbed options dialog	642
seq64::optionsfile	Provides a file for reading and writing the application' main configuration file	649
seq64::perfedit	This class supports a Performance Editor that is used to arrange the patterns/sequences defined in the patterns panel	654

seq64::perfnames	This class implements the left-side keyboard in the patterns window	673
seq64::perform	This class supports the performance mode	681
seq64::performcallback	Provides for notification of events	799
seq64::perfroll	This class implements the performance roll user interface	800
seq64::perftime	This class implements drawing the piano time at the top of the "performance window" (the "song editor")	822
seq64::midi_port_info::port_info_t	Hold the information for a single port	832
seq64::rc_settings	This class contains the options formerly named "global_XXXXXX"	834
seq64::rect	Supports a simple rectangle and some common manipulations needed by the user-interface	860
seq64::rterror	Exception handling class for rtexmidi	867
seq64::rtmidi	The main class of the rtmidi API	870
seq64::rtmidi_in	A realtime MIDI input class	878
seq64::rtmidi_in_data	The rtmidi_in_data structure is used to pass private class data to the MIDI input handling function or thread	881
seq64::rtmidi_info	A class for enumerating MIDI clients and ports	887
seq64::rtmidi_out	A realtime MIDI output class	897
seq64::Seq24PerfInput	Implements the default (Seq24) performance input characteristics of this application	899
seq64::Seq24SeqEventInput	This structure implement the normal interaction methods for Seq24	905
seq64::seqdata	This class supports drawing piano-roll events on a window	907
seq64::seqedit	Implements the Pattern Editor, which has references to:	920
seq64::seqevent	Implements the piano event drawing area	958
seq64::seqkeys	This class implements the left side piano of the pattern/sequence editor	973
seq64::seqmenu	This class handles the right-click menu of the sequence slots in the pattern window	985
seq64::seqroll	Implements the piano roll section of the pattern editor	998
seq64::seqtime	This class implements the piano time, whatever that is	1032
seq64::sequence	Firstly a receptable for a single track of MIDI data read from a MIDI file or edited into a pattern	1038
seq64::trigger	This class hold a single trigger for a sequence object	1123
seq64::triggers	Receptable the triggers that can be used with a sequence object	1128
seq64::user_instrument	Provides data about the MIDI instruments, readable from the "user" configuration file	1148

[seq64::user_instrument_t](#)

This structure corresponds to [user-instrument-N] definitions in the `~/ .seq24usr` or `~/ .config/sequencer64/sequencer64.usr` file 1152

[seq64::user_midi_bus](#)

Provides data about the MIDI busses, readable from the "user" configuration file 1153

[seq64::user_midi_bus_t](#)

This structure corresponds to [user-midi-bus-0] definitions in the `~/ .seq24usr` ("user") file (`~/ .config/sequencer64/sequencer64.usr` in the latest version of the application) 1158

[seq64::user_settings](#)

Holds the current values of sequence settings and settings that can modify the number of sequences and the configuration of the user-interface 1158

[seq64::userfile](#)

Supports the user's `~/ .config/sequencer64/sequencer64.usr` and `~/ .seq24usr` configuration file 1204

Chapter 9

Namespace Documentation

9.1 Gtk Namespace Reference

9.2 seq64 Namespace Reference

Define this macro to use the new seq24 v.

Data Structures

- class [automutex](#)
Provides a mutex that locks automatically when created, and unlocks when destroyed.
- class [busarray](#)
Holds a number of businfo objects.
- class [businfo](#)
A new class to consolidate a number of bus-related arrays into one array.
- class [click](#)
Encapsulates any possible mouse click.
- class [condition_var](#)
A mutex works best in conjunction with a condition variable.
- class [configfile](#)
This class is the abstract base class for optionsfile and userfile.
- class [editable_event](#)
Provides for the management of MIDI editable events.
- class [editable_events](#)
Provides for the management of an ordered collection MIDI editable events.
- class [event](#)
Provides events for management of MIDI events.
- class [event_list](#)
The [event_list](#) class is a receptable for MIDI events.
- class [eventedit](#)
This class supports an Event Editor that is used to tweak the details of events and get a better idea of the mix of events in a sequence.
- class [eventslots](#)
This class implements the left-side list of events in the pattern event-edit window.

- class [font](#)
This class provides a wrapper for rendering fonts that are encoded as a 16 x 16 pixmap file in XPM format.
- class [FruityPerfInput](#)
Implements the performance input of that certain fruity sequencer that people seem to like.
- class [FruitySeqEventInput](#)
This class implements the interaction methods for the "fruity" mode of operation in the event panel of the seqroll.
- class [FruitySeqRollInput](#)
Implements the fruity mouse interaction paradigm for the seqroll.
- class [gui_assistant](#)
This class provides an interface for some of the GUI support needed in Sequencer64.
- class [gui_assistant_gtk2](#)
This class provides an interface for some of the Gtk/Gdk/Glib support needed in Sequencer64.
- class [gui_drawingarea_gtk2](#)
Implements the basic drawing areas of the application.
- class [gui_palette_gtk2](#)
Implements a stock palette of Gdk::Color elements.
- class [gui_window_gtk2](#)
This class supports a basic interface for Gtk::Window-derived objects.
- class [jack_assistant](#)
This class provides the performance mode JACK support.
- class [jack_scratchpad](#)
Provide a temporary structure for passing data and results between a perform and [jack_assistant](#) object.
- struct [jack_status_pair_t](#)
Provides an internal type to make it easier to display a specific and accurate human-readable message when a JACK operation fails.
- class [keybindentry](#)
Class for management of application key-bindings.
- class [keys_perform](#)
This class supports the performance mode.
- class [keys_perform_gtk2](#)
This class supports the performance mode.
- struct [keys_perform_transfer](#)
Provides a data-transfer structure to make it easier to fill in a [keys_perform](#) object's members using sscanf().
- class [keystroke](#)
Encapsulates any practical keystroke.
- class [lash](#)
This class supports LASH operations, if compiled with LASH support (i.e.
- class [lfownd](#)
One LFO window class.
- class [maintime](#)
This class provides the drawing of the progress bar at the top of the main window, along with two "pills" that move in time with the beat and measure.
- class [mainwid](#)
This class implements the piano roll area of the application.
- class [mainwnd](#)
This class implements the functionality of the main window of the application, except for the Patterns Panel functionality, which is implemented in the mainwid class.
- class [mastermidibase](#)
The class that "supervises" all of the midibus objects?
- class [mastermidibus](#)
The class that "supervises" all of the midibus objects.

- class [midi_alsa](#)
This class implements the ALSA version of the [midi_api](#).
- class [midi_alsa_info](#)
The class for handling ALSA MIDI input.
- class [midi_api](#)
Subclasses of [midi_in_api](#) and [midi_out_api](#) contain all API- and OS-specific code necessary to fully implement the [rtmidi](#) API.
- class [midi_container](#)
This class is the abstract base class for a container of MIDI track information.
- class [midi_control](#)
This class (formerly a struct) contains the control information for sequences that make up a live set.
- class [midi_in_alsa](#)
This class implements the ALSA version of a MIDI input object.
- class [midi_in_jack](#)
The class for handling JACK MIDI input.
- class [midi_info](#)
The class for holding basic information on the MIDI input and output ports currently present in the system.
- class [midi_jack](#)
This class implements with JACK version of the [midi_alsa](#) object.
- struct [midi_jack_data](#)
Contains the JACK MIDI API data as a kind of scratchpad for this object.
- class [midi_jack_info](#)
The class for handling JACK MIDI port enumeration.
- class [midi_list](#)
This class is the `std::list` implementation of the [midi_container](#).
- class [midi_measures](#)
Provides a data structure to hold the numeric equivalent of the measures string "measures:beats:divisions" ("m:b:d").
- class [midi_message](#)
Provides a handy capsule for a MIDI message, based on the `std::vector<unsigned char>` data type from the [RtMidi](#) project.
- class [midi_out_alsa](#)
This class implements the ALSA version of a MIDI output object.
- class [midi_out_jack](#)
The JACK MIDI output API class.
- class [midi_port_info](#)
A class for holding port information.
- class [midi_queue](#)
Provides a queue of [midi_message](#) structures.
- class [midi_splitter](#)
This class handles the parsing and writing of MIDI files.
- class [midi_timing](#)
We anticipate the need to have a small structure holding the parameters needed to calculate MIDI times within an arbitrary song.
- class [midi_vector](#)
This class is the `std::vector` implementation of the [midi_container](#).
- class [midibase](#)
This class implements with ALSA version of the midibase object.
- class [midibus](#)
This class implements with [rtmidi](#) version of the midibus object.
- class [midifile](#)
This class handles the parsing and writing of MIDI files.

- class [mutex](#)
The mutex class provides a simple wrapper for the pthread_mutex_t type used as a recursive mutex.
- class [options](#)
This class supports a full tabbed options dialog.
- class [optionsfile](#)
Provides a file for reading and writing the application's main configuration file.
- class [perfeddit](#)
This class supports a Performance Editor that is used to arrange the patterns/sequences defined in the patterns panel.
- class [perfnames](#)
This class implements the left-side keyboard in the patterns window.
- class [perform](#)
This class supports the performance mode.
- struct [performcallback](#)
Provides for notification of events.
- class [perfull](#)
This class implements the performance roll user interface.
- class [perftime](#)
This class implements drawing the piano time at the top of the "performance window" (the "song editor").
- class [rc_settings](#)
This class contains the options formerly named "global_XXXXXX".
- class [rect](#)
Supports a simple rectangle and some common manipulations needed by the user-interface.
- class [rterror](#)
Exception handling class for rtexmidi.
- class [rtmidi](#)
The main class of the rtmidi API.
- class [rtmidi_in](#)
A realtime MIDI input class.
- class [rtmidi_in_data](#)
The rtmidi_in_data structure is used to pass private class data to the MIDI input handling function or thread.
- class [rtmidi_info](#)
A class for enumerating MIDI clients and ports.
- class [rtmidi_out](#)
A realtime MIDI output class.
- class [Seq24PerfInput](#)
Implements the default (Seq24) performance input characteristics of this application.
- class [Seq24SeqEventInput](#)
This structure implements the normal interaction methods for Seq24.
- class [seqdata](#)
This class supports drawing piano-roll events on a window.
- class [seqedit](#)
Implements the Pattern Editor, which has references to:
- class [sequevent](#)
Implements the piano event drawing area.
- class [seqkeys](#)
This class implements the left side piano of the pattern/sequence editor.
- class [seqmenu](#)
This class handles the right-click menu of the sequence slots in the pattern window.
- class [seqroll](#)
Implements the piano roll section of the pattern editor.

- class [seqtime](#)
This class implements the piano time, whatever that is.
- class [sequence](#)
The sequence class is firstly a receptable for a single track of MIDI data read from a MIDI file or edited into a pattern.
- class [trigger](#)
This class hold a single trigger for a sequence object.
- class [triggers](#)
The triggers class is a receptable the triggers that can be used with a sequence object.
- class [user_instrument](#)
Provides data about the MIDI instruments, readable from the "user" configuration file.
- struct [user_instrument_t](#)
This structure corresponds to [user-instrument-N] definitions in the `~/ .seq24usr` or `~/ .config/sequencer64/sequsr` file.
- class [user_midi_bus](#)
Provides data about the MIDI busses, readable from the "user" configuration file.
- struct [user_midi_bus_t](#)
This structure corresponds to [user-midi-bus-0] definitions in the `~/ .seq24usr` ("user") file (`~/ .config/sequencer64/sequencer64 usr` in the latest version of the application).
- class [user_settings](#)
Holds the current values of sequence settings and settings that can modify the number of sequences and the configuration of the user-interface.
- class [userfile](#)
Supports the user's `~/ .config/sequencer64/sequencer64.usr` and `~/ .seq24usr` configuration file.

Typedefs

- typedef unsigned char [midibyte](#)
Provides a fairly common type definition for a byte value.
- typedef std::basic_string< [midibyte](#) > [midistring](#)
Provides a string specialization to explicitly use unsigned characters.
- typedef unsigned char [bussbyte](#)
Distinguishes a buss/bus number from other MIDI bytes.
- typedef unsigned short [midishort](#)
Distinguishes a short value from the unsigned short values implicit in short-valued MIDI numbers.
- typedef unsigned long [midilong](#)
Distinguishes a long value from the unsigned long values implicit in long-valued MIDI numbers.
- typedef char [colorbyte](#)
Provides a way to save a sequence palette color in a single byte.
- typedef long [midipulse](#)
Distinguishes a long value from the unsigned long values implicit in MIDI time measurements.
- typedef double [midibpm](#)
Provides the data type for BPM (beats per minute) values.
- typedef void(* [rterror_callback](#)) ([rterror::Type](#) type, const std::string &errmsg, void *userdata)
rtmidi error callback function prototype.
- typedef void(* [rtmidi_callback_t](#)) ([midi_message](#) &message, void *userdata)
MIDI caller callback function type definition.

Enumerations

- enum `wave_type_t` {
`WAVE_NONE`,
`WAVE_SINE`,
`WAVE_SAWTOOTH`,
`WAVE_REVERSE_SAWTOOTH`,
`WAVE_TRIANGLE` }

Provides a clear enumeration of wave types supported by the wave function.

- enum `seq_modifier_t` {
`SEQ64_NO_MASK`,
`SEQ64_SHIFT_MASK`,
`SEQ64_LOCK_MASK`,
`SEQ64_CONTROL_MASK`,
`SEQ64_MOD1_MASK`,
`SEQ64_MOD2_MASK`,
`SEQ64_MOD3_MASK`,
`SEQ64_MOD4_MASK`,
`SEQ64_MOD5_MASK`,
`SEQ64_BUTTON1_MASK`,
`SEQ64_BUTTON2_MASK`,
`SEQ64_BUTTON3_MASK`,
`SEQ64_BUTTON4_MASK`,
`SEQ64_BUTTON5_MASK`,
`SEQ64_SUPER_MASK`,
`SEQ64_HYPER_MASK`,
`SEQ64_META_MASK`,
`SEQ64_RELEASE_MASK`,
`SEQ64_MASK_MAX` }

Types of modifiers, essentially copied from `gtk-2.0/gdk/gdktypes.h`.

- enum `seq_event_type_t` {
`SEQ64_NOTHING`,
`SEQ64_DELETE`,
`SEQ64_DESTROY`,
`SEQ64_EXPOSE`,
`SEQ64_MOTION_NOTIFY`,
`SEQ64_BUTTON_PRESS`,
`SEQ64_2BUTTON_PRESS`,
`SEQ64_3BUTTON_PRESS`,
`SEQ64_BUTTON_RELEASE`,
`SEQ64_KEY_PRESS`,
`SEQ64_KEY_RELEASE`,
`SEQ64_SCROLL`,
`SEQ64_EVENT_LAST` }

Event types copied from `gtk-2.0/gdk/gdkevents.h` for use with this application.

- enum `seq_scroll_direction_t` {
`SEQ64_SCROLL_UP`,
`SEQ64_SCROLL_DOWN`,
`SEQ64_SCROLL_LEFT`,
`SEQ64_SCROLL_RIGHT` }

Types of scroll events, essentially copied from `gtk-2.0/gdk/gdkevents.h`.

- enum `clock_e` {
`e_clock_disabled`,
`e_clock_off`,
`e_clock_pos`,
`e_clock_mod` }

A clock enumeration, as used in the File / Options / MIDI Clock dialog.

- enum `interaction_method_t` {
`e_seq24_interaction`,
`e_fruity_interaction`,
`e_number_of_interactions` }

Provides mutually-exclusive codes for the mouse-handling used by the application.

- enum `mute_group_handling_t` {
`e_mute_group_stomp`,
`e_mute_group_preserve`,
`e_mute_group_max` }

Provides mutually-exclusive codes for handling the reading of mute-groups from the "rc" file versus the "MIDI" file.

- enum `c_music_scales` {
`c_scale_off`,
`c_scale_major`,
`c_scale_minor`,
`c_scale_harmonic_minor`,
`c_scale_melodic_minor`,
`c_scale_c_whole_tone`,
`c_scale_blues`,
`c_scale_major_pentatonic`,
`c_scale_minor_pentatonic`,
`c_scale_size` }

Corresponds to the small number of musical scales that the application can handle.

- enum `draw_type_t` {
`DRAW_FIN`,
`DRAW_NORMAL_LINKED`,
`DRAW_NOTE_ON`,
`DRAW_NOTE_OFF`,
`DRAW_TEMPO` }

Provides a set of methods for drawing certain items.

- enum `edit_mode_t` {
`EDIT_MODE_NOTE`,
`EDIT_MODE_DRUM` }

Provides two editing modes for a sequence.

- enum `loop_record_t` {
`LOOP_RECORD_LEGACY`,
`LOOP_RECORD_OVERWRITE`,
`LOOP_RECORD_EXPAND` }

Provides the supported looping recording modes.

- enum `mouse_action_e` {
`e_action_select`,
`e_action_draw`,
`e_action_grow` }

Mouse actions, for the Pattern Editor.

- enum `edit_action_t` {
`c_select_all_notes`,
`c_select_all_events`,
`c_select_inverse_notes`,
`c_select_inverse_events`,
`c_quantize_notes`,
`c_quantize_events`,
`c_tighten_events`,
`c_tighten_notes`,
`c_transpose_notes`,
`c_reserved`,
`c_transpose_h`,
`c_expand_pattern`,

```
c_compress_pattern,
c_select_even_notes,
c_select_odd_notes,
c_swing_notes }
```

Actions.

- enum `rtmidi_api` {
`RTMIDI_API_UNSPECIFIED`,
`RTMIDI_API_LINUX_ALSA`,
`RTMIDI_API_UNIX_JACK`,
`RTMIDI_API_MAXIMUM` }

MIDI API specifier arguments.

Functions

- void `swap` (`busarray` &buses0, `busarray` &buses1)
This free function swaps the contents of two busarray objects.
- std::string `wave_type_name` (`wave_type_t` wavetype)
Converts a wave type value to a string.
- int `extract_timing_numbers` (const std::string &s, std::string &part_1, std::string &part_2, std::string &part_3, std::string &fraction)
Extracts up to 4 numbers from a colon-delimited string.
- int `tokenize_string` (const std::string &source, std::vector< std::string > &tokens)
Tokenizes a string using the colon, space, or period as delimiters.
- std::string `pulses_to_string` (`midipulse` p)
Converts MIDI pulses (also known as ticks, clocks, or divisions) into a string.
- std::string `pulses_to_measurestring` (`midipulse` p, const `midi_timing` &seqparms)
Converts a MIDI pulse/ticks/clock value into a string that represents "measures:beats:ticks" ("measures:beats:division").
- bool `pulses_to_midi_measures` (`midipulse` p, const `midi_timing` &seqparms, `midi_measures` &measures)
Converts a MIDI pulse/ticks/clock value into a string that represents "measures:beats:ticks" ("measures:beats:division").
- std::string `pulses_to_timestring` (`midipulse` p, const `midi_timing` &timinginfo)
Converts a MIDI pulse/ticks/clock value into a string that represents "hours:minutes:seconds.fraction".
- std::string `pulses_to_timestring` (`midipulse` p, `midibpm` bpm, int ppqn, bool showus)
Converts a MIDI pulse/ticks/clock value into a string that represents "hours:minutes:seconds.fraction".
- `midipulse` `measurestring_to_pulses` (const std::string &measures, const `midi_timing` &seqparms)
Converts a string that represents "measures:beats:division" to a MIDI pulse/ticks/clock value.
- `midipulse` `midi_measures_to_pulses` (const `midi_measures` &measures, const `midi_timing` &seqparms)
Converts a string that represents "measures:beats:division" to a MIDI pulse/ticks/clock value.
- `midipulse` `timestring_to_pulses` (const std::string ×tring, int bpm, int ppqn)
Converts a time string to pulses.
- `midipulse` `string_to_midipulse` (const std::string &s)
Converts a string to a MIDI byte.
- std::string `shorten_file_spec` (const std::string &path, int leng)
Shortens a file-specification to make sure it is no longer than the provided length value.
- bool `string_not_void` (const std::string &s)
Tests that a string is not empty and has non-space characters.
- bool `string_is_void` (const std::string &s)
Tests that a string is empty or has only white-space characters.
- bool `strings_match` (const std::string &target, const std::string &x)

- Compares two strings for a form of semantic equality, for the purposes of `editable_event()`, for example.*

 - int `log2_time_sig_value` (int tsd)

Calculates the log-base-2 value of a number that is already a power of 2.
- int `zoom_power_of_2` (int ppqn)

Calculates a suitable starting zoom value for the given PPQN value.
- int `beat_pow2` (int logbase2)

Internal function for simple calculation of a power of 2 without a lot of math.
- midibyte `beat_log2` (int value)

Calculates the base-2 log of a number.
- double `tempo_us_from_bytes` (const midibyte t[3])

Calculates the tempo in microseconds from the bytes read from a Tempo event in the MIDI file.
- void `tempo_us_to_bytes` (midibyte t[3], int tempo_us)

Provide a way to convert a tempo value (microseconds per quarter note) into the three bytes needed as value in a Tempo meta event.
- midibyte `tempo_to_note_value` (midibpm tempovalue)

Converts a tempo value to a MIDI note value for the purpose of displaying a tempo value in the mainwid, seqdata section (hopefully!), and the perroll.
- midibpm `note_value_to_tempo` (midibyte note)

The inverse of `tempo_to_note_value()`.
- bool `ppqn_is_valid` (int ppqn)

Common code for handling PPQN settings.
- double `tempo_us_from_bpm` (midibpm bpm)

Converts tempo (e.g.
- midibpm `bpm_from_tempo_us` (double tempous)

This function calculates the effective beats-per-minute based on the value of a Tempo meta-event.
- midibpm `bpm_from_bytes` (midibyte t[3])

Provides a direct conversion from a midibyte array to the beats/minute value.
- double `pulse_length_us` (midibpm bpm, int ppqn)

Calculates pulse-length from the BPM (beats-per-minute) and PPQN (pulses-per-quarter-note) values.
- double `delta_time_us_to_ticks` (unsigned long us, midibpm bpm, int ppqn)

Converts delta time in microseconds to ticks.
- double `ticks_to_delta_time_us` (midipulse delta_ticks, midibpm bpm, int ppqn)

Converts the time in ticks ("clocks") to delta time in microseconds.
- double `clock_tick_duration_bogus` (midibpm bpm, int ppqn)

Calculates the duration of a clock tick based on PPQN and BPM settings.
- int `clock_ticks_from_ppqn` (int ppqn)

A simple calculation to convert PPQN to MIDI clock ticks.
- double `double_ticks_from_ppqn` (int ppqn)

A simple calculation to convert PPQN to MIDI clock ticks.
- midipulse `pulses_per_measure` (int ppqn=SEQ64_DEFAULT_PPQN)

Calculates the pulses per measure.
- midipulse `measures_to_ticks` (int bpb, int ppqn, int bw, int measures=1)

Calculates the length of an integral number of measures, in ticks.
- int `ticks_to_measures` (int bpb, int ppqn, int bw, midipulse ticks=192)

The inverse of `measures_to_ticks`.
- midipulse `pulse_divide` (midipulse numerator, midipulse denominator, midipulse &remainder)

Calculates the quotient and remainder of a midipulse division, which is a common operation in Sequencer64.
- double `wave_func` (double angle, wave_type_t wavetype)

Calculates a wave function for use as an LFO (low-frequency oscillator) for modifying data values in a sequence.
- bool `extract_port_names` (const std::string &fullname, std::string &clientname, std::string &portname)

Extracts the two names from the ALSA/JACK client/port name format, "[0] 128:0 clientname:portname".

- `std::string extract_bus_name` (const `std::string` &fullname)
Extracts the buss name from "bus:port".
- `std::string extract_port_name` (const `std::string` &fullname)
Extracts the port name from "bus:port".
- `std::string current_date_time` ()
Gets the current date/time.
- `bool help_check` (int argc, char *argv [])
Checks to see if the first option is a help or version argument, just so we can skip the "Reading configuration ..." messages.
- `bool parse_options_files` (perform &p, `std::string` &errmsg, int argc, char *argv [])
Provides the command-line option support, as well as some setup support, extracted from the main routine of Sequencer64.
- `bool parse_mute_groups` (perform &p, `std::string` &errmsg)
Like `parse_options_files()`, but reads only the [mute-group] section.
- `bool parse_o_options` (int argc, char *argv [])
Checks the putative command-line arguments for any of the new "options" options.
- `bool parse_log_option` (int argc, char *argv [])
Checks the putative command-line arguments for the "log" option.
- `int parse_command_line_options` (perform &p, int argc, char *argv [])
Parses the command-line options on behalf of the application.
- `bool write_options_files` (const perform &p)
Saves all options to the "rc" and "user" configuration files.
- `std::string build_details` ()
Generates a string describing the features of the build.
- `bool check_daemonize` (int argc, char *argv [])
- `uint32_t daemonize` (const `std::string` &appname, const `std::string` &cwd=".", int mask=0)
- `void undaemonize` (uint32_t previous_umask)
- `bool reroute_stdio` (const `std::string` &logfile, bool closen)
Alters the standard terminal file descriptors so that they either route to to a log file, under Linux or Windows.
- `bool is_note_off_velocity` (midibyte status, midibyte data)
This free function is used in the midifile module and in the `event::is_note_off_recorded()` member function.
- `std::string to_string` (const event &ev)
A free function to convert an event into an informative string, just enough to save some debugging time.
- `event create_tempo_event` (midipulse tick, midibpm tempo)
Creates and returns a tempo event.
- `bool file_access` (const `std::string` &targetfile, int mode)
- `bool file_exists` (const `std::string` &filename)
Checks a file for existence.
- `bool file_readable` (const `std::string` &filename)
Checks a file for readability.
- `bool file_writable` (const `std::string` &filename)
Checks a file for writability.
- `bool file_accessible` (const `std::string` &filename)
Checks a file for readability and writability.
- `bool file_executable` (const `std::string` &filename)
Checks a file for the ability to be executed.
- `bool file_is_directory` (const `std::string` &filename)
Checks a file to see if it is a directory.
- `bool make_directory` (const `std::string` &pathname)
A function to ensure that the `~/config/sequencer64` directory exists.
- `std::string get_current_directory` ()

- Provides the path name of the current working directory.*
- `std::string get_full_path` (const std::string &path)
 - Given a path, relative or not, this function returns the full path.*
- `std::string normalize_path` (const std::string &path, bool to_unix)
 - Makes sure that the path-name is a UNIX path, separated by forward slashes (the solidus).*
- `std::string strip_quotes` (const std::string &item)
 - Strips the double quotes from a string.*
- `std::string file_extension` (const std::string &path)
 - Gets a file extension, defined simply as the text after the last period in the path.*
- `bool strcasecmp` (const std::string &a, const std::string &b)
- `bool file_extension_match` (const std::string &path, const std::string &target)
 - Sees if file-extensions match, case-insensitively.*
- `void jack_shutdown_callback` (void *arg)
 - Global functions for JACK support and JACK sessions.*
- `void jack_timebase_callback` (jack_transport_state_t state, jack_nframes_t nframes, jack_position_t *pos, int new_pos, void *arg)
 - The JACK timebase function defined here sets the JACK position structure.*
- `int jack_transport_callback` (jack_nframes_t nframes, void *arg)
 - Implemented second patch for JACK Transport from freddix/seq24 GitHub project.*
- `jack_client_t * create_jack_client` (const std::string &clientname, const std::string &uuid)
 - A more full-featured initialization for a JACK client, which is meant to be called by the init() function.*
- `void show_jack_statuses` (unsigned bits)
 - Loops through the full set of JACK bits, showing the information for any bits that are set in the given parameter.*
- `long get_current_jack_position` (void *arg)
 - This function gets the current JACK position.*
- `void jack_session_callback` (jack_session_event_t *ev, void *arg)
 - Set the m_session_ev (event) value of the perform object.*
- `bool invalid_key` (unsigned key)
 - Free functions for keyboard support.*
- `void keyval_normalize` (keys_perform_transfer &k)
 - For the case in which the "rc" file is missing or corrupt, this function makes sure that each control key has a reasonable value.*
- `bool create_lash_driver` (perform &p, int argc, char **argv)
 - Creates and starts a lash object.*
- `lash * lash_driver` ()
 - Provides access to the lash object.*
- `void delete_lash_driver` ()
 - Deletes the last object.*
- `void millisleep` (unsigned long ms)
- `bool is_null_midipulse` (midipulse p)
 - Compares a midipulse value to SEQ64_NULL_MIDIPULSE.*
- `void * output_thread_func` (void *p)
 - Global functions defined in perform.cpp.*
- `void * input_thread_func` (void *myperf)
 - Set up the performance, and set the process to realtime privileges.*
- `rc_settings & rc` ()
 - Returns a reference to the global rc_settings object.*
- `user_settings & usr` ()
 - Returns a reference to the global user_settings object, for better encapsulation.*
- `int choose_ppqn` (int ppqn)
 - Common code for handling PPQN settings.*

- [midipulse timestring_to_pulses](#) (const std::string ×tring, [midibpm](#) bpm, int ppqn)
Converts a string that represents "hours:minutes:seconds.fraction" into a MIDI pulse/ticks/clock value.
- static std::string [get_compound_option](#) (const std::string &compound, std::string &optionname)
Gets a compound option argument.
- bool [set_current_directory](#) (const std::string &path)
Sets the current directory for the application.
- std::string [message_concatenate](#) (const char *m1, const char *m2)
This function concatenates two C string pointers and returns them as a string message.
- bool [info_message](#) (const std::string &msg)
Common-code for console messages.
- bool [error_message](#) (const std::string &msg)
Common-code for error messages.
- static bool [casecompare](#) (char a, char b)
- int [jack_dummy_callback](#) (jack_nframes_t nframes, void *arg)
Provides a dummy callback.
- const std::string & [seq_app_name](#) ()
Returns the name of the application.
- const std::string & [seq_client_name](#) ()
Returns the name of the client for the application.
- const std::string & [seq_version](#) ()
Returns the version of the application.
- static std::string [make_section_name](#) (const std::string &label, int value)
Provides a purely internal, ad hoc helper function to create numbered section names for the userfile class.
- [font](#) & [font_render](#) ()
The p_font_renderer pointer was once created in the main module, sequencer64.cpp.
- Gtk::Adjustment & [adjustment_dummy](#) ()
Provides a way to provide a dummy Gtk::Adjustment object, but not create one until it is actually needed, so that the Glib/Gtk infrastructure is ready for it.
- bool [is_ctrl_key](#) (GdkEventKey *ev)
Encapsulates the safe test for the control key, as described here: <https://developer.gnome.org/gtk3/stable/checklist-modifiers.html>.
- bool [is_shift_key](#) (GdkEventKey *ev)
Encapsulates the safe test for the shift key.
- bool [is_no_modifier](#) (GdkEventScroll *ev)
Encapsulates the safe test for no modifier keys, for a scroll event.
- bool [is_ctrl_key](#) (GdkEventScroll *ev)
Encapsulates the safe test for the control key for scrolling.
- bool [is_shift_key](#) (GdkEventScroll *ev)
Encapsulates the safe test for the shift key.
- bool [is_ctrl_key](#) (GdkEventButton *ev)
Encapsulates the safe test for the control key for buttons.
- bool [is_shift_key](#) (GdkEventButton *ev)
Encapsulates the safe test for the shift key.
- bool [is_ctrl_shift_key](#) (GdkEventButton *ev)
Encapsulates the safe test for the ctrl-shift key combination.
- bool [is_super_key](#) (GdkEventButton *ev)
Encapsulates the test for the super (mod4, windows) key for buttons.
- void [test_widget_click](#) (GtkWidget *w)
Not sure where I was going with this one!
- bool [is_left_drag](#) (GdkEventMotion *ev)
Tests for a left-drag motion being in force.

- bool [is_drag_motion](#) (GdkEventMotion *ev)
Tests for a left-, right-, and middle-drag motion being in force.
- std::string [keyval_name](#) (unsigned key)
Free functions for keyboard support.
- void [update_mainwid_sequences](#) ()
This global function in the [seq64](#) namespace calls `mainwid :: update_sequences_on_window()`, if the global `mainwid` object exists.
- void [update_perfedit_sequences](#) ()
This global function in the [seq64](#) namespace calls `perfedit :: draw_sequences()`, if the global `perfedit` objects exist.
- int [FF_RW_timeout](#) (void *arg)
This global function in the [seq64](#) namespace is passed to the `gtk_timeout_add` callback.
- static long [clamp](#) (long val, long low, long hi)
An internal function used by the [FruitySeqRollInput](#) class.
- void [silence_jack_errors](#) (bool silent=true)
- void [silence_jack_info](#) (bool silent=true)
- std::string [midi_api_name](#) (int i)
- int [midi_probe](#) ()
- bool [midi_input_test](#) (rtmidi_info &info, int portindex)

Variables

- std::string [c_controller_names](#) [SEQ64_MIDI_COUNT_MAX]
Provides the default names of MIDI controllers.
- const [midibyte](#) [EVENT_STATUS_BIT](#)
This highest bit of the status byte is always 1.
- const [midibyte](#) [EVENT_ANY](#)
Channel Voice Messages.
- const [midibyte](#) [EVENT_NOTE_OFF](#)
- const [midibyte](#) [EVENT_NOTE_ON](#)
- const [midibyte](#) [EVENT_AFTERTOUCH](#)
- const [midibyte](#) [EVENT_CONTROL_CHANGE](#)
- const [midibyte](#) [EVENT_PROGRAM_CHANGE](#)
- const [midibyte](#) [EVENT_CHANNEL_PRESSURE](#)
- const [midibyte](#) [EVENT_PITCH_WHEEL](#)
- const [midibyte](#) [EVENT_CTRL_VOLUME](#)
Control Change Messages.
- const [midibyte](#) [EVENT_CTRL_BALANCE](#)
- const [midibyte](#) [EVENT_CTRL_PAN](#)
- const [midibyte](#) [EVENT_CTRL_EXPRESSION](#)
- const [midibyte](#) [EVENT_MIDI_REALTIME](#)
System Messages.
- const [midibyte](#) [EVENT_MIDI_SYSEX](#)
- const [midibyte](#) [EVENT_MIDI_QUARTER_FRAME](#)
- const [midibyte](#) [EVENT_MIDI_SONG_POS](#)
- const [midibyte](#) [EVENT_MIDI_SONG_SELECT](#)
- const [midibyte](#) [EVENT_MIDI_SONG_F4](#)
- const [midibyte](#) [EVENT_MIDI_SONG_F5](#)
- const [midibyte](#) [EVENT_MIDI_TUNE_SELECT](#)
- const [midibyte](#) [EVENT_MIDI_SYSEX_END](#)
- const [midibyte](#) [EVENT_MIDI_SYSEX_CONTINUE](#)
- const [midibyte](#) [EVENT_MIDI_CLOCK](#)
- const [midibyte](#) [EVENT_MIDI_SONG_F9](#)

- const [midibyte EVENT_MIDI_START](#)
- const [midibyte EVENT_MIDI_CONTINUE](#)
- const [midibyte EVENT_MIDI_STOP](#)
- const [midibyte EVENT_MIDI_SONG_FD](#)
- const [midibyte EVENT_MIDI_ACTIVE_SENSE](#)
- const [midibyte EVENT_MIDI_RESET](#)
- const [midibyte EVENT_MIDI_META](#)

0xFF is a MIDI "escape code" used in MIDI files to introduce a MIDI meta event.

- const [midibyte EVENT_META_SEQ_NUMBER](#)

Provides values for the currently-supported Meta events, and many others:

- const [midibyte EVENT_META_TEXT_EVENT](#)
- const [midibyte EVENT_META_COPYRIGHT](#)
- const [midibyte EVENT_META_TRACK_NAME](#)
- const [midibyte EVENT_META_INSTRUMENT](#)
- const [midibyte EVENT_META_LYRIC](#)
- const [midibyte EVENT_META_MARKER](#)
- const [midibyte EVENT_META_CUE_POINT](#)
- const [midibyte EVENT_META_MIDI_CHANNEL](#)
- const [midibyte EVENT_META_MIDI_PORT](#)
- const [midibyte EVENT_META_END_OF_TRACK](#)
- const [midibyte EVENT_META_SET_TEMPO](#)
- const [midibyte EVENT_META_SMPTE_OFFSET](#)
- const [midibyte EVENT_META_TIME_SIGNATURE](#)
- const [midibyte EVENT_META_KEY_SIGNATURE](#)
- const [midibyte EVENT_META_SEQSPEC](#)
- const [midibyte EVENT_META_ILLEGAL](#)

As a "type" (overloaded on channel) value for a Meta event, 0xFF indicates an illegal meta type.

- const [midibyte EVENT_NULL_CHANNEL](#)

This value of 0xFF is Sequencer64's channel value that indicates that the event's m_channel value is bogus.

- const [midibyte EVENT_GET_CHAN_MASK](#)

These file masks are used to obtain or to mask off the channel data from a status byte.

- const [midibyte EVENT_CLEAR_CHAN_MASK](#)
- const int [EVENTS_ALL](#)

Variable from the "stazed" extras.

- const int [EVENTS_UNSELECTED](#)
- const int [c_midibus_output_size](#)

Manifest global constants.

- const int [c_midibus_input_size](#)

The c_midibus_input_size value is passed, in mastermidibus, to snd_seq_set_input_buffer_size().

- const int [c_midibus_sysex_chunk](#)

Controls the amount a SysEx data sent at one time, in the midibus module.

- const [midilong c_midibus](#)

Provides tags used by the midifile class to control the reading and writing of the extra "proprietary" information stored in a Seq24 MIDI file.

- const [midilong c_midich](#)

Track channel number.

- const [midilong c_midiclocks](#)

Track clocking.

- const [midilong c_triggers](#)

See c_triggers_new.

- const [midilong c_notes](#)

Song data.

- const [midilong c_timesig](#)
Track time signature.
- const [midilong c_bpmtag](#)
Song beats/minute.
- const [midilong c_triggers_new](#)
Track trigger data.
- const [midilong c_mutegroups](#)
Song mute group data.
- const [midilong c_gap_A](#)
Gap.
- const [midilong c_gap_B](#)
Gap.
- const [midilong c_gap_C](#)
Gap.
- const [midilong c_gap_D](#)
Gap.
- const [midilong c_gap_E](#)
Gap.
- const [midilong c_gap_F](#)
Gap.
- const [midilong c_midictrl](#)
Song MIDI control.
- const [midilong c_musickey](#)
The track's key.
- const [midilong c_musicscale](#)
The track's scale.
- const [midilong c_backsequence](#)
Track background sequence.
- const [midilong c_transpose](#)
Track transpose value.
- const [midilong c_perf_bp_mes](#)
Perfedit beats/measure.
- const [midilong c_perf_bw](#)
Perfedit beat-width.
- const [midilong c_tempo_map](#)
Reserve seq32 tempo map.
- const [midilong c_reserved_1](#)
Reserved for expansion.
- const [midilong c_reserved_2](#)
Reserved for expansion.
- const [midilong c_tempo_track](#)
Alternate tempo track no.
- const [midilong c_seq_color](#)
Future feature Kepler34.
- const [midilong c_seq_edit_mode](#)
Future feature Kepler34.
- const int [c_midi_track_ctrl](#)
Pseudo control values for associating MIDI events (I think) with automation of some of the controls in seq24.
- const int [c_midi_control_bpm_up](#)
- const int [c_midi_control_bpm_dn](#)
- const int [c_midi_control_ss_up](#)

- const int [c_midi_control_ss_dn](#)
- const int [c_midi_control_mod_replace](#)
- const int [c_midi_control_mod_snapshot](#)
- const int [c_midi_control_mod_queue](#)
- const int [c_midi_control_mod_gmute](#)
- const int [c_midi_control_mod_glearn](#)
- const int [c_midi_control_play_ss](#)
- const int [c_midi_controls](#)
- const int [c_midi_control_playback](#)
- const int [c_midi_control_song_record](#)
- const int [c_midi_control_solo](#)
- const int [c_midi_control_thru](#)
- const int [c_midi_control_bpm_page_up](#)
- const int [c_midi_control_bpm_page_dn](#)
- const int [c_midi_control_ss_set](#)
- const int [c_midi_control_record](#)
- const int [c_midi_control_quan_record](#)
- const int [c_midi_control_reset_seq](#)
- const int [c_midi_controls_extended](#)
- int [g_midi_control_limit](#)
- const int [c_status_replace](#)

These were purely internal constants used with the functions that implement MIDI control (and also some keystroke control) for the application.

- const int [c_status_snapshot](#)

This value signals the "snapshot" functionality.

- const int [c_status_queue](#)

This value signals the "queue" functionality.

- const int [c_status_oneshot](#)

This value signals the Kepler34 "one-shot" functionality.

- const bool [c_scales_policy](#) [[c_scale_size](#)][SEQ64_OCTAVE_SIZE]

Each value in the kind of scale is denoted by a true value in these arrays.

- const int [c_scales_transpose_up](#) [[c_scale_size](#)][SEQ64_OCTAVE_SIZE]

Increment values needed to transpose each scale up so that it remains in the same key.

- const int [c_scales_transpose_dn](#) [[c_scale_size](#)][SEQ64_OCTAVE_SIZE]

Making these positive makes it easier to read, but the actual array contains negative values.

- const char [c_scales_text](#) [[c_scale_size](#)][20]

The names of the currently-supported scales.

- const char [c_key_text](#) [SEQ64_OCTAVE_SIZE][4]

Provides the entries for the Key dropdown menu in the Pattern Editor window.

- const char [c_interval_text](#) [16][4]

Provides the entries for the Interval dropdown menu in the Pattern Editor window.

- const char [c_chord_text](#) [8][6]

Provides the entries for the Chord dropdown menu in the Pattern Editor window.

- const int [c_chord_number](#)

Additional support data for the chord-generation feature from Stazed's seq32 project.

- const char [c_chord_table_text](#) [[c_chord_number](#)][12]

Additional support data for the chord-generation feature from Stazed's seq32 project.

- const int [c_chord_size](#)

Provides the number of chord values in each chord's specification array.

- const int [c_chord_table](#) [[c_chord_number](#)][[c_chord_size](#)]

Additional support data for the chord-generation feature from Stazed's seq32 project.

- const int [c_max_instruments](#)

- Provides the maximum number of instruments that can be defined in the `~/ .seq24usr` or `~/ .config/sequencer64/sequencer.rc` file.*
- `const int c_max_busses`
Provides the maximum number of MIDI buss definitions supported in the "user" file.
 - `static const std::string versiontext`
Sets up the "hardwired" version text for Sequencer64.
 - `static struct option long_options []`
A structure for command parsing that provides the long forms of command-line arguments, and associates them with their respective short form.
 - `static const std::string s_arg_list`
Provides a complete list of the short options, and is passed to `getopt_long()`.
 - `static const char *const s_help_1a`
Provides help text.
 - `static const char *const s_help_1b`
More help text.
 - `static const char *const s_help_2`
Still more help text.
 - `static const char *const s_help_3`
Still still more help text.
 - `static const char *const s_help_4`
Still still more more help text.
 - `static const char *const s_help_5`
Still still still more more more help text.
 - `static const std::string s_bitness`
This section of variables provide static information about the options enabled or disabled during the build.
 - `jack_status_pair_t s_status_pairs []`
Provides a list of JACK status bits, and a brief string to explain the status bit.
 - `struct charpair_t s_character_mapping []`
The array of mappings of the non-alphabetic characters.
 - `static lash * s_global_lash_driver`
The global pointer to the LASH driver instance.
 - `static rc_settings g_rc_settings`
Provides the replacement for all of the other "global_xxx" variables.
 - `static user_settings g_user_settings`
Provides the replacement for all of the other settings in the "user" configuration file, plus some of the "constants" in the `globals` module.
 - `static const int s_jitter_amount`
An internal variable for user-jitter control.
 - `static mainwid * gs_mainwid_pointer`
Holds a pointer to the single instance of `mainwnd` for the entire application, once it is created.
 - `static perfedit * gs_perfedit_pointer_0`
Holds a pointer to the first instance of `perfedit` for the entire application, once it is created.
 - `static perfedit * gs_perfedit_pointer_1`
Holds a pointer to the second instance of `perfedit` for the entire application, once it is created.
 - `static const long s_handlesize`
An internal variable for handle size.

9.2.1 Detailed Description

Do not document a namespace; it breaks Doxygen.

0.9.3 delta-tick calculation code. This code doesn't quite work for generating the proper rate of MIDI clocks, and so have disabled that code until we can figure out what it is we're doing wrong. Do not enable it unless you are willing to test it.

9.2.2 Typedef Documentation

9.2.2.1 midibyte

```
typedef unsigned char seq64::midibyte
```

This can be used for a MIDI buss/port number or for a MIDI channel number. See the SEQ64_INVALID_MIDIBYTE macro.

9.2.2.2 midistring

```
typedef std::basic_string<midibyte> seq64::midistring
```

9.2.2.3 bussbyte

```
typedef unsigned char seq64::bussbyte
```

9.2.2.4 midishort

```
typedef unsigned short seq64::midishort
```

9.2.2.5 midilong

```
typedef unsigned long seq64::midilong
```

9.2.2.6 colorbyte

```
typedef char seq64::colorbyte
```

This value is signed since we need a value of -1 to indicate no color, and 0 to 127 to indicate the index that "points" to a palette color.

9.2.2.7 midipulse

```
typedef long seq64::midipulse
```

HOWEVER, CURRENTLY, if you make this value unsigned, then perfroll won't show any notes in the sequence bars!!! Also, a number of manipulations of this type currently depend upon it being a signed value.

9.2.2.8 midibpm

```
typedef double seq64::midibpm
```

This value used to be an integer, but we need to provide more precision in order to support better tempo matching.

9.2.2.9 rterror_callback

```
typedef void(* seq64::rterror_callback) (rterror::Type type, const std::string &errmsg, void *userdata)
```

Note that class behaviour is undefined after a critical error (not a warning) is reported.

Parameters

<i>type</i>	Type of error.
<i>errorText</i>	Error description.

9.2.2.10 rtmidi_callback_t

```
typedef void(* seq64::rtmidi_callback_t) (midi_message &message, void *userdata)
```

Used to be nested in the [rtmidi_in](#) class. The timestamp parameter has been folded into the [midi_message](#) class (a wrapper for `std::vector<unsigned char>`), and the pointer has been replaced by a reference.

9.2.3 Enumeration Type Documentation

9.2.3.1 wave_type_t

```
enum seq64::wave_type_t
```

We still have to clarify these type values, though.

Enumerator

WAVE_NONE	No waveform, never used.
WAVE_SINE	Sine wave modulation.
WAVE_SAWTOOTH	Saw-tooth (ramp) modulation.
WAVE_REVERSE_SAWTOOTH	Reverse saw-tooth (decay).
WAVE_TRIANGLE	No waveform, never used.

9.2.3.2 seq_modifier_t

```
enum seq64::seq_modifier_t
```

We have to tweak the names to avoid redeclaration errors and to "personalize" the values. We change "GDK" to "SEQ64".

Since we're getting events from, say Gtk-2.4, but using our (matching) values for comparison, use the `CAST_EQ↔UIVALENT()` macro to compare them. Note that we might still end up having to a remapping (e.g. if trying to get the code to work with the Qt framework).

Enumerator

SEQ64_NO_MASK	
SEQ64_SHIFT_MASK	
SEQ64_LOCK_MASK	
SEQ64_CONTROL_MASK	
SEQ64_MOD1_MASK	
SEQ64_MOD2_MASK	
SEQ64_MOD3_MASK	
SEQ64_MOD4_MASK	
SEQ64_MOD5_MASK	
SEQ64_BUTTON1_MASK	
SEQ64_BUTTON2_MASK	
SEQ64_BUTTON3_MASK	
SEQ64_BUTTON4_MASK	
SEQ64_BUTTON5_MASK	
SEQ64_SUPER_MASK	Bits 13 and 14 are used by XKB, bits 15 to 25 are unused. Bit 29 is used internally.
SEQ64_HYPER_MASK	
SEQ64_META_MASK	
SEQ64_RELEASE_MASK	
SEQ64_MASK_MAX	

9.2.3.3 seq_event_type_t

```
enum seq64::seq_event_type_t
```

Only the values we need have been grabbed. We have to tweak the names to avoid redeclaration errors and to "personalize" the values. We change "GDK" to "SEQ64", but, for convenience (to hide errors? :-D), we keep the number the same.

Since we're getting events from, say Gtk-2.4, but using our (matching) values for comparison, use the `CAST_EQ↔UIVALENT()` macro to compare them. Note that we might still end up having to do a remapping (e.g. if trying to get the code to work with the Qt framework).

Enumerator

SEQ64_NOTHING	
SEQ64_DELETE	
SEQ64_DESTROY	
SEQ64_EXPOSE	
SEQ64_MOTION_NOTIFY	
SEQ64_BUTTON_PRESS	
SEQ64_2BUTTON_PRESS	
SEQ64_3BUTTON_PRESS	
SEQ64_BUTTON_RELEASE	
SEQ64_KEY_PRESS	
SEQ64_KEY_RELEASE	
SEQ64_SCROLL	
SEQ64_EVENT_LAST	

9.2.3.4 seq_scroll_direction_t

```
enum seq64::seq_scroll_direction_t
```

We have to tweak the names to avoid redeclaration errors and to "personalize" the values. We change "SEQ64" to "SEQ64".

Since we're getting events from, say Gtk-2.4, but using our (matching) values for comparison, use the `CAST_EQUIVALENT()` macro to compare them. Note that we might still end up having to a remapping (e.g. if trying to get the code to work with the Qt framework).

Enumerator

SEQ64_SCROLL_UP	
SEQ64_SCROLL_DOWN	
SEQ64_SCROLL_LEFT	
SEQ64_SCROLL_RIGHT	

9.2.3.5 clock_e

```
enum seq64::clock_e
```

This enumeration was also defined in `midibus_portmidi.h`, but we put it into this common module to avoid duplication.

Enumerator

e_clock_disabled	A new, currently-hidden value to indicate to ignore/disable an output port. If a port always fails to open, we want to just ignore it.
------------------	--

Enumerator

e_clock_off	Corresponds to the "Off" selection in the MIDI Clock tab. With this setting, the MIDI Clock is disabled for the buss using this setting. Notes will still be sent that buss, of course. Some software synthesizers might require this setting in order to make a sound.
e_clock_pos	Corresponds to the "Pos" selection in the MIDI Clock tab. With this setting, MIDI Clock will be sent to this buss, and, if playback is starting beyond tick 0, then MIDI Song Position and MIDI Continue will also be sent on this buss.
e_clock_mod	Corresponds to the "Mod" selection in the MIDI Clock tab. With this setting, MIDI Clock and MIDI Start will be sent. But clocking won't begin until the Song Position has reached the start modulo (in 1/16th notes) that is specified.

9.2.3.6 interaction_method_t

```
enum seq64::interaction_method_t
```

Moved here from the globals.h module.

Enumerator

e_seq24_interaction	Use the normal mouse interactions.
e_fruity_interaction	The "fruity" mouse interactions.
e_number_of_interactions	Keep this last... a size value.

9.2.3.7 mute_group_handling_t

```
enum seq64::mute_group_handling_t
```

There's no GUI way to set this item yet.

e_mute_group_stomp: This is the legacy (seq24) option, which reads the mute-groups from the MIDI file, and saves them back to the "rc" file and to the MIDI file. However, for Sequencer64 MIDI files such as b4uacuse-stress.midi, seq24 never reads the mute-groups in that MIDI file! In any case, this can be considered a corruption of the "rc" file.

e_mute_group_preserve: In this option, the mute groups are only written to the "rc" file if the MIDI file did not contain non-zero mute groups. This option prevents the contamination of the "rc" mute-groups by the MIDI file's mute-groups. We're going to make this the default option.

Enumerator

e_mute_group_stomp	Save main group to "rc"/MIDI files.
e_mute_group_preserve	Write new groups only to MIDI file.
e_mute_group_max	Keep this last... a size value.

9.2.3.8 c_music_scales

```
enum seq64::c_music_scales
```

Scales can be shown in the piano roll as gray bars for reference purposes.

We've added three more scales; there are still a number of them that could be fruitfully added to the list of scales.

It would be good to offload this stuff into a new "scale" class.

Enumerator

c_scale_off	
c_scale_major	
c_scale_minor	
c_scale_harmonic_minor	
c_scale_melodic_minor	
c_scale_c_whole_tone	
c_scale_blues	
c_scale_major_pentatonic	
c_scale_minor_pentatonic	
c_scale_size	

9.2.3.9 draw_type_t

```
enum seq64::draw_type_t
```

These values are used in the sequence, seqroll, perffroll, and mainwid classes.

Enumerator

DRAW_FIN	Indicates that drawing is finished.
DRAW_NORMAL_LINKED	Used for drawing linked notes.
DRAW_NOTE_ON	For starting the drawing of a note.
DRAW_NOTE_OFF	For finishing the drawing of a note.
DRAW_TEMPO	For drawing tempo meta events.

9.2.3.10 edit_mode_t

```
enum seq64::edit_mode_t
```

A feature adapted from Kepler34. Not yet ready for prime time.

Enumerator

EDIT_MODE_NOTE	Edit as Note input, the normal edit mode.
EDIT_MODE_DRUM	Edit as Drum input, using short notes.

9.2.3.11 loop_record_t

```
enum seq64::loop_record_t
```

These values are used by the seqedit class, which provides a button with a popup menu to select one of these recording modes.

Enumerator

LOOP_RECORD_LEGACY	Incoming events are merged into the loop.
LOOP_RECORD_OVERWRITE	Incoming events overwrite the loop.
LOOP_RECORD_EXPAND	Incoming events increase size of loop.

9.2.3.12 mouse_action_e

```
enum seq64::mouse_action_e
```

Enumerator

e_action_select	
e_action_draw	
e_action_grow	

9.2.3.13 edit_action_t

```
enum seq64::edit_action_t
```

These variables represent actions that can be applied to a selection of notes. One idea would be to add a swing-quantize action. We will reserve the value here, for notes only; not yet used or part of the action menu.

Enumerator

c_select_all_notes	
c_select_all_events	
c_select_inverse_notes	
c_select_inverse_events	

Enumerator

c_quantize_notes	
c_quantize_events	
c_tighten_events	
c_tighten_notes	
c_transpose_notes	
c_reserved	
c_transpose_h	
c_expand_pattern	
c_compress_pattern	
c_select_even_notes	
c_select_odd_notes	
c_swing_notes	

9.2.3.14 rtmidi_api

```
enum seq64::rtmidi_api
```

These items used to be nested in the rtmidi class, but that only worked when RtMidi.cpp/h were large monolithic modules.

Enumerator

RTMIDI_API_UNSPECIFIED	Search for a working compiled API.
RTMIDI_API_LINUX_ALSA	Advanced Linux Sound Architecture API.
RTMIDI_API_UNIX_JACK	JACK Low-Latency MIDI Server API.
RTMIDI_API_MAXIMUM	A count of APIs; an erroneous value.

9.2.4 Function Documentation

9.2.4.1 swap()

```
void seq64::swap (
    busarray & buses0,
    busarray & buses1 )
```

Parameters

<i>buses0</i>	Provides the first buss in the swap.
<i>buses1</i>	Provides the second buss in the swap.

9.2.4.2 wave_type_name()

```
std::string seq64::wave_type_name (
    wave_type_t wavetype )
```

These names are short because I cannot figure out how to get the window pad out to show the longer names.

Parameters

<i>wavetype</i>	The wave-type value to be displayed.
-----------------	--------------------------------------

Returns

Returns a short description of the wave type.

9.2.4.3 extract_timing_numbers()

```
int seq64::extract_timing_numbers (
    const std::string & s,
    std::string & part_1,
    std::string & part_2,
    std::string & part_3,
    std::string & fraction )
```

- measures : beats : divisions
 - "8" represents solely the number of pulses. That is, if the user enters a single number, it is treated as the number of pulses.
 - "8:1" represents a measure and a beat.
 - "213:4:920" represents a measure, a beat, and pulses.
- hours : minutes : seconds . fraction. We really don't support this concept at present. Beware!
 - "2:04:12.14"
 - "0:1:2"

Warning

This is not the most efficient implementation you'll ever see. At some point we will tighten it up. This function is tested in the seq64-tests project, in the "calculations_unit_test" module. At present this test is surely BROKEN!

Parameters

	<i>s</i>	Provides the input time string, in measures or time format, to be processed.
out	<i>part_1</i>	The destination reference for the first part of the time. In some contexts, this number alone is a pulse (ticks) value; in other contexts, it is a measures value.
out	<i>part_2</i>	The destination reference for the second part of the time.
out	<i>part_3</i>	The destination reference for the third part of the time.
out	<i>fraction</i>	The destination reference for the fractional part of the time.

Returns

Returns the number of parts provided, ranging from 0 to 4.

9.2.4.4 tokenize_string()

```
int seq64::tokenize_string (
    const std::string & source,
    std::vector< std::string > & tokens )
```

They are treated equally, and the caller must determine what to do with the parts. Here are the steps:

```
-# Skip any delimiters found at the beginning. The position will
   either exist, or there will be nothing to parse.
-# Get to the next delimiter. This will exist, or not. Get all
   non-delimiters until the next delimiter or the end of the string.
-# Repeat until no more delimiters exist.
```

Parameters

<i>source</i>	The string to be parsed and tokenized.
<i>tokens</i>	Provides a vector into which to push the tokens.

Returns

Returns the number of tokens pushed (i.e. the final size of the tokens vector).

9.2.4.5 pulses_to_string()

```
std::string seq64::pulses_to_string (
    midipulse p )
```

Todo Still needs to be unit tested.

Parameters

<i>p</i>	The MIDI pulse/tick value to be converted.
----------	--

Returns

Returns the string as an unsigned ASCII integer number.

9.2.4.6 pulses_to_measurestring()

```
std::string seq64::pulses_to_measurestring (
    midipulse p,
    const midi_timing & seqparms )
```

Parameters

<i>p</i>	The number of MIDI pulses (clocks, divisions, ticks, you name it) to be converted. If the value is SEQ64_NULL_MIDIPULSE, it is converted to 0, because callers don't generally worry about such niceties, and the least we can do is convert illegal measure-strings (like "000:0:000") to a legal value.
<i>seqparms</i>	This small structure provides the beats/measure, beat-width, and PPQN that hold for the sequence involved in this calculation. These values are needed in the calculations.

Returns

Returns the string, in measures notation, for the absolute pulses that mark this duration.

9.2.4.7 pulses_to_midi_measures()

```
bool seq64::pulses_to_midi_measures (
    midipulse p,
    const midi_timing & seqparms,
    midi_measures & measures )
```

$$m = p * W / (4 * P * B)$$

Parameters

	<i>p</i>	Provides the MIDI pulses (as in "pulses per quarter note") that are to be converted to MIDI measures format.
	<i>seqparms</i>	This small structure provides the beats/measure (B), beat-width (W), and PPQN (P) that hold for the sequence involved in this calculation. The beats/minute (T for tempo) value is not needed.
out	<i>measures</i>	Provides the current MIDI song time structure to hold the results, which are the measures, beats, and divisions values for the time of interest. Note that the measures and beats are corrected to be re 1, not 0.

Returns

Returns true if the calculations were able to be made. The P, B, and W values all need to be greater than 0.

9.2.4.8 `pulses_to_timestring()` [1/2]

```
std::string seq64::pulses_to_timestring (
    midipulse p,
    const midi_timing & timinginfo )
```

See the other `pulses_to_timestring()` overload.

Todo Still needs to be unit tested.

Parameters

<i>p</i>	Provides the number of ticks, pulses, or divisions in the MIDI event time.
<i>timinginfo</i>	Provides the tempo of the song, in beats/minute, and the pulse-per-quarter-note of the song.

Returns

Returns the return-value of the other `pulses_to_timestring()` function.

9.2.4.9 `pulses_to_timestring()` [2/2]

```
std::string seq64::pulses_to_timestring (
    midipulse p,
    midibpm bpm,
    int ppqn,
    bool showus )
```

If the fraction part is 0, then it is not shown. Examples:

```
- "0:0:0"
- "0:0:0.102333"
- "12:3:1"
- "12:3:1.000001"
```

Parameters

<i>p</i>	Provides the number of ticks, pulses, or divisions in the MIDI event time.
<i>bpm</i>	Provides the tempo of the song, in beats/minute.
<i>ppqn</i>	Provides the pulses-per-quarter-note of the song.
<i>showus</i>	If true (the default), shows the microseconds as well.

Returns

Returns the time-string representation of the pulse (ticks) value.

9.2.4.10 measurestring_to_pulses()

```
midipulse seq64::measurestring_to_pulses (
    const std::string & measures,
    const midi_timing & seqparms )
```

If the third value (the MIDI pulses or ticks value) is set to the dollar sign ("\$"), then the pulses are set to PPQN-1, as a handy shortcut to indicate the end of the beat.

Warning

If only one number is provided, it is treated in this function like a measures value, not a pulses value.

Parameters

<i>measures</i>	Provides the current MIDI song time in "measures:beats:divisions" format, where divisions are the MIDI pulses in "pulses-per-quarter-note".
<i>seqparms</i>	This small structure provides the beats/measure, beat-width, and PPQN that hold for the sequence involved in this calculation.

Returns

Returns the absolute pulses that mark this duration. If the input string is empty, then 0 is returned.

9.2.4.11 midi_measures_to_pulses()

```
midipulse seq64::midi_measures_to_pulses (
    const midi_measures & measures,
    const midi_timing & seqparms )
```

$p = 4 * P * m * B / W$ p == pulse count (ticks or pulses) m == number of measures B == beats per measure (constant) P == pulses per quarter-note (constant) W == beat width in beats per measure (constant)

Note that the 0-pulse MIDI measure is "1:1:0", which means "at the beginning of the first beat of the first measure, no pulses". It is not "0:0:0" as one might expect. If we get a 0 for measures or for beats, we treat them as if they were 1. It is too easy for the user to mess up.

We should consider clamping the beats to the beat-width value as well.

Parameters

<i>measures</i>	Provides the current MIDI song time structure holding the measures, beats, and divisions values for the time of interest.
<i>seqparms</i>	This small structure provides the beats/measure, beat-width, and PPQN that hold for the sequence involved in this calculation.

Returns

Returns the absolute pulses that mark this duration. If the pulse-value cannot be calculated, then SEQ64_↔ NULL_MIDIPULSE is returned.

9.2.4.12 timestring_to_pulses() [1/2]

```
midipulse seq64::timestring_to_pulses (
    const std::string & timestring,
    int bpm,
    int ppqn )
```

9.2.4.13 string_to_pulses()

```
midipulse seq64::string_to_pulses (
    const std::string & s,
    const midi_timing & mt )
```

First, the type of string is deduced by the characters in the string. If the string contains two colons and a decimal point, it is assumed to be a time-string ("hh:mm:ss.frac"); in addition ss will have to be less than 60. ??? Actually, now we only care if four numbers are provided.

If the string just contains two colons, then it is assumed to be a measure-string ("measures:beats:divisions").

If it has none of the above, it is assumed to be pulses. Testing is not rigorous.

Parameters

<i>s</i>	Provides the string to convert to pulses.
<i>mt</i>	Provides the structure needed to provide BPM and other values needed for some of the conversions done by this function.

Returns

Returns the string as converted to MIDI pulses (or divisions, clocks, ticks, whatever you call it).

9.2.4.14 string_to_midibyte()

```
midibyte seq64::string_to_midibyte (
    const std::string & s )
```

This function bypasses characters until it finds a digit (whether part of the number or a "0x" construct), and then converts it.

Parameters

<i>s</i>	Provides the string to convert to a MIDI byte.
----------	--

Returns

Returns the MIDI byte value represented by the string.

9.2.4.15 shorten_file_spec()

```
std::string seq64::shorten_file_spec (
    const std::string & fpath,
    int leng )
```

This is done by removing character in the middle, if necessary, and replacing them with an ellipse.

This function operates by first trying to find the `/home` directory. If found, it strips off `/home/username` and replace it with the Linux `~` replacement for the `$HOME` environment variable. This function assumes that the "username" portion *must* exist, and that there's no goofy stuff like double-slashes in the path.

Parameters

<i>fpath</i>	The file specification, including the full path to the file, and the name of the file.
<i>leng</i>	Provides the length to which to limit the string.

Returns

Returns the *fpath* parameter, possibly shortened to fit within the desired length.

9.2.4.16 string_not_void()

```
bool seq64::string_not_void (
    const std::string & s )
```

Provides essentially the opposite test that [string_is_void\(\)](#) provides. The definition of white-space is provided by the `std::isspace()` function/macro.

Parameters

<i>s</i>	The string pointer to check for emptiness.
----------	--

Returns

Returns true if the pointer is valid, the string has a non-zero length, and is not just white-space.

9.2.4.17 string_is_void()

```
bool seq64::string_is_void (
    const std::string & s )
```

Meant to have essentially the opposite result of [string_not_void\(\)](#). The meaning of empty is special here, as it refers to a string being useless as a token:

- The string is of zero length.
- The string has only white-space characters in it, where the `isspace()` macro provides the definition of white-space.

Parameters

<i>s</i>	The string pointer to check for emptiness.
----------	--

Returns

Returns true if the string has a zero length, or is only white-space.

9.2.4.18 strings_match()

```
bool seq64::strings_match (
    const std::string & target,
    const std::string & x )
```

The [strings_match\(\)](#) function returns true if the comparison items are identical, without case-sensitivity in character content up to the length of the secondary string. This allows abbreviations to match. (And, in scanning routines, the first match is immediately accepted.)

Parameters

<i>target</i>	The primary string in the comparison. This is the target string, the one we hope to match. It is <i>assumed</i> to be non-empty, and the result is false if it is empty..
<i>x</i>	The secondary string in the comparison. It must be no longer than the target string, or the match is false.

Returns

Returns true if both strings are identical in characters, up to the length of the secondary string, with the case of the characters being insignificant. Otherwise, false is returned.

9.2.4.19 log2_time_sig_value()

```
int seq64::log2_time_sig_value (
    int tsd )
```

Useful in converting a time signature's denominator to a Time Signature meta event's "dd" value.

Parameters

<i>tsd</i>	The time signature denominator, which must be a power of 2: 2, 4, 8, 16, or 32.
------------	---

Returns

Returns the power of 2 that achieves the *tsd* parameter value.

9.2.4.20 zoom_power_of_2()

```
int seq64::zoom_power_of_2 (
    int ppqn )
```

The default starting zoom is 2, but this value is suitable only for PPQN of 192 and below. Also, zoom currently works consistently only if it is a power of 2. For starters, we scale the zoom to the selected ppqn, and then shift it each way to get a suitable power of two.

Parameters

<i>ppqn</i>	The ppqn of interest.
-------------	-----------------------

Returns

Returns the power of 2 appropriate for the given PPQN value.

9.2.4.21 beat_pow2()

```
int seq64::beat_pow2 (
    int logbase2 )
```

Use for calculating the denominator of a time signature.

Parameters

<i>logbase2</i>	Provides the power to which 2 is to be raised. This integer is probably only rarely greater than 4 (which represents a denominator of 16).
-----------------	--

Returns

Returns 2 raised to the logbase2 power.

9.2.4.22 beat_log2()

```
midibyte seq64::beat_log2 (
    int value )
```

This number is truncated to an integer byte value, as it is used in calculating values to be written to a MIDI file.

Parameters

<i>value</i>	The integer value for which log2(value) is needed.
--------------	--

Returns

Returns log2(value).

9.2.4.23 tempo_us_from_bytes()

```
double seq64::tempo_us_from_bytes (
    const midibyte tt[3] )
```

Is it correct to simply cast the bytes to a double value?

Parameters

<i>tt</i>	Provides the 3-byte array of values making up the raw tempo data.
-----------	---

Returns

Returns the result of converting the bytes to a double value.

9.2.4.24 tempo_us_to_bytes()

```
void seq64::tempo_us_to_bytes (
    midibyte t[3],
    int tempo_us )
```

Recall the format of a Tempo event:

0 FF 51 03 t2 t1 t0 (tempo as number of microseconds per quarter note)

This code is the inverse of the lines of code around line 768 in midifile.cpp, which is basically $((t_2 * 256) + t_1) * 256 + t_0$.

As a test case, note that the default tempo is 120 beats/minute, which is equivalent to $t_{total}=500000$ (0x07A120). The output of this function will be $t[] = \{ 0x07, 0xa1, 0x20 \}$ [the indices go 0, 1, 2].

Parameters

<i>t</i>	Provides a small array of 3 elements to hold each tempo byte.
<i>tempo_us</i>	Provides the temp value in microseconds per quarter note. This is always an integer, not a double, so do not get confused here.

9.2.4.25 tempo_to_note_value()

```
midibyte seq64::tempo_to_note_value (
    midibpm tempovalue )
```

It implements the following linear equation, with clamping just in case.

$$N = N_0 + (B - B_0) \frac{N_1 - N_0}{B_1 - B_0} \quad \text{where } (N_1 - N_0) \text{ is always } 127$$

$$N = (B - B_0) \frac{127}{B_1 - B_0}$$

where $N_0 = 0$ (MIDI note 0 is the minimum), $N_1 = 127$ (the maximum MIDI note), B_0 is the value of `usr().midi_bpm_minimum()`, B_1 is the value of `usr().midi_bpm_maximum()`, B is the input beats/minute, and N is the resulting note value. As a precaution due to rounding error, we clamp the values between 0 and 127.

Parameters

<i>tempovalue</i>	The tempo in beats/minute.
-------------------	----------------------------

Returns

Returns the tempo value scaled to the range 0 to 127, based on the configured BPM minimum and maximum values.

9.2.4.26 note_value_to_tempo()

```
midibpm seq64::note_value_to_tempo (
    midibyte note )
```

$$B = B0 + \frac{(N - N0) (B1 - B0)}{N1 - N0}$$

$$B = B0 + N \frac{(B1 - B0)}{127}$$

Parameters

<i>note</i>	The note value used for displaying the tempo in the seqdata pane, the perfroll, and in a mainwid slot.
-------------	--

Returns

Returns the tempo in beats/minute.

9.2.4.27 ppqn_is_valid()

```
bool seq64::ppqn_is_valid (
    int ppqn ) [inline]
```

Validates a PPQN value.

Parameters

<i>ppqn</i>	Provides the PPQN value to be used.
-------------	-------------------------------------

Returns

Returns true if the ppqn parameter is between MINIMUM_PPQN and MAXIMUM_PPQN, or is set to SE↔Q64_USE_DEFAULT_PPQN (-1).

9.2.4.28 tempo_us_from_bpm()

```
double seq64::tempo_us_from_bpm (
    midibpm bpm ) [inline]
```

120 beats/minute) to microseconds. This function is the inverse of [bpm_from_tempo_us\(\)](#).

Parameters

<i>bpm</i>	The value of beats-per-minute. If this value is 0, we'll get an arithmetic exception.
------------	---

Returns

Returns the tempo in qn/us. If the bpm value is 0, then 0 is returned.

9.2.4.29 bpm_from_tempo_us()

```
midibpm seq64::bpm_from_tempo_us (
    double tempous ) [inline]
```

The tempo event's numeric value is given in 3 bytes, and is in units of microseconds-per-quarter-note (us/qn).

Parameters

<i>tempous</i>	The value of the Tempo meta-event, in units of us/qn. If this value is 0, we'll get an arithmetic exception.
----------------	--

Returns

Returns the beats per minute. If the tempo value is 0, then 0 is returned.

9.2.4.30 bpm_from_bytes()

```
midibpm seq64::bpm_from_bytes (
    midibyte t[3] ) [inline]
```

It might be worthwhile to provide an std::vector version at some point.

Parameters

<i>t</i>	The 3 tempo midibytes that were read directly from a MIDI file.
----------	---

9.2.4.31 pulse_length_us()

```
double seq64::pulse_length_us (
    midibpm bpm,
    int ppqn ) [inline]
```

The formula for the pulse-length in seconds is:

$$P = \frac{60}{\text{BPM} * \text{PPQN}}$$

Parameters

<i>bpm</i>	Provides the beats-per-minute value. No sanity check is made. If this value is 0, we'll get an arithmetic exception.
<i>ppqn</i>	Provides the pulses-per-quarter-note value. No sanity check is made. If this value is 0, we'll get an arithmetic exception.

Returns

Returns the pulse length in microseconds. If either parameter is invalid, then this function will crash. :-D

9.2.4.32 `delta_time_us_to_ticks()`

```
double seq64::delta_time_us_to_ticks (
    unsigned long us,
    midibpm bpm,
    int ppqn ) [inline]
```

This function is the inverse of [ticks_to_delta_time_us\(\)](#).

Please note that terms "ticks" and "pulses" are equivalent, and refer to the "pulses" in "pulses per quarter note".

$$P = 120 \frac{\text{beats}}{\text{minute}} * 192 \frac{\text{pulses}}{\text{beats}} * T \text{ us} * \frac{1 \text{ minute}}{60 \text{ sec}} * \frac{1 \text{ sec}}{1,000,000 \text{ us}}$$

Note that this formula assumes that a beat is a quarter note. If a beat is an eighth note, then the P value would be halved, because there would be only 96 pulses per beat. We will implement an additional function to account for the beat; the current function merely blesses some calculations made in the application.

Parameters

<i>us</i>	The number of microseconds in the delta time.
<i>bpm</i>	Provides the beats-per-minute value, otherwise known as the "tempo".
<i>ppqn</i>	Provides the pulses-per-quarter-note value, otherwise known as the "division".

Returns

Returns the tick value.

9.2.4.33 `ticks_to_delta_time_us()`

```
double seq64::ticks_to_delta_time_us (
    midipulse delta_ticks,
```

```
midibpm bpm,
int ppqn ) [inline]
```

The inverse of [delta_time_us_to_ticks\(\)](#).

Please note that terms "ticks" and "pulses" are equivalent, and refer to the "pulses" in "pulses per quarter note".

Old: $60000000.0 * \text{double}(\text{delta_ticks}) / (\text{double}(\text{bpm}) * \text{double}(\text{ppqn}))$;

Parameters

<i>delta_ticks</i>	The number of ticks or "clocks".
<i>bpm</i>	Provides the beats-per-minute value, otherwise known as the "tempo".
<i>ppqn</i>	Provides the pulses-per-quarter-note value, otherwise known as the "division".

Returns

Returns the time value in microseconds.

9.2.4.34 clock_tick_duration_bogus()

```
double seq64::clock_tick_duration_bogus (
    midibpm bpm,
    int ppqn ) [inline]
```

Deprecated This is a somewhat bogus calculation used only for "statistical" output in the old perform module. Name changed to reflect this unfortunate fact. Use [pulse_length_us\(\)](#) instead.

$$\text{us} = \frac{60000000 \text{ ppqn}}{\text{MIDI_CLOCK_IN_PPQN} * \text{bpm} * \text{ppqn}}$$

MIDI_CLOCK_IN_PPQN is 24.

Parameters

<i>bpm</i>	Provides the beats-per-minute value. No sanity check is made. If this value is 0, we'll get an arithmetic exception.
<i>ppqn</i>	Provides the pulses-per-quarter-note value. No sanity check is made. If this value is 0, we'll get an arithmetic exception.

Returns

Returns the clock tick duration in microseconds. If either parameter is invalid, this will crash. Who wants to waste time on value checks here? :-D

9.2.4.35 clock_ticks_from_ppqn()

```
int seq64::clock_ticks_from_ppqn (
    int ppqn ) [inline]
```

Parameters

<i>ppqn</i>	The number of pulses per quarter note. For example, the default value for Seq24 is 192.
-------------	---

Returns

The integer value of $ppqn / 24$ [MIDI_CLOCK_IN_PPQN] is returned.

9.2.4.36 double_ticks_from_ppqn()

```
double seq64::double_ticks_from_ppqn (
    int ppqn ) [inline]
```

The same as [clock_ticks_from_ppqn\(\)](#), but returned as a double float.

Parameters

<i>ppqn</i>	The number of pulses per quarter note.
-------------	--

Returns

The double value of $ppqn / 24$ [SEQ64_MIDI_CLOCK_IN_PPQN] is returned.

9.2.4.37 pulses_per_measure()

```
midipulse seq64::pulses_per_measure (
    int ppqn = SEQ64_DEFAULT_PPQN ) [inline]
```

This calculation is extremely simple, and it provides an important constraint to pulse (ticks) calculations: the number of pulses in a measure is always 4 times the PPQN value, regardless of the time signature. The number pulses in a 7/8 measure is the the same as in a 4/4 measure.

9.2.4.38 measures_to_ticks()

```
midipulse seq64::measures_to_ticks (
    int bpb,
    int ppqn,
    int bw,
    int measures = 1 ) [inline]
```

This function is called in `seqedit::apply_length()`, when the user selects a sequence length in measures. That function calculates the length in ticks. The number of pulses is given by the number of quarter notes times the pulses per quarter note. The number of quarter notes is given by the measures times the quarter notes per measure. The quarter notes per measure is given by the beats per measure times 4 divided by beat_width beats. So:

```
p = 4 * P * m * B / W
p == pulse count (ticks or pulses)
m == number of measures
B == beats per measure (constant)
P == pulses per quarter-note (constant)
W == beat width in beats per measure (constant)
```

For our "b4uacuse" MIDI file, M can be about 100 measures, B is 4, P can be 192 (but we want to support higher values), and W is 4. So $p = 100 * 4 * 4 * 192 / 4 = 76800$ ticks.

Note that $4 * P$ is a constraint encapsulated by the inline function `pulses_per_measure()`.

Parameters

<i>bpb</i>	The B value in the equation, beats/measure or beats/bar.
<i>ppqn</i>	The P value in the equation, pulses/qn.
<i>bw</i>	The W value in the equation, the denominator of the time signature. If this value is 0, we'll get an arithmetic exception (crash), so we just return 0 in this case.
<i>measures</i>	The M value in the equation. It defaults to 1, in case one desires a simple "ticks per measure" number.

Returns

Returns the L value (ticks or pulses) as calculated via the given equation. If bw is 0, then 0 is returned.

9.2.4.39 ticks_to_measures()

```
int seq64::ticks_to_measures (
    int bpb,
    int ppqn,
    int bw,
    midipulse ticks = 192 ) [inline]
```

Parameters

<i>bpb</i>	The B value in the equation, beats/measure or beats/bar.
<i>ppqn</i>	The P value in the equation, pulses/qn.
<i>bw</i>	The W value in the equation, the denominator of the time signature. If this value is 0, we'll get an arithmetic exception (crash), so we just return 0 in this case.
<i>ticks</i>	The p (pulses) value in the equation. It defaults to 192, in case one desires a default "ticks per measure" number.

Returns

Returns the M value (measures or bars) as calculated via the inverse equation. If ppqn or bpb are 0, then 0 is returned.

9.2.4.40 pulse_divide()

```
midipulse seq64::pulse_divide (
    midipulse numerator,
    midipulse denominator,
    midipulse & remainder )
```

This function also avoids division by zero (and currently ignores negative denominators, which are still possible with the current definition of the midipulse typedef.

Parameters

	<i>numerator</i>	Provides the numerator in the division operation.
	<i>denominator</i>	Provides the denominator in the division operation.
out	<i>remainder</i>	The remainder is written here. If the division cannot be done, it is set to 0.

Returns

Returns the result of the division.

9.2.4.41 wave_func()

```
double seq64::wave_func (
    double angle,
    wave_type_t wavetype )
```

We extracted this function from mattias's lfound module, as it is more generally useful. The angle parameter is provided by the lfound object. It is calculated by

$$\text{angle} = \frac{\text{speed} * \text{tick} * \text{BW}}{\text{seqlength}} + \text{phase}$$

The speed ranges from 0 to 16; the ratio of tick/seqlength ranges from 0 to 1; BW (beat width) is generally 4; the phase ranges from 0 to 1.

Parameters

<i>angle</i>	Provides the radial angle to be applied. Units of radians, apparently.
<i>wavetype</i>	Provides the wave_type_t value to select the type of wave data-point to be generated.

9.2.4.42 `extract_port_names()`

```
bool seq64::extract_port_names (
    const std::string & fullname,
    std::string & clientname,
    std::string & portname )
```

It's a bit kruffy to have to rely on that strict format; changes in the bus/port code could break this function.

And when a2jmidid is running, indeed this function breaks. The name of a port changes to

```
a2j:Midi Through [14] (playback): Midi Through Port-0
```

with "a2j" as the client name and the rest, including the second colon, as the port name.

TODO: FIX ME!!!!!!

Parameters

	<i>fullname</i>	The full port specification to be split.
out	<i>clientname</i>	The destination for the client name portion, "clientname".
out	<i>portname</i>	The destination for the port name portion, "portname".

Returns

Returns true if all items are non-empty after the process.

9.2.4.43 `extract_bus_name()`

```
std::string seq64::extract_bus_name (
    const std::string & fullname )
```

Sometimes we don't need both parts at once.

However, when a2jmidid is active. the port name will have a colon in it.

Parameters

<i>fullname</i>	The "bus:port" name.
-----------------	----------------------

Returns

Returns the "bus" portion of the string. If there is no colon, then it is assumed there is no buss name, so an empty string is returned.

9.2.4.44 `extract_port_name()`

```
std::string seq64::extract_port_name (
    const std::string & fullname )
```

Sometimes we don't need both parts at once.

However, when a2jmidid is active. the port name will have a colon in it.

Parameters

<i>fullname</i>	The "bus:port" name.
-----------------	----------------------

Returns

Returns the "port" portion of the string. If there is no colon, then it is assumed that the name is a port name, and so *fullname* is returned.

9.2.4.45 `current_date_time()`

```
std::string seq64::current_date_time ( )
```

Returns

Returns

9.2.4.46 `help_check()`

```
bool seq64::help_check (
    int argc,
    char * argv[] )
```

Also check for the `--legacy` option. Finally, it also checks for the `"?"` option that people sometimes use as a guess to get help.

Parameters

<i>argc</i>	The number of command-line arguments.
<i>argv</i>	The array of command-line argument pointers.

Returns

Returns true only if `-v`, `-V`, `--version`, `-#`, `-h`, `--help`, or `"?"` were encountered. If the legacy options occurred, then `rc().legacy_format(true)` is called, as a side effect, because it will be needed before we parse the options.

9.2.4.47 parse_options_files()

```
bool seq64::parse_options_files (
    perform & p,
    std::string & errmsg,
    int argc,
    char * argv[] )
```

It probably requires this call preceding: `Gtk::Main kit(argc, argv)`, to strip any GTK+-specific parameters the knowledgeable user may have added. Usage:

```
Gtk::Main kit(argc, argv);
seq64::gui_assistant_gtk2 gui;
seq64::perform p(gui);
```

It also requires the caller to call `rc().set_defaults()` and `usr().set_defaults()` at the appropriate time, which is before any parsing of the command-line options. The caller can then use the command-line to make any modifications to the setting that will be used here. The biggest example is the `-r/--reveal-alsa-ports` option, which determines if the MIDI buss definition strings are read from the 'user' configuration file.

Instead of the legacy Seq24 names, we use the new configuration file-names, located in the `~/config/sequencer64` directory. However, if they are not found, we no longer fall back to the legacy configuration file-names. If the `-` legacy option is in force, use only the legacy configuration file-name. The code also ensures the directory exists. CURRENTLY LINUX-SPECIFIC. See the `rc_settings` class for how this works.

```
std::string cfg_dir = seq64::rc().home_config_directory();
if (cfg_dir.empty())
    return EXIT_FAILURE;
```

Change Note ca 2016-04-03 We were parsing the user-file first, but we now need to parse the rc-file first, to get the manual-alsa-ports option, so that we can avoid overriding the port names that the ALSA system provides, if the manual-alsa-option is false.

Parameters

	<i>p</i>	Provides the perform object that will be affected by the new parameters.
out	<i>errmsg</i>	Provides a string into which to dump any error-message that might occur. Still not thoroughly supported in the "rc" and "user" configuration files.
	<i>argc</i>	The number of command-line arguments.
	<i>argv</i>	The array of command-line argument pointers.

Returns

Returns true if the reading of both configuration files succeeded, or if they did not exist. In the latter case, they will be saved as new files upon exit.

9.2.4.48 parse_mute_groups()

```
bool seq64::parse_mute_groups (
    perform & p,
    std::string & errmsg )
```


Parameters

<i>p</i>	The perform object to alter by reading the mute-groups.
<i>errmsg</i>	A return parameter for any error message that might occur.

Returns

Returns true if no errors occurred in reading the mute-groups. If not true, the caller should output the error message.

9.2.4.49 parse_o_options()

```
bool seq64::parse_o_options (
    int argc,
    char * argv[] )
```

These are flagged by "-o" or "--option". These options are then passed to the [user_settings](#) module. Multiple options can be supplied; the whole command-line is processed.

Parameters

<i>argc</i>	The number of command-line parameters, including the name of the application as parameter 0.
<i>argv</i>	The array of pointers to the command-line parameters.

Returns

Returns true if any "options" option was found, and false otherwise. If the options flags ("-o" or "--option") were found, but were not valid options, then we break out of processing and return false.

9.2.4.50 parse_log_option()

```
bool seq64::parse_log_option (
    int argc,
    char * argv[] )
```

Generally, this function needs to be called near the beginning of main(). See the rtmidi version of the main() function, for example.

Parameters

<i>argc</i>	The number of command-line parameters, including the name of the application as parameter 0.
<i>argv</i>	The array of pointers to the command-line parameters.

Returns

Returns true if stdio was rerouted to the "usr"-specified log-file.

9.2.4.51 parse_command_line_options()

```
int seq64::parse_command_line_options (
    perform & p,
    int argc,
    char * argv[] )
```

Note that, since we call this function twice (once before the configuration files are parsed, and once after), we have to make sure that the global value `optind` is reset to 0 before calling this function. Note that the traditional reset value for `optind` is 1, but 0 is used in GNU code to trigger the internal initialization routine of `get_opt()`.

Parameters

<i>p</i>	The performance object that implements some of the command-line options.
<i>argc</i>	The number of command-line arguments.
<i>argv</i>	The array of command-line argument pointers.

Returns

Returns the value of `optind` if no help-related options were provided.

9.2.4.52 write_options_files()

```
bool seq64::write_options_files (
    const perform & p )
```

This function gets any legacy global variables, on the theory that they might have been changed.

Parameters

<i>p</i>	Provides the perform object that may provide new values for the parameters.
----------	---

Returns

Returns true if both files were saved successfully. Otherwise returns false. But even if one write failed, the other might have succeeded.

9.2.4.53 build_details()

```
std::string seq64::build_details ( )
```

Returns

Returns an ordered, human-readable string enumerating the built-in features of this application.

9.2.4.54 check_daemonize()

```
bool seq64::check_daemonize (
    int argc,
    char * argv[] )
```

9.2.4.55 daemonize()

```
uint32_t seq64::daemonize (
    const std::string & appname,
    const std::string & cwd = ".",
    int mask = 0 )
```

9.2.4.56 undaemonize()

```
void seq64::undaemonize (
    uint32_t previous_umask )
```

9.2.4.57 reroute_stdio()

```
bool seq64::reroute_stdio (
    const std::string & logfile,
    bool closem )
```

Todo Implement "daemonizing" for Windows, including redirection to the Windows Event Log.

Still need to figure out a way to do this very simply, a la' Microsoft's 'svchost' executable.

Parameters

<i>logfile</i>	The optional name of the file to which to log messages. Defaults to an empty string.
<i>closem</i>	Just closes the standard file descriptors, rather than rerouting them to /dev/null. Defaults to false. This is the value needed if the <i>logfile</i> parameter is not empty.

Returns

Returns true if the log-file functionality has been enabled. I think :-D.

9.2.4.58 is_note_off_velocity()

```
bool seq64::is_note_off_velocity (
    midibyte status,
    midibyte data ) [inline]
```

Parameters

<i>status</i>	The type of event, which might be EVENT_NOTE_ON.
<i>data</i>	The data byte to check. It should be zero for a note-on is note-off event.

9.2.4.59 to_string()

```
std::string seq64::to_string (
    const event & ev )
```

Nothing fancy. If you want that, use the midicvt project.

Parameters

<i>ev</i>	The event to put on show.
-----------	---------------------------

Returns

Returns the string representation of the event parameter.

9.2.4.60 create_tempo_event()

```
event seq64::create_tempo_event (
    midipulse tick,
    midibpm tempo )
```

9.2.4.61 file_access()

```
bool seq64::file_access (
    const std::string & targetfile,
    int mode )
```

9.2.4.62 file_exists()

```
bool seq64::file_exists (
    const std::string & filename )
```

Parameters

<i>filename</i>	Provides the name of the file to be checked.
-----------------	--

Returns

Returns 'true' if the file exists.

9.2.4.63 file_readable()

```
bool seq64::file_readable (
    const std::string & filename )
```

Parameters

<i>filename</i>	Provides the name of the file to be checked.
-----------------	--

Returns

Returns 'true' if the file is readable.

9.2.4.64 file_writable()

```
bool seq64::file_writable (
    const std::string & filename )
```

Parameters

<i>filename</i>	Provides the name of the file to be checked.
-----------------	--

Returns

Returns 'true' if the file is writable.

9.2.4.65 file_accessible()

```
bool seq64::file_accessible (
    const std::string & filename )
```

An even stronger test than `file_exists`. At present, we see no need to distinguish read and write permissions. We assume the file is accessible only if the file has both permissions.

Parameters

<i>filename</i>	Provides the name of the file to be checked.
-----------------	--

Returns

Returns 'true' if the file is readable and writable.

9.2.4.66 `file_executable()`

```
bool seq64::file_executable (
    const std::string & filename )
```

Parameters

<i>filename</i>	Provides the name of the file to be checked.
-----------------	--

Returns

Returns 'true' if the file exists.

9.2.4.67 `file_is_directory()`

```
bool seq64::file_is_directory (
    const std::string & filename )
```

This function is also used in the function of the same name in `fileutilities.cpp`.

Parameters

<i>filename</i>	Provides the name of the directory to be checked.
-----------------	---

Returns

Returns 'true' if the file is a directory.

9.2.4.68 `make_directory()`

```
bool seq64::make_directory (
    const std::string & pathname )
```

This function is actually a little more general than that, but it is not sufficiently general, in general, General.

Parameters

<i>pathname</i>	Provides the name of the path to create. The parent directory of the final directory must already exist.
-----------------	--

Returns

Returns true if the path-name exists.

9.2.4.69 get_current_directory()

```
std::string seq64::get_current_directory ( )
```

This function is a wrapper for `getcwd()` and other such functions. It obtains the current working directory in the application.

Returns

The pointer to the string containing the name of the current directory. This name is the full path name for the directory. If an error occurs, then an empty string is returned.

9.2.4.70 get_full_path()

```
std::string seq64::get_full_path (
    const std::string & path )
```

It uses the Linux function `realpath(3)`, which returns the canonicalized absolute path-name. For Windows, the function `_fullpath()` is used.

Parameters

<i>path</i>	Provides the path, which may be relative.
-------------	---

Returns

Returns the full path. If a problem occurs, the result is empty.

9.2.4.71 normalize_path()

```
std::string seq64::normalize_path (
    const std::string & path,
    bool to_unix )
```

Parameters

<i>path</i>	Provides the path, which should be a full path-name.
<i>to_unix</i>	Defaults to true, which converts "\" to "/". False converts in the opposite direction.

Returns

The possibly modified path is returned.

9.2.4.72 strip_quotes()

```
std::string seq64::strip_quotes (
    const std::string & item )
```

Meant mainly for removing quotes around a file-name, so it works only if the first character is a quote, and the last character is a quote.

Parameters

<i>item</i>	The string to be massaged.
-------------	----------------------------

Returns

The string without double quotes. If it didn't have any, the string should be unchanged.

9.2.4.73 file_extension()

```
std::string seq64::file_extension (
    const std::string & path )
```

We could use libmagic to determine the actual file type, but that is probably overkill for our purposes.

Parameters

<i>path</i>	Provides the file name or the full path to the file.
-------------	--

Returns

Returns the file extension. If there is no period, then an empty string is returned.

9.2.4.74 strcasecmp()

```
bool seq64::strcasecmp (
    const std::string & a,
    const std::string & b )
```

9.2.4.75 file_extension_match()

```
bool seq64::file_extension_match (
    const std::string & path,
    const std::string & target )
```

Parameters

<i>path</i>	Provides the file name or the full path to the file.
<i>target</i>	Provides the file extension to match against, without the period.

Returns

Returns true if the file extensions match.

9.2.4.76 jack_shutdown_callback()

```
void seq64::jack_shutdown_callback (
    void * arg )
```

This callback is to shut down JACK by clearing the [jack_assistant](#) :: m_jack_running flag.

Parameters

<i>arg</i>	Points to the jack_assistant in charge of JACK support for the perform object.
------------	--

9.2.4.77 jack_timebase_callback()

```
void seq64::jack_timebase_callback (
    jack_transport_state_t state,
    jack_nframes_t nframes,
    jack_position_t * pos,
    int new_pos,
    void * arg )
```

The original version of the function worked properly with Hydrogen, but not with Klick. The new code seems to work with both. More testing and clarification is needed. This new code was "discovered" in the source-code for the "SooperLooper" project:

<http://essey.net/sooperlooper/>

The first difference with the new code is that it handles the case where the JACK position is moved (`new_pos == true`). If this is true, and the `JackPositionBBT` bit is off in `pos->valid`, then the new BBT value is set.

The second set of differences are in the "else" clause. In the new code, it is very simple: calculate the new tick value, back it off by the number of ticks in a beat, and perhaps go to the first beat of the next bar.

In the old code (complex!), the simple BBT adjustment is always made. This changes (perhaps) the `beats_per_bar`, `beat_type`, etc. We need to make these settings use the actual global values for beats set for `Sequencer64`. Then, if transitioning from `JackTransportStarting` to `JackTransportRolling` (instead of checking `new_pos!`), the BBT values (bar, beat, and tick) are finally adjusted. Here are the steps, with old and new steps noted:

```
-# Calculate the "delta" ticks based on the current frame, the
  ticks_per_beat, the beats_per_minute, and the frame_rate. The old
  code saves this in a local, the new code assigns it to pos->tick.
-# Old code: save this delta as a positive value.
-# Figure out the settings and modify bar, beat, tick, and
  bar_start_tick. The old and new code seem to have the same intent,
  but it seems like the new code is faster and also correct.
  - Old code: Calculations are made by division and mod
    operations.
  - New code: Calculations are made by increments and decrements
    in a while loop.
```

Stazed:

The call to [jack_timebase_callback\(\)](#) to supply JACK with BBT, etc. would occasionally fail when the pos information had zero or some garbage in the `pos.frame_rate` variable. This would occur when there was a rapid change of frame position by another client... i.e. `qjackctl`. From the JACK API:

```
pos address of the position structure for the next cycle; pos->frame
will be its frame number. If new_pos is FALSE, this structure contains
extended position information from the current cycle. If TRUE, it
contains whatever was set by the requester. The timebase_callback's
task is to update the extended information here."
```

The "If TRUE" line seems to be the issue. It seems that `qjackctl` does not always set `pos.frame_rate` so we get garbage and some strange BBT calculations that display in `qjackctl`. So we need to set it here and just use `m_↵ jack_frame_rate` for calculations instead of `pos.frame_rate`.

Parameters

<i>state</i>	Indicates the current state of JACK transport.
<i>nframes</i>	The number of JACK frames in the current time period.
<i>pos</i>	Provides the position structure to be filled in, the address of the position structure for the next cycle; <code>pos->frame</code> will be its frame number. If <code>new_pos</code> is FALSE, this structure contains extended position information from the current cycle. If TRUE, it contains whatever was set by the requester. The <code>timebase_callback</code> 's task is to update the extended information here.
<i>new_pos</i>	TRUE (non-zero) for a newly requested pos, or for the first cycle after the <code>timebase_callback</code> is defined. This is usually 0 in <code>Sequencer64</code> at present, and 1 if one, say, presses "rewind" in <code>qjackctl</code> .
<i>arg</i>	Provides the jack_assistant pointer, currently unchecked for nullity.

Todo Shouldn't we process the first clause ONLY if `new_pos` is true?

Todo Shouldn't we process the first clause ONLY if `new_pos` is true?

9.2.4.78 jack_transport_callback()

```
int seq64::jack_transport_callback (
    jack_nframes_t nframes,
    void * arg )
```

Added the following function. This function is supposed to allow seq24/sequencer64 to follow JACK transport.

For more advanced ideas, see the `MetronomeJack::process_callback()` function in the `klick` project. It plays a metronome tick after calculating if it needs to or not. (Maybe we could use it to provide our own tick for recording patterns.)

If debugging is enabled in the build, then this function outputs the current JACK position information every couple of seconds, which can be useful to examine the interactions with other JACK clients.

The code enabled via `USE_JACK_BBT_OFFSET` sets the JACK position field `bbt_offset` to 0. It doesn't seem to have any effect, though it can be seen when calling `show_position()` in the [jack_transport_callback\(\)](#) function.

Stazed:

```
This process callback is called by JACK whether stopped or rolling.
Assuming every JACK cycle... "...client supplied function that is
called by the engine anytime there is work to be done". There seems to
be no definition of '...work to be done'. nframes = buffer_size -- is
not used.
```

Parameters

<i>nframes</i>	Unused.
<i>arg</i>	Used for debug output now. Note that this function will be called very often, and this pointer will currently not be used unless debugging is turned on.

Returns

Returns 0 on success, non-zero on error.

9.2.4.79 create_jack_client()

```
jack_client_t * seq64::create_jack_client (
    const std::string & clientname,
    const std::string & uuid )
```

Do not call this function if the JACK client handle is already open.

Status bits for `jack_status_t` return pointer:

```
JackNameNotUnique means that the client name was not unique. With
JackUseExactName, this is fatal. Otherwise, the name was modified by
appending a dash and a two-digit number in the range "-01" to "-99".
The jack_get_client_name() function returns the exact string used. If
the specified client_name plus these extra characters would be too
long, the open fails instead.
```

```
JackServerStarted means that the JACK server was started as a result
of this operation. Otherwise, it was running already. In either case
the caller is now connected to jackd, so there is no race condition.
When the server shuts down, the client will find out.
```

JackOpenOptions:

```
JackSessionID | JackServerName | JackNoStartServer | JackUseExactName
```

helgrind:

Valgrind's helgrind tool shows

```
Possible data race during read of size 4 at 0xF854E58 by thread #1
  by 0x267602: seq64::create_jack_client(...)
This conflicts with a previous write of size 4 by thread #2
  by 0x267602: seq64::create_jack_client(...)
```

```
So we add a static mutex to use with our automutex. Does not prevent
that message..... WHY?
```

We've never disabled the SEQ64_JACK_SESSION macro, and we like the error-reporting we get by that method. So we've commented out the following code in favor of using the session-uuid code:

```
ifndef SEQ64_JACK_SESSION
```

```
else
```

```
jack_status_t * ps = NULL; result = jack_client_open(name, JackNullOption, ps);
```

```
endif
```

Parameters

<i>clientname</i>	Provides the name of the client, used in the call to jack_client_open(). By default, this name is the macro SEQ64_PACKAGE (i.e. "sequencer64"). The name scope is local to each server. Unless forbidden by the JackUseExactName option, the server will modify this name to create a unique variant, if needed.
<i>uuid</i>	The optional UUID to assign to the new client. If empty, there is no UUID.

Returns

Returns a pointer to the JACK client if JACK has opened the client connection successfully. Otherwise, a null pointer is returned.

9.2.4.80 show_jack_statuses()

```
void seq64::show_jack_statuses (
    unsigned bits )
```

For reference, here are the enumeration values from /usr/include/jack/types.h:

JackFailure	= 0x01
JackInvalidOption	= 0x02
JackNameNotUnique	= 0x04
JackServerStarted	= 0x08
JackServerFailed	= 0x10
JackServerError	= 0x20
JackNoSuchClient	= 0x40
JackLoadFailure	= 0x80
JackInitFailure	= 0x100
JackShmFailure	= 0x200
JackVersionError	= 0x400
JackBackendError	= 0x800
JackClientZombie	= 0x1000

Parameters

<i>bits</i>	The mask of the bits to be shown in the output.
-------------	---

9.2.4.81 get_current_jack_position()

```
long seq64::get_current_jack_position (
    void * arg )
```

The Seq32 version also uses its tempo map to adjust this, but Sequencer64 currently does not.

Warning

Currently valgrind flags `j->client()` as uninitialized.

Parameters

<i>arg</i>	Provides the putative jack_assistant pointer, assumed to be not null.
------------	---

Returns

Returns the calculated tick position if no errors occur. Otherwise, returns 0.

9.2.4.82 jack_session_callback()

```
void seq64::jack_session_callback (
    jack_session_event_t * ev,
    void * arg )
```

Glib is then used to connect in `perform::jack_session_event()`. However, the `perform` object's GUI-support interface is used instead of the following, so that the `libseq64` library can be independent of a specific GUI framework:

```
Glib::signal_idle().
    connect(sigc::mem_fun(*jack, &jack_assistant::session_event));
```

Parameters

<i>ev</i>	The JACK event to be set.
<i>arg</i>	The pointer to the jack_assistant object. Currently not checked for nullity.

9.2.4.83 `invalid_key()`

```
bool seq64::invalid_key (
    unsigned key ) [inline]
```

The implementation of this function will ultimately depend on the GUI environment; currently it is GTK 2.x, so the implementation is in `seq_gtkmm2/src/keys_perform_gtk2.cpp`.

9.2.4.84 `keyval_normalize()`

```
void seq64::keyval_normalize (
    keys_perform_transfer & k )
```

Otherwise, random values, unchecked, can cause the application to crash.

Any field that is 0 or greater than 65536 is fixed. Not perfect, but better than allowing random values to be used.

Parameters

<i>k</i>	The structure to be validated and normalized.
----------	---

9.2.4.85 `create_lash_driver()`

```
bool seq64::create_lash_driver (
    perform & p,
    int argc,
    char ** argv )
```

Initializes the lash driver (strips lash-specific command line arguments), then connects to the LASH daemon and polls events.

This function will always be called from the main routine, and called only once. Note that we don't need that darn `SEQ64_LASH_SUPPORT` macro in client code anymore.

Parameters

<i>p</i>	The perform object that needs to implement LASH support.
<i>argc</i>	The number of command-line arguments.
<i>argv</i>	The command-line arguments.

Returns

This function returns true if a lash object was created. This function will not create one if not configured to, if the command-line options did not specify the creation of the LASH driver, or if the LASH driver was already created.

9.2.4.86 lash_driver()

```
lash * seq64::lash_driver ( )
```

Returns

Returns the pointer to the LASH driver if it exists. Otherwise a null pointer is returned. The caller *must always check* the return value.

9.2.4.87 delete_lash_driver()

```
void seq64::delete_lash_driver ( )
```

This function will always be called from the main routine, once. The other lash-pointer functions will know if the pointer has been deleted.

9.2.4.88 millisleep()

```
void seq64::millisleep (
    unsigned long ms )
```

9.2.4.89 is_null_midipulse()

```
bool seq64::is_null_midipulse (
    midipulse p ) [inline]
```

By "null" in this case, we mean "unusable", not 0. Sigh, it's always something.

9.2.4.90 output_thread_func()

```
void * seq64::output_thread_func (
    void * myperf )
```

Set up the performance, set the process to realtime privileges, and then start the output function.

Parameters

<i>myperf</i>	Provides the perform object instance that is to be used. Its output_func() is called. Currently, this parameter is not validated, for speed.
---------------	--

Returns

Always returns nullptr.

9.2.4.91 input_thread_func()

```
void * seq64::input_thread_func (
    void * myperf )
```

Parameters

<i>myperf</i>	Provides the perform object instance that is to be used. Its output_func() is called. Currently, this parameter is not validated, for speed.
---------------	--

Returns

Always returns nullptr.

9.2.4.92 rc()

```
rc_settings & seq64::rc ( )
```

Why a function instead of direct variable access? Encapsulation. We are then free to change the way "global" settings are accessed, without changing client code.

Returns

Returns the global object g_rc_settings.

9.2.4.93 usr()

```
user_settings & seq64::usr ( )
```

Returns

Returns the global object g_user_settings.

9.2.4.94 choose_ppqn()

```
int seq64::choose_ppqn (
    int ppqn )
```

Putting it here means we can reduce the reliance on the global ppqn.

However, this function works completely only if the "user" configuration file has already been read. In some cases we may need to retrofit the desired PPQN value!

Parameters

<i>ppqn</i>	Provides the PPQN value to be used.
-------------	-------------------------------------

Returns

Returns the ppqn parameter, unless that parameter is SEQ64_USE_DEFAULT_PPQN (-1), then [usr\(\).midi←_ppqn](#) is returned. If that value is also -1, then we return SEQ64_DEFAULT_PPQN (192).

9.2.4.95 timestring_to_pulses() [2/2]

```
midipulse seq64::timestring_to_pulses (
    const std::string & timestring,
    midibpm bpm,
    int ppqn )
```

Parameters

<i>timestring</i>	The time value to be converted, which must be of the form "hh:mm:ss" or "hh:mm:ss.fraction". That is, all four parts must be found.
<i>bpm</i>	The beats-per-minute tempo (e.g. 120) of the current MIDI song.
<i>ppqn</i>	The parts-per-quarter note precision (e.g. 192) of the current MIDI song.

Returns

Returns 0 if an error occurred or if the number actually translated to 0.

This conversion assumes that the fractional parts of the seconds is padded with zeroes on the left or right to 6 digits.

9.2.4.96 get_compound_option()

```
static std::string seq64::get_compound_option (
    const std::string & compound,
    std::string & optionname ) [static]
```

An option argument is a value flagged on the command line by the -o/-option options. Each option has a value associated with it, as the next argument on the command-line. A compound option is of the form name=value, of which one example is:

```
log=filename
```

This function extracts both the name and the value. If the name is empty, then the option is unsupported and should be ignored. If the value is empty, then there may be a default value to use.

Parameters

	<i>compound</i>	The putative compound option.
out	<i>optionname</i>	This value is filled in with the name of the option-value, if there is an equals sign in the <i>compound</i> parameter.

Returns

Returns the value part of the compound option, or just the compound parameter if there is no '=' sign. That is, it returns the option-value.

Side-effect(s) The name portion is returned in the optionname parameter.

9.2.4.97 set_current_directory()

```
bool seq64::set_current_directory (
    const std::string & path )
```

A wrapper replacement for `chdir()` or `_chdir()`. It sets the current working directory in the application. This function is necessary in order to make sure the current working directory is a safe place to work.

Returns

Returns true if the path name is good, and the `chdir()` call succeeded. Otherwise, false is returned.

Parameters

<i>path</i>	The full or relative name of the directory.
-------------	---

9.2.4.98 message_concatenate()

```
std::string seq64::message_concatenate (
    const char * m1,
    const char * m2 )
```

Note that we don't bother with error-checking the pointers. You're on your own, Hoss.

Parameters

<i>m1</i>	The first message, often a func macro.
<i>m2</i>	The second message.

Returns

Returns "m1: m2" as a standard C++ string.

9.2.4.99 info_message()

```
bool seq64::info_message (
    const std::string & msg )
```

Adds markers and a newline.

Parameters

<i>msg</i>	The message to print, sans the newline.
------------	---

Returns

Returns true.

9.2.4.100 error_message()

```
bool seq64::error_message (
    const std::string & msg )
```

Adds markers, and returns false.

Parameters

<i>msg</i>	The message to print, sans the newline.
------------	---

Returns

Returns false for convenience/brevity in setting function return values.

9.2.4.101 casecompare()

```
static bool seq64::casecompare (
    char a,
    char b ) [inline], [static]
```

9.2.4.102 jack_dummy_callback()

```
int seq64::jack_dummy_callback (
    jack_nframes_t nframes,
    void * arg )
```

Parameters

<i>nframes</i>	An unused parameter.
<i>arg</i>	Provides the jack_assistant pointer.

Returns

Does nothing, but returns nframes. If the arg parameter is null, then 0 is returned.

9.2.4.103 seq_app_name()

```
const std::string& seq64::seq_app_name ( )
```

We could continue to use the macro SEQ64_APP_NAME, but we might eventually want to make this name configurable. Not too likely, but possible.

Returns

Returns SEQ64_APP_NAME.

9.2.4.104 seq_client_name()

```
const std::string& seq64::seq_client_name ( )
```

We could continue to use the macro SEQ64_CLIENT_NAME, but we might eventually want to make this name configurable. More likely to be a configuration option in the future.

Returns

Returns SEQ64_CLIENT_NAME.

9.2.4.105 seq_version()

```
const std::string& seq64::seq_version ( )
```

We could continue to use the macro SEQ64_VERSION, but ... let's be consistent. :-D

Returns

Returns SEQ64_VERSION.

9.2.4.106 make_section_name()

```
static std::string seq64::make_section_name (
    const std::string & label,
    int value ) [static]
```

Parameters

<i>label</i>	The base-name of the section.
<i>value</i>	The numeric value to append to the section name.

Returns

Returns a string of the form "[basename-1]".

9.2.4.107 font_render()

```
font& seq64::font_render ( ) [inline]
```

We've going to render this pointer obsolete, though, and use a smart factory function to ensure the existence of this pointer, and return a reference to the font object.

We wanted to make the font a const object, but `mainwid::on_realize()` calls the `font::init()` function with its window object, and using const is impractical. We don't want to force every caller to deal with the overhead of passing even a null window pointer, either.

However, at some point we need some guarantee that the `init()` function is called before rendering a string. Right now, we guarantee it only by build order.

Returns

Returns a reference to the object pointed to by `sp_font_renderer`.

9.2.4.108 adjustment_dummy()

```
Gtk::Adjustment & seq64::adjustment_dummy ( )
```

This static object is used so we have an `Adjustment` to assign to the `Adjustment` members for classes that don't use them. Clumsy? We shall see.

Anyway, the parameters for this constructor are value, lower, upper, step-increment, and two more values.

9.2.4.109 is_ctrl_key() [1/3]

```
bool seq64::is_ctrl_key (
    GdkEventKey * ev )
```

It's a shame that `GdkEventAny` doesn't also encapsulate the keyboard state, since that is also available for other events, such as scroll events.

Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

Returns

Returns true if the event state includes SEQ64_CONTROL_MASK.

9.2.4.110 is_shift_key() [1/3]

```
bool seq64::is_shift_key (  
    GdkEventKey * ev )
```

Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

Returns

Returns true if the event state includes SEQ64_SHIFT_MASK.

9.2.4.111 is_no_modifier()

```
bool seq64::is_no_modifier (  
    GdkEventScroll * ev )
```

Parameters

<i>ev</i>	The scroll event to be tested.
-----------	--------------------------------

Returns

Returns true if there are no modifiers in force.

9.2.4.112 is_ctrl_key() [2/3]

```
bool seq64::is_ctrl_key (  
    GdkEventScroll * ev )
```

Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

Returns

Returns true if the event state includes SEQ64_CONTROL_MASK.

9.2.4.113 is_shift_key() [2/3]

```
bool seq64::is_shift_key (  
    GdkEventScroll * ev )
```

Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

Returns

Returns true if the event state includes SEQ64_SHIFT_MASK.

9.2.4.114 is_ctrl_key() [3/3]

```
bool seq64::is_ctrl_key (  
    GdkEventButton * ev )
```

Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

Returns

Returns true if the event state includes SEQ64_CONTROL_MASK.

9.2.4.115 is_shift_key() [3/3]

```
bool seq64::is_shift_key (  
    GdkEventButton * ev )
```

Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

Returns

Returns true if the event state includes SEQ64_SHIFT_MASK.

9.2.4.116 is_ctrl_shift_key()

```
bool seq64::is_ctrl_shift_key (  
    GdkEventButton * ev )
```

Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

Returns

Returns true if the event state includes SEQ64_SHIFT_MASK and SEQ64_CONTROL_MASK.

9.2.4.117 is_super_key()

```
bool seq64::is_super_key (  
    GdkEventButton * ev )
```

Basically just masks off the MOD4 bit; the "safe" method does not work for this key.

Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

Returns

Returns true if the event state includes SEQ64_MOD4_MASK.

9.2.4.118 test_widget_click()

```
void seq64::test_widget_click (  
    GtkWidget * w )
```


Parameters

<i>w</i>	Points to the widget being clicked.
----------	-------------------------------------

9.2.4.119 `is_left_drag()`

```
bool seq64::is_left_drag (
    GdkEventMotion * ev )
```

Parameters

<i>ev</i>	Points to the event-motion structure.
-----------	---------------------------------------

Returns

Returns true if the SEQ64_BUTTON1_MASK bit is active.

9.2.4.120 `is_drag_motion()`

```
bool seq64::is_drag_motion (
    GdkEventMotion * ev )
```

Parameters

<i>ev</i>	Points to the event-motion structure.
-----------	---------------------------------------

Returns

Returns true if one of the SEQ64_BUTTON1_MASK, SEQ64_BUTTON2_MASK, or SEQ64_BUTTON3_MASK bits are active.

9.2.4.121 `keyval_name()`

```
std::string seq64::keyval_name (
    unsigned key )
```

Obtains the name of the key.

The implementation of this function will ultimately depend on the GUI environment; currently it is GTK 2.x, so the implementation is in `seq_gtkmm2/src/keys_perform_gtk2.cpp`.

In `gtkmm`, this is done via the `gdk_keyval_name()` function. Here, in the base class, we just provide an easy-to-create string. Note that this is a free function, not a class member.

Parameters

<i>key</i>	Provides the key-number to be converted to a key name.
------------	--

Returns

Returns the key name as looked up by the GDK infrastructure. If the key is not found, then an empty string is returned.

9.2.4.122 update_mainwid_sequences()

```
void seq64::update_mainwid_sequences ( )
```

It is used by other objects that can modify the currently-edited sequence shown in the mainwid (main window).

9.2.4.123 update_perfeddit_sequences()

```
void seq64::update_perfeddit_sequences ( )
```

It is used by other objects (seqedit and eventedit) that can modify the currently-edited sequence shown in the perfedit (song window).

9.2.4.124 FF_RW_timeout()

```
int seq64::FF_RW_timeout (
    void * arg ) [inline]
```

Parameters

<i>arg</i>	Provides a putative pointer to the perform object that actually implements the timeout functionality.
------------	---

Returns

Returns the value of the [perform::FF_RW_timeout\(\)](#) call if seq32 transport support is enabled and the arg parameter is good, otherwise false is returned.

9.2.4.125 clamp()

```
static long seq64::clamp (
    long val,
    long low,
    long hi ) [inline], [static]
```

9.2.4.126 `silence_jack_errors()`

```
void seq64::silence_jack_errors (
    bool silent = true )
```

9.2.4.127 `silence_jack_info()`

```
void seq64::silence_jack_info (
    bool silent = true )
```

9.2.4.128 `midi_api_name()`

```
std::string seq64::midi_api_name (
    int i )
```

9.2.4.129 `midi_probe()`

```
int seq64::midi_probe ( )
```

9.2.4.130 `midi_input_test()`

```
bool seq64::midi_input_test (
    rtmidi_info & info,
    int portindex )
```

9.2.5 Variable Documentation**9.2.5.1** `c_controller_names`

```
std::string seq64::c_controller_names
```

This array is used only by the seqedit class.

9.2.5.2 EVENT_STATUS_BIT

```
const midibyte seq64::EVENT_STATUS_BIT
```

9.2.5.3 EVENT_ANY

```
const midibyte seq64::EVENT_ANY
```

The following MIDI events are channel messages. The comments represent the one or two data-bytes of the message.

Note that Channel Mode Messages use the same code as the Control Change, but uses reserved controller numbers ranging from 122 to 127.

The EVENT_ANY (0x00) value may prove to be useful in allowing any event to be dealt with. Not sure yet, but the cost is minimal.

9.2.5.4 EVENT_NOTE_OFF

```
const midibyte seq64::EVENT_NOTE_OFF
```

9.2.5.5 EVENT_NOTE_ON

```
const midibyte seq64::EVENT_NOTE_ON
```

9.2.5.6 EVENT_AFTERTOUCH

```
const midibyte seq64::EVENT_AFTERTOUCH
```

9.2.5.7 EVENT_CONTROL_CHANGE

```
const midibyte seq64::EVENT_CONTROL_CHANGE
```

9.2.5.8 EVENT_PROGRAM_CHANGE

```
const midibyte seq64::EVENT_PROGRAM_CHANGE
```

9.2.5.9 EVENT_CHANNEL_PRESSURE

```
const midibyte seq64::EVENT_CHANNEL_PRESSURE
```

9.2.5.10 EVENT_PITCH_WHEEL

```
const midibyte seq64::EVENT_PITCH_WHEEL
```

9.2.5.11 EVENT_CTRL_VOLUME

```
const midibyte seq64::EVENT_CTRL_VOLUME
```

9.2.5.12 EVENT_CTRL_BALANCE

```
const midibyte seq64::EVENT_CTRL_BALANCE
```

9.2.5.13 EVENT_CTRL_PAN

```
const midibyte seq64::EVENT_CTRL_PAN
```

9.2.5.14 EVENT_CTRL_EXPRESSION

```
const midibyte seq64::EVENT_CTRL_EXPRESSION
```

9.2.5.15 EVENT_MIDI_REALTIME

```
const midibyte seq64::EVENT_MIDI_REALTIME
```

The following MIDI events have no channel. We have included redundant constant variables for the SysEx Start and End bytes just to make it clear that they are part of this sequence of values, though usually treated separately.

Only the following constants are followed by some data bytes:

```
- EVENT_MIDI_SYSEX           = 0xF0
- EVENT_MIDI_QUARTER_FRAME   = 0xF1      // undefined?
- EVENT_MIDI_SONG_POS        = 0xF2
- EVENT_MIDI_SONG_SELECT     = 0xF3
```

A MIDI System Exclusive (SYSEX) message starts with F0, followed by the manufacturer ID (how many? bytes), a number of data bytes, and ended by an F7.

MIDI System Real-Time Messages:

- https://en.wikipedia.org/wiki/MIDI_beat_clock
- <http://www.midi.org/techspecs/midimessages.php>

9.2.5.16 EVENT_MIDI_SYSEX

```
const midibyte seq64::EVENT_MIDI_SYSEX
```

9.2.5.17 EVENT_MIDI_QUARTER_FRAME

```
const midibyte seq64::EVENT_MIDI_QUARTER_FRAME
```

9.2.5.18 EVENT_MIDI_SONG_POS

```
const midibyte seq64::EVENT_MIDI_SONG_POS
```

9.2.5.19 EVENT_MIDI_SONG_SELECT

```
const midibyte seq64::EVENT_MIDI_SONG_SELECT
```

9.2.5.20 EVENT_MIDI_SONG_F4

```
const midibyte seq64::EVENT_MIDI_SONG_F4
```

9.2.5.21 EVENT_MIDI_SONG_F5

```
const midibyte seq64::EVENT_MIDI_SONG_F5
```

9.2.5.22 EVENT_MIDI_TUNE_SELECT

```
const midibyte seq64::EVENT_MIDI_TUNE_SELECT
```

9.2.5.23 EVENT_MIDI_SYSEX_END

```
const midibyte seq64::EVENT_MIDI_SYSEX_END
```

9.2.5.24 EVENT_MIDI_SYSEX_CONTINUE

```
const midibyte seq64::EVENT_MIDI_SYSEX_CONTINUE
```

9.2.5.25 EVENT_MIDI_CLOCK

```
const midibyte seq64::EVENT_MIDI_CLOCK
```

9.2.5.26 EVENT_MIDI_SONG_F9

```
const midibyte seq64::EVENT_MIDI_SONG_F9
```

9.2.5.27 EVENT_MIDI_START

```
const midibyte seq64::EVENT_MIDI_START
```

9.2.5.28 EVENT_MIDI_CONTINUE

```
const midibyte seq64::EVENT_MIDI_CONTINUE
```

9.2.5.29 EVENT_MIDI_STOP

```
const midibyte seq64::EVENT_MIDI_STOP
```

9.2.5.30 EVENT_MIDI_SONG_FD

```
const midibyte seq64::EVENT_MIDI_SONG_FD
```

9.2.5.31 EVENT_MIDI_ACTIVE_SENSE

```
const midibyte seq64::EVENT_MIDI_ACTIVE_SENSE
```

9.2.5.32 EVENT_MIDI_RESET

```
const midibyte seq64::EVENT_MIDI_RESET
```

9.2.5.33 EVENT_MIDI_META

```
const midibyte seq64::EVENT_MIDI_META
```

Note that it has the same code (0xFF) as the Reset message, but the Meta message is read from a MIDI file, while the Reset message is sent to the sequencer by other MIDI participants.

9.2.5.34 EVENT_META_SEQ_NUMBER

```
const midibyte seq64::EVENT_META_SEQ_NUMBER
```

- Set Tempo (0x51)
- Time Signature (0x58)

9.2.5.35 EVENT_META_TEXT_EVENT

```
const midibyte seq64::EVENT_META_TEXT_EVENT
```

9.2.5.36 EVENT_META_COPYRIGHT

```
const midibyte seq64::EVENT_META_COPYRIGHT
```

9.2.5.37 EVENT_META_TRACK_NAME

```
const midibyte seq64::EVENT_META_TRACK_NAME
```

9.2.5.38 EVENT_META_INSTRUMENT

```
const midibyte seq64::EVENT_META_INSTRUMENT
```


9.2.5.39 EVENT_META_LYRIC

```
const midibyte seq64::EVENT_META_LYRIC
```

9.2.5.40 EVENT_META_MARKER

```
const midibyte seq64::EVENT_META_MARKER
```

9.2.5.41 EVENT_META_CUE_POINT

```
const midibyte seq64::EVENT_META_CUE_POINT
```

9.2.5.42 EVENT_META_MIDI_CHANNEL

```
const midibyte seq64::EVENT_META_MIDI_CHANNEL
```

9.2.5.43 EVENT_META_MIDI_PORT

```
const midibyte seq64::EVENT_META_MIDI_PORT
```

9.2.5.44 EVENT_META_END_OF_TRACK

```
const midibyte seq64::EVENT_META_END_OF_TRACK
```

9.2.5.45 EVENT_META_SET_TEMPO

```
const midibyte seq64::EVENT_META_SET_TEMPO
```

9.2.5.46 EVENT_META_SMPTE_OFFSET

```
const midibyte seq64::EVENT_META_SMPTE_OFFSET
```

9.2.5.47 EVENT_META_TIME_SIGNATURE

```
const midibyte seq64::EVENT_META_TIME_SIGNATURE
```

9.2.5.48 EVENT_META_KEY_SIGNATURE

```
const midibyte seq64::EVENT_META_KEY_SIGNATURE
```

9.2.5.49 EVENT_META_SEQSPEC

```
const midibyte seq64::EVENT_META_SEQSPEC
```

9.2.5.50 EVENT_META_ILLEGAL

```
const midibyte seq64::EVENT_META_ILLEGAL
```

9.2.5.51 EVENT_NULL_CHANNEL

```
const midibyte seq64::EVENT_NULL_CHANNEL
```

However, it also means that the channel, if applicable to the event, is encoded in the `m_status` byte itself. This is our work around to be able to hold a multi-channel SMF 0 track in a sequence. In a Sequencer64 SMF 0 track, every event has a channel. In a Sequencer64 SMF 1 track, the events do not have a channel. Instead, the channel is a global value of the sequence, and is stuffed into each event when the event is played or is written to a MIDI file.

9.2.5.52 EVENT_GET_CHAN_MASK

```
const midibyte seq64::EVENT_GET_CHAN_MASK
```

9.2.5.53 EVENT_CLEAR_CHAN_MASK

```
const midibyte seq64::EVENT_CLEAR_CHAN_MASK
```

9.2.5.54 EVENTS_ALL

```
const int seq64::EVENTS_ALL
```

We reversed the parts of each token for consistency with the macros defined above.

9.2.5.55 EVENTS_UNSELECTED

```
const int seq64::EVENTS_UNSELECTED
```

9.2.5.56 c_midibus_output_size

```
const int seq64::c_midibus_output_size
```

These constants were also defined in `midibus_portmidi.h`, but we made them common to both implementations here.

The `c_midibus_output_size` value is passed, in `mastermidibus`, to `snd_seq_set_output_buffer_size()`. Not sure if the value needs to be so large.

9.2.5.57 c_midibus_input_size

```
const int seq64::c_midibus_input_size
```

Not sure if the value needs to be so large.

9.2.5.58 c_midibus_sysex_chunk

```
const int seq64::c_midibus_sysex_chunk
```

9.2.5.59 c_midibus

```
const midilong seq64::c_midibus
```

Some of the information is stored with each track (and in the midi_container-derived classes), and some is stored in the proprietary header.

Track (sequencer-specific) data:

```
c_midibus
c_midich
c_timesig
c_triggers (deprecated)
c_triggers_new
c_musickey (can be in footer, as well)
c_musicscale (ditto)
c_backsequence (ditto)
c_transpose
c_seq_color (performance colors for a sequence)
```

Note that c_seq_color in Sequencer64 is stored per sequence, and only if not PaletteColor::NONE. In Kepler34, all 1024 sequence colors are stored in a "proprietary", whole-song section, whether used or not. There are also numeric differences in many of these "c_feature" constants.

Footer ("proprietary", whole-song) data:

```
c_midictrl
c_midiclocks
c_notes
c_bpmtag (beats per minute)
c_mutegroups
c_perf_bp_mes (perfedit's beats-per-measure setting)
c_perf_bw (perfedit's beat-width setting)
c_tempo_map (seq32's tempo map)
c_reserved_1 and c_reserved_2
c_tempo_track (holds the song's particular tempo track)
c_seq_edit_mode (a potential future feature from Kepler34).
```

Also see the PDF file in the following project for more information about the "proprietary" data:

<https://github.com/ahlstromcj/sequencer64-doc.git>

Note that the track data is read from the MIDI file, but not written directly to the MIDI file. Instead, it is stored in the MIDI container as sequences are edited to use these "sequencer-specific" features. Also note that c_triggers has been replaced by c_triggers_new as the code that marks the triggers stored with a sequence.

As an extension, we can also grab the key, scale, and background sequence value selected in a sequence and write these values as track data, where they can be read in and applied to a specific sequence, when the seqedit object is created. These values would not be stored in the legacy format.

Something like this could be done in the "user" configuration file, but then the key and scale would apply to all songs. We don't want that.

We could also add snap and note-length to the per-song defaults, but the "user" configuration file seems like a better place to store these preferences.

Note

- The value `c_transpose` value is from Stazed's seq32 project. The code to support this option is turned on via the build-configurable `SEQ64_STAZED_TRANSPOSE` macro, but here we reserved the value even if that option is not enabled by the user. There are additional values from Stazed/seq32, not yet used.
- The values below are compatible with Seq32, but they are not compatible with Kepler34. It uses 0x24240011 and 0x24240012 for difference purposes. See the asterisks below.
- Note the new "gap" values. We just noticed this gap, which has existed between 0x24240009 and 0x24240010 since Seq24!Track buss number.

9.2.5.60 `c_midich`

```
const midilong seq64::c_midich
```

9.2.5.61 `c_midiclocks`

```
const midilong seq64::c_midiclocks
```

9.2.5.62 `c_triggers`

```
const midilong seq64::c_triggers
```

9.2.5.63 `c_notes`

```
const midilong seq64::c_notes
```

9.2.5.64 `c_timesig`

```
const midilong seq64::c_timesig
```

9.2.5.65 `c_bpmtag`

```
const midilong seq64::c_bpmtag
```

9.2.5.66 c_triggers_new

```
const midilong seq64::c_triggers_new
```

9.2.5.67 c_mutegroups

```
const midilong seq64::c_mutegroups
```

9.2.5.68 c_gap_A

```
const midilong seq64::c_gap_A
```

A.

9.2.5.69 c_gap_B

```
const midilong seq64::c_gap_B
```

B.

9.2.5.70 c_gap_C

```
const midilong seq64::c_gap_C
```

C.

9.2.5.71 c_gap_D

```
const midilong seq64::c_gap_D
```

D.

9.2.5.72 c_gap_E

```
const midilong seq64::c_gap_E
```

E.

9.2.5.73 c_gap_F

```
const midilong seq64::c_gap_F
```

F.

9.2.5.74 c_midictrl

```
const midilong seq64::c_midictrl
```

9.2.5.75 c_musickey

```
const midilong seq64::c_musickey
```

-

9.2.5.76 c_musicscale

```
const midilong seq64::c_musicscale
```

-

9.2.5.77 c_backsequence

```
const midilong seq64::c_backsequence
```

9.2.5.78 c_transpose

```
const midilong seq64::c_transpose
```

9.2.5.79 c_perf_bp_mes

```
const midilong seq64::c_perf_bp_mes
```

9.2.5.80 c_perf_bw

```
const midilong seq64::c_perf_bw
```

9.2.5.81 c_tempo_map

```
const midilong seq64::c_tempo_map
```

9.2.5.82 c_reserved_1

```
const midilong seq64::c_reserved_1
```

9.2.5.83 c_reserved_2

```
const midilong seq64::c_reserved_2
```

9.2.5.84 c_tempo_track

```
const midilong seq64::c_tempo_track
```

9.2.5.85 c_seq_color

```
const midilong seq64::c_seq_color
```

-

9.2.5.86 c_seq_edit_mode

```
const midilong seq64::c_seq_edit_mode
```

-

9.2.5.87 c_midi_track_ctrl

```
const int seq64::c_midi_track_ctrl
```

The lowest value is `c_seqs_in_set * 2 = 64`.

I think the reason for that value is to perhaps handle two sets or something like that. Will figure it out later.

The controls are read in from the "rc" configuration files, and are written to the `c_midictrl` section of the "proprietary" final track in a Seq24/Sequencer64 MIDI file.

Note that we are adding some more MIDI control entries to support the following additional functions:

- Start
- Pause
- Stop
- (any more???)

To help with backward compatibility, the old `c_midi_controls` limit is supplemented with a new, higher limit, `c_midi_controls_extended`. We also add a number of placeholders so we don't have to adjust the new limit again later. To aid the transition, `g_midi_control_limit` replaces `c_midi_controls`, though, for now, it has the same value.

9.2.5.88 c_midi_control_bpm_up

```
const int seq64::c_midi_control_bpm_up
```

9.2.5.89 c_midi_control_bpm_dn

```
const int seq64::c_midi_control_bpm_dn
```

9.2.5.90 c_midi_control_ss_up

```
const int seq64::c_midi_control_ss_up
```

9.2.5.91 c_midi_control_ss_dn

```
const int seq64::c_midi_control_ss_dn
```

9.2.5.92 c_midi_control_mod_replace

```
const int seq64::c_midi_control_mod_replace
```

9.2.5.93 c_midi_control_mod_snapshot

```
const int seq64::c_midi_control_mod_snapshot
```

9.2.5.94 c_midi_control_mod_queue

```
const int seq64::c_midi_control_mod_queue
```

9.2.5.95 c_midi_control_mod_gmute

```
const int seq64::c_midi_control_mod_gmute
```

9.2.5.96 c_midi_control_mod_glearn

```
const int seq64::c_midi_control_mod_glearn
```

9.2.5.97 c_midi_control_play_ss

```
const int seq64::c_midi_control_play_ss
```

9.2.5.98 c_midi_controls

```
const int seq64::c_midi_controls
```

9.2.5.99 c_midi_control_playback

```
const int seq64::c_midi_control_playback
```

9.2.5.100 c_midi_control_song_record

```
const int seq64::c_midi_control_song_record
```

9.2.5.101 c_midi_control_solo

```
const int seq64::c_midi_control_solo
```

9.2.5.102 c_midi_control_thru

```
const int seq64::c_midi_control_thru
```

9.2.5.103 c_midi_control_bpm_page_up

```
const int seq64::c_midi_control_bpm_page_up
```

9.2.5.104 c_midi_control_bpm_page_dn

```
const int seq64::c_midi_control_bpm_page_dn
```

9.2.5.105 c_midi_control_ss_set

```
const int seq64::c_midi_control_ss_set
```

9.2.5.106 c_midi_control_record

```
const int seq64::c_midi_control_record
```

9.2.5.107 c_midi_control_quan_record

```
const int seq64::c_midi_control_quan_record
```

9.2.5.108 c_midi_control_reset_seq

```
const int seq64::c_midi_control_reset_seq
```

9.2.5.109 c_midi_controls_extended

```
const int seq64::c_midi_controls_extended
```

9.2.5.110 g_midi_control_limit

```
int seq64::g_midi_control_limit
```

9.2.5.111 c_status_replace

```
const int seq64::c_status_replace
```

However, we now have to expose them for the Qt5 implementation, until we can entirely reconcile/refactor the Kepler34-based body of code. Note how they specify different bit values, as it they could be masked together to signal multiple functions.

This value signals the "replace" functionality. If this bit is set, then [perform::sequence_playing_toggle\(\)](#) unsets this status and calls [perform::off_sequences\(\)](#), which calls [sequence::set_playing\(false\)](#) for all active sequences.

It works like this:

```
-# The user presses the Replace key, or the MIDI control message for
   c_midi_control_mod_replace is received.
-# This bit is OR'd into perform::m_control_status. This status bit
   is used in perform::sequence_playing_toggle().
   - Called in perform::sequence_key() so that keystrokes in
     the main window toggle patterns in the main window.
   - Called in perform::toggle_other_seqs() to implement
     Shift-click to toggle all other patterns but the one
     clicked.
   - Called in seqmenu::toggle_current_sequence(), called in
     mainwid to implement clicking on a pattern.
   - Also used in MIDI control to toggle patterns 0 to 31,
     offset by the screen-set.
   - perform::sequence_playing_off(), similarly used in MIDI control.
   - perform::sequence_playing_on(), similarly used in MIDI control.
-# When the key is released, this bit is AND'd out of
   perform::m_control_status.
```

Both the MIDI control and the keystroke set the sequence to be "replaced".

9.2.5.112 c_status_snapshot

```
const int seq64::c_status_snapshot
```

By default, [perform::sequence_playing_toggle\(\)](#) calls [sequence::toggle_playing\(\)](#) on the given sequence number, plus what is noted for [c_status_snapshot](#). It works like this:

```
-# The user presses the Snapshot key.
-# This bit is OR'd into perform::m_control_status.
-# The playing state of the patterns is saved by
   perform::save_playing_state().
-# When the key is released, this bit is AND'd out of
   perform::m_control_status.
-# The playing state of the patterns is restored by
   perform::restore_playing_state().
```

9.2.5.113 c_status_queue

```
const int seq64::c_status_queue
```

If this bit is set, then `perform::sequence_playing_toggle()` calls `sequence::toggle_queued()` on the given sequence number. The regular queue key (configurable in File / Options / Keyboard) sets this bit when pressed, and unsets it when released. The keep-queue key sets it, but it is not unset until the regular queue key is pressed and released.

9.2.5.114 c_status_oneshot

```
const int seq64::c_status_oneshot
```

If this bit is set, then `perform::sequence_playing_toggle()` calls `sequence::toggle_oneshot()` on the given sequence number.

9.2.5.115 c_scales_policy

```
const bool seq64::c_scales_policy[c_scale_size][SEQ64_OCTAVE_SIZE]
```

See the following sites for more information:

- <http://method-behind-the-music.com/theory/scalesandkeys/>
- https://en.wikipedia.org/wiki/Heptatonic_scale
- https://en.wikibooks.org/wiki/Music_Theory/Scales_and_Intervals

Note that melodic minor descends in the same way as the natural minor scale, so it descends differently than it ascends. We don't deal with that trick, at all. In the following table, the scales all start with C, but seq24/sequencer64 allow other starting notes (e.g. "keys").

Chromatic	C	C#	D	D#	E	F	F#	G	G#	A	A#	B	Notes, chord
Major	C	.	D	.	E	F	.	G	.	A	.	B	
Minor	C	.	D	Eb	.	F	.	G	Ab	.	Bb	.	
Harmonic Minor	C	.	D	Eb	.	F	.	G	Ab	.	.	B	
Melodic Minor	C	.	D	Eb	.	F	.	G	.	A	.	B	Descending diff.
C Whole Tone	C	.	D	.	E	.	F#	.	G#	.	A#	.	C+7 chord
Blues	C	.	.	Eb	.	F	Gb	G	.	.	Bb	.	
Major Pentatonic	C	.	D	.	E	.	.	G	.	A	.	.	
Minor Pentatonic	C	.	.	Eb	.	F	.	G	.	.	Bb	.	
Octatonic 1	C	.	D	Eb	.	F	Gb	.	Ab	A	.	B	Unimplemented
Octatonic 2	C	Db	.	Eb	E	F	F#	G	.	A	Bb	.	Unimplemented

9.2.5.116 c_scales_transpose_up

```
const int seq64::c_scales_transpose_up[c_scale_size][SEQ64_OCTAVE_SIZE]
```

For example, if we simply add 1 semitone to each note, it remains a minor key, but it is in a different minor key. Using the transpositions in these arrays, the minor key remains the same minor key.

Major	C	.	D	.	E	F	.	G	.	A	.	B
Transpose up	2	0	2	0	1	2	0	2	0	2	0	1
Result up	D	.	E	.	F	G	.	A	.	B	.	C

Minor	C	.	D	D#	.	F	.	G	G#	.	A#	.
Transpose up	2	0	1	2	0	2	0	1	2	0	2	0
Result up	D	.	D#	F	.	G	.	G#	A#	.	C	.

Harmonic minor	C	.	D	Eb	.	F	.	G	Ab	.	.	B
Transpose up	2	.	1	2	.	2	.	1	3	.	.	1
Result up	D	.	Eb	F	.	G	.	Ab	B	.	.	C

Melodic minor	C	.	D	Eb	.	F	.	G	.	A	.	B
Transpose up	2	.	1	2	.	2	.	2	.	2	.	1
Result up	D	.	Eb	F	.	G	.	A	.	B	.	C

C Whole Tone	C	.	D	.	E	.	F#	.	G#	.	A#	.
Transpose up	2	.	2	.	2	.	2	.	2	.	2	.
Result up	D	.	E	.	F#	.	G#	.	A#	.	C	.

Blues	C	.	.	Eb	.	F	Gb	G	.	.	Bb	.
Transpose up	3	.	.	2	.	1	1	3	.	.	2	.
Result up	Eb	.	.	F	.	Gb	G	Bb	.	.	C	.

Major Pentatonic	C	.	D	.	E	.	.	G	.	A	.	.
Transpose up	2	.	2	.	3	.	.	2	.	3	.	.
Result up	D	.	E	.	G	.	.	A	.	C	.	.

Minor Pentatonic	C	.	.	Eb	.	F	.	G	.	.	Bb	.
Transpose up	3	.	.	2	.	2	.	3	.	.	2	.
Result up	Eb	.	.	F	.	G	.	Bb	.	.	C	.

9.2.5.117 c_scales_transpose_dn

```
const int seq64::c_scales_transpose_dn[c_scale_size][SEQ64_OCTAVE_SIZE]
```

Major	C	.	D	.	E	F	.	G	.	A	.	B
Transpose down	1	.	2	.	2	1	.	2	.	2	.	2
Result down	B	.	C	.	D	E	.	F	.	G	.	A

Minor	C	.	D	D#	.	F	.	G	G#	.	A#	.
Transpose down	2	.	2	1	.	2	.	2	1	.	2	.
Result down	A#	.	C	D	.	D#	.	F	G	.	G#	.

Harmonic minor	C . D Eb . F . G Ab . . B
Transpose down	1 . 2 1 . 2 . 2 1 . . 3
Result down	B . C D . Eb . F G . . Ab
Melodic minor	C . D Eb . F . G . A . B
Transpose down	1 . 2 1 . 2 . 2 . 2 . 2
Result down	B . C D . Eb . F . G . A
C whole tone	C . D . E . F# . G# . A# .
Transpose down	2 . 2 . 2 . 2 . 2 . 2 .
Result down	A# . C . D . E . F# . G# .
Blues	C . . Eb . F Gb G . . Bb .
Transpose down	2 . . 3 . 2 1 1 . . 3 .
Result down	Bb . . C . Eb F Gb . . G .
Major Pentatonic	C . D . E . . G . A . .
Transpose down	3 . 2 . 2 . . 3 . 2 . .
Result down	A . C . D . . E . G . .
Minor Pentatonic	C . . Eb . F . G . . Bb .
Transpose down	2 . . 3 . 2 . 2 . . 3 .
Result down	Bb . . C . Eb . F . . G .

9.2.5.118 c_scales_text

```
const char seq64::c_scales_text[c_scale_size][20]
```

9.2.5.119 c_key_text

```
const char seq64::c_key_text[SEQ64_OCTAVE_SIZE][4]
```

9.2.5.120 c_interval_text

```
const char seq64::c_interval_text[16][4]
```

9.2.5.121 c_chord_text

```
const char seq64::c_chord_text[8][6]
```

However, we have not seen this menu in the GUI! Ah, it only appears if the user has selected a musical scale like Major or Minor.

9.2.5.122 c_chord_number

```
const int seq64::c_chord_number
```

The chord-number is a count of the number of entries in c_chord_table_text. Will never change, luckily.

9.2.5.123 c_chord_table_text

```
const char seq64::c_chord_table_text[c_chord_number][12]
```

These chords appear in the sequence-editor chord-button dropdown menu. The longest string is 11 characters, and we add one for the null terminator. A good case for using std::string here. :-)

9.2.5.124 c_chord_size

```
const int seq64::c_chord_size
```

9.2.5.125 c_chord_table

```
const int seq64::c_chord_table[c_chord_number][c_chord_size]
```

These values indicate the note offsets needed for a particular kind of chord. 0 means no offset, and a -1 ends the list of note offsets for the chord.

9.2.5.126 c_max_instruments

```
const int seq64::c_max_instruments
```

With a value of 64, this is more of a sanity-check than a realistic number of instruments defined by a user.

9.2.5.127 c_max_busses

```
const int seq64::c_max_busses
```

9.2.5.128 versiontext

```
const std::string seq64::versiontext [static]
```

This value ultimately comes from the configure.ac script.

This was too redundant:

```
SEQ64_PACKAGE " " SEQ64_VERSION " (" SEQ64_GIT_VERSION ") " DATE "\n"
```

This is out-of-date:

```
SEQ64_PACKAGE " " SEQ64_GIT_VERSION " " DATE "\n";
```


9.2.5.129 long_options

```
struct option seq64::long_options[] [static]
```

Note the terminating null structure..

9.2.5.130 s_arg_list

```
const std::string seq64::s_arg_list [static]
```

The following string keeps track of the characters used so far. An 'x' means the character is used; an 'o' means it is used for the legacy spelling of the option, which uses underscores instead of hyphens. An 'a' indicates we could repurpose the key with minimal impact. An asterisk indicates the option is reserved for application-specific options. Currently we will use it for options like "daemonize" in the seq64cli application.

```
0123456789 @AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz#
ooooooooo oxxxxxx x  xx  xx xxx xxxxxxxx *xx xxxxx xxxxx  x  x
```

Previous arg-list, items missing! "ChVH:lRrb:q:Lni:jJmaAM:pPusSU:x:"

* Also note that 'o' options argument cannot be included here due to issues involving parse_o_options(), but it is *reserved* here, without the argument indicator.

9.2.5.131 s_help_1a

```
const char* const seq64::s_help_1a [static]
```

9.2.5.132 s_help_1b

```
const char* const seq64::s_help_1b [static]
```

9.2.5.133 s_help_2

```
const char* const seq64::s_help_2 [static]
```

9.2.5.134 s_help_3

```
const char* const seq64::s_help_3 [static]
```

9.2.5.135 s_help_4

```
const char* const seq64::s_help_4 [static]
```

9.2.5.136 s_help_5

```
const char* const seq64::s_help_5 [static]
```

9.2.5.137 s_bitness

```
const std::string seq64::s_bitness [static]
```

9.2.5.138 s_status_pairs

```
jack_status_pair_t seq64::s_status_pairs[]
```

Terminated by a 0 value and an empty string.

9.2.5.139 s_character_mapping

```
struct charpair_t seq64::s_character_mapping[]
```

9.2.5.140 s_global_lash_driver

```
lash* seq64::s_global_lash_driver [static]
```

It is actually hidden in this module now, so that a function can be used in its place.

Like the font renderer, this item was once created in the main module, sequencer64.cpp. Now we make it a safer, more fool-proof, function. However, unlike the font-render, which always exists, the LASH driver is conditional, and might not be wanted. Therefore, we cannot return a reference, because there's no such thing as a null reference in C++. We have to return a pointer.

9.2.5.141 g_rc_settings

```
rc_settings seq64::g_rc_settings [static]
```

9.2.5.142 g_user_settings

```
user_settings seq64::g_user_settings [static]
```

9.2.5.143 s_jitter_amount

```
const int seq64::s_jitter_amount [static]
```

9.2.5.144 gs_mainwid_pointer

```
mainwid* seq64::gs_mainwid_pointer [static]
```

We have decided that passing along a mainwnd reference among a number of constructors is too much and actually harder to understand and more error prone. This value is set at the end of the mainwnd constructor, but only the first time that constructor is called.

9.2.5.145 gs_perfeddit_pointer_0

```
perfeddit* seq64::gs_perfeddit_pointer_0 [static]
```

9.2.5.146 gs_perfeddit_pointer_1

```
perfeddit* seq64::gs_perfeddit_pointer_1 [static]
```

9.2.5.147 s_handlesize

```
const long seq64::s_handlesize [static]
```


Chapter 10

Data Structure Documentation

10.1 seq64::automutex Class Reference

Provides a mutex that locks automatically when created, and unlocks when destroyed.

Public Member Functions

- [automutex](#) ([mutex](#) &my_mutex)
Principal constructor gets a reference to a mutex parameter, and then locks the mutex.
- [~automutex](#) ()
The destructor unlocks the mutex.

Private Member Functions

- [automutex](#) ()
- [automutex](#) (const [automutex](#) &)
- [automutex](#) & [operator=](#) (const [automutex](#) &)

Private Attributes

- [mutex](#) & [m_safety_mutex](#)
Provides the mutex reference to be used for locking.

10.1.1 Detailed Description

This has a couple of benefits. First, it is threadsafe in the face of exception handling. Secondly, it can be done with just one line of code.

10.1.2 Constructor & Destructor Documentation

10.1.2.1 automutex() [1/3]

```
seq64::automutex::automutex ( ) [private]
```

10.1.2.2 automutex() [2/3]

```
seq64::automutex::automutex (
    const automutex & ) [private]
```

10.1.2.3 automutex() [3/3]

```
seq64::automutex::automutex (
    mutex & my_mutex ) [inline]
```

Parameters

<i>my_mutex</i>	The caller's mutex to be used for locking.
-----------------	--

10.1.2.4 ~automutex()

```
seq64::automutex::~~automutex ( ) [inline]
```

10.1.3 Member Function Documentation**10.1.3.1 operator=()**

```
automutex& seq64::automutex::operator= (
    const automutex & ) [private]
```

10.1.4 Field Documentation**10.1.4.1 m_safety_mutex**

```
mutex& seq64::automutex::m_safety_mutex [private]
```

10.2 seq64::busarray Class Reference

Holds a number of businfo objects.

Public Member Functions

- [busarray](#) ()
A new class to hold a number of MIDI busses and flags for more controlled access than using arrays of booleans and pointers.
- [~busarray](#) ()
Removes components from the container.
- [bool add](#) ([midibus](#) *bus, [clock_e](#) clock)
Creates and adds a new midibus object to the list.
- [bool add](#) ([midibus](#) *bus, [bool](#) inputing)
Creates and adds a new midibus object to the list.
- [bool initialize](#) ()
Initializes all busses.
- [int count](#) () const
- [midibus](#) * [bus](#) ([bussbyte](#) b)
- [void start](#) ()
Starts all of the busses; used for output busses only, but no check is made at present.
- [void stop](#) ()
Stops all of the busses; used for output busses only, but no check is made at present.
- [void continue_from](#) ([midipulse](#) tick)
Continues from the given tick for all of the busses; used for output busses only.
- [void init_clock](#) ([midipulse](#) tick)
Initializes the clocking at the given tick for all of the busses; used for output busses only.
- [void clock](#) ([midipulse](#) tick)
Clocks at the given tick for all of the busses; used for output busses only.
- [void sysex](#) ([event](#) *ev)
Handles SysEx events; used for output busses.
- [void play](#) ([bussbyte](#) bus, [event](#) *e24, [midibyte](#) channel)
Plays an event, if the bus is proper.
- [bool set_clock](#) ([bussbyte](#) bus, [clock_e](#) clocktype)
Sets the clock type for the given bus, usually the output buss.
- [void set_all_clocks](#) ()
Sets the clock type for all busses, usually the output buss.
- [clock_e get_clock](#) ([bussbyte](#) bus)
Gets the clock type for the given bus, usually the output buss.
- [std::string get_midi_bus_name](#) ([int](#) bus)
Get the MIDI output buss name (i.e.
- [void print](#) () const
Print some information about the available MIDI input or output busses.
- [void port_exit](#) ([int](#) client, [int](#) port)
Turn off the given port for the given client.
- [bool set_input](#) ([bussbyte](#) bus, [bool](#) inputing)
Set the status of the given input buss, if a legal buss number.
- [void set_all_inputs](#) ()
Set the status of all input busses.
- [bool get_input](#) ([bussbyte](#) bus)

- *Get the input for the given (legal) buss number.*
- bool `is_system_port (bussbyte bus)`
Get the system-port status for the given (legal) buss number.
- int `poll_for_midi ()`
Initiate a poll() on the existing poll descriptors.
- bool `get_midi_event (event *inev)`
Gets the first MIDI event in finds on an input bus.
- int `replacement_port (int bus, int port)`
Provides a function to use in api_port_start(), to determine if the port is to be a "replacement" port.

Private Attributes

- std::vector< `businfo` > `m_container`
The full set of businfo objects, only some of which will actually be used.

10.2.1 Constructor & Destructor Documentation

10.2.1.1 `busarray()`

```
seq64::busarray::busarray ( )
```

10.2.1.2 `~busarray()`

```
seq64::busarray::~~busarray ( )
```

Question However, now that we swap containers, we cannot call this functionality, because it deletes the bus's midibus pointer and nullifies it. But we do call it, and it seems to work.

10.2.2 Member Function Documentation

10.2.2.1 `add()` [1/2]

```
bool seq64::busarray::add (
    midibus * bus,
    clock_e clock )
```

Then the clock value is set. This function is meant for output ports.

We need to belay the initialization until later, when we know the configured clock settings for the output ports. So initialization has been removed from the constructor and moved to the `initialize()` function.

Parameters

<i>bus</i>	The midibus to be hooked into the array of busses.
<i>clock</i>	The clocking value for the bus.

Returns

Returns true if the bus was added successfully, though, really, it cannot fail.

10.2.2.2 add() [2/2]

```
bool seq64::busarray::add (
    midibus * bus,
    bool inputing )
```

Then the inputing value is set. This function is meant for input ports.

We need to delay the initialization until later, when we know the configured inputing settings for the input ports. So initialization has been removed from the constructor and moved to the [initialize\(\)](#) function. However, now we know the configured status and can apply it right away.

Parameters

<i>bus</i>	The midibus to be hooked into the array of busses.
<i>inputing</i>	The input flag value for the bus. If true, this value indicates that the user has selected this bus to be the input MIDI bus.

Returns

Returns true if the bus was added successfully, though, really, it cannot fail.

10.2.2.3 initialize()

```
bool seq64::busarray::initialize ( )
```

Not sure we need this function.

Returns

Returns true if all busses initialized successfully.

10.2.2.4 count()

```
int seq64::busarray::count ( ) const [inline]
```

10.2.2.5 bus()

```
midibus* seq64::busarray::bus (
    bussbyte b ) [inline]
```

10.2.2.6 start()

```
void seq64::busarray::start ( )
```

10.2.2.7 stop()

```
void seq64::busarray::stop ( )
```

10.2.2.8 continue_from()

```
void seq64::busarray::continue_from (
    midipulse tick )
```

Parameters

<i>tick</i>	Provides the tick value for all busses to continue from.
-------------	--

10.2.2.9 init_clock()

```
void seq64::busarray::init_clock (
    midipulse tick )
```

Parameters

<i>tick</i>	Provides the tick value for all busses use as the clock tick.
-------------	---

10.2.2.10 clock()

```
void seq64::busarray::clock (
    midipulse tick )
```

Parameters

<i>tick</i>	Provides the tick value for all busses use as the clock tick.
-------------	---

10.2.2.11 sysex()

```
void seq64::busarray::sysex (
    event * ev )
```

Parameters

<i>ev</i>	Provides the SysEx event to handle.
-----------	-------------------------------------

10.2.2.12 play()

```
void seq64::busarray::play (
    bussbyte bus,
    event * e24,
    midibyte channel )
```

Parameters

<i>bus</i>	The MIDI buss on which to play the event. The buss number must be valid (in range) and the bus must be active.
<i>e24</i>	A pointer to the event to be played.
<i>channel</i>	The MIDI channel on which to play the event. Sequencer64 controls the actual channel of playback, no matter what the channel specified in the event.

10.2.2.13 set_clock()

```
bool seq64::busarray::set_clock (
    bussbyte bus,
    clock_e clocktype )
```

This code is a bit more restrictive than the original code in [mastermidibus::set_clock\(\)](#).

Parameters

<i>bus</i>	The MIDI bus for which the clock is to be set.
<i>clocktype</i>	Provides the type of clocking for the buss.

Returns

Returns true if the change was made.

10.2.2.14 set_all_clocks()

```
void seq64::busarray::set_all_clocks ( )
```

Note that the settings to apply are added when the [add\(\)](#) call is made.

10.2.2.15 get_clock()

```
clock_e seq64::busarray::get_clock (
    bussbyte bus )
```

Parameters

<i>bus</i>	The MIDI bus for which the clock is to be set.
------------	--

Returns

Returns the clock value set for the desired buss. If the buss is invalid, `e_clock_off` is returned. If the buss is not active, we still get the existing clock value. The theory here is that we don't want to junk the current clock value; it could alter what was read from the "rc" file.

10.2.2.16 get_midi_bus_name()

```
std::string seq64::busarray::get_midi_bus_name (
    int bus )
```

the full display name) for the given (legal) buss number.

This function adds the retrieval of client and port numbers that are not needed in the portmidi implementation, but seem generally useful to support in all implementations. It's main use is to display the full portname in one of two forms:

```
- "[0] 0:0 clientname:portname"
- "[0] 0:0 portname"
```

The second version is chosen if "clientname" is already included in the port name, as many MIDI clients do that. However, the name gets modified to reflect the remote system port to which it will connect.

Parameters

<i>bus</i>	Provides the output buss number. Checked before usage. Actually should now be an index number
------------	---

Returns

Returns the buss name as a standard C++ string, truncated to 80-1 characters. Also contains an indication that the buss is disconnected or unconnected. If the buss number is illegal, this string is empty.

10.2.2.17 print()

```
void seq64::busarray::print ( ) const
```

10.2.2.18 port_exit()

```
void seq64::busarray::port_exit (
    int client,
    int port )
```

Both the busses for the given client are stopped: that is, set to inactive.

This function is called by `api_get_midi_event()` when the ALSA event `SND_SEQ_EVENT_PORT_EXIT` is received. Since `port_exit()` has no direct API-specific code in it, we do not need to create a virtual `api_port_exit()` function to implement the port-exit event.

Parameters

<i>client</i>	The client to be matched and acted on. This value is actually an ALSA concept.
<i>port</i>	The port to be acted on. Both parameter must be matched before the buss is made inactive. This value is actually an ALSA concept.

10.2.2.19 set_input()

```
bool seq64::busarray::set_input (
    bussbyte bus,
    bool inputing )
```

There's currently no implementation-specific API function called directly here. What happens is that `midibase::set_input()` uses the *inputing* parameter to decide whether to call `init_in()` or `deinit_in()`, and these functions ultimately lead to an API specific called.

Note that the call to `midibase::set_input()` will set its `m_inputing` flag, and then call `init_in()` or `deinit_in()` if that flag changed. This change is important, so we have to call `midibase::set_input()` first. Then the call to `businfo::init_input()` will set that flag again (plus another flag). A bit confusing in sequence and in function naming.

This function should be used only for the input busarray, obviously.

Threadsafe

Parameters

<i>bus</i>	Provides the buss number.
<i>inputing</i>	True if the input bus will be inputting MIDI data.

Returns

Returns true if the buss number is valid and was active, and so could be set.

10.2.2.20 `set_all_inputs()`

```
void seq64::busarray::set_all_inputs ( )
```

There's no implementation-specific API function here. This function should be used only for the input busarray, obviously. Note that the input settings used here were stored when the [add\(\)](#) function was called. They can be changed by the user via the Options / MIDI Input tab.

10.2.2.21 `get_input()`

```
bool seq64::busarray::get_input (
    bussbyte bus )
```

There's currently no implementation-specific API function here.

Parameters

<i>bus</i>	Provides the buss number.
------------	---------------------------

Returns

If the buss is a system buss, always returns true. Otherwise, if the buss is inactive, returns false. Otherwise, the buss's [get_input\(\)](#) status is returned.

10.2.2.22 `is_system_port()`

```
bool seq64::busarray::is_system_port (
    bussbyte bus )
```

Parameters

<i>bus</i>	Provides the buss number.
------------	---------------------------

Returns

Returns the selected buss's is-system-port status. If the buss number is out of range, then false is returned.

10.2.2.23 poll_for_midi()

```
int seq64::busarray::poll_for_midi ( )
```

This is a primitive poll, which exits when some data is obtained. It also applies only to the input busses.

One issue is that we have no way of knowing here which MIDI input device has MIDI input events waiting. Should we randomize the order of polling in order to avoid starving some input devices?

Returns

Returns the number of MIDI events detected on one of the busses. Note that this is no longer a boolean value.

10.2.2.24 get_midi_event()

```
bool seq64::busarray::get_midi_event (
    event * inev )
```

Note that this function risks starving the second input device if more than one is enabled in Sequencer64. We will figure that one out later.

Parameters

<i>inev</i>	A pointer to the event to be modified by incoming data, if any.
-------------	---

Returns

Returns true if an event's data was copied into the event pointer.

10.2.2.25 replacement_port()

```
int seq64::busarray::replacement_port (
    int bus,
    int port )
```

This function is meant only for the output buss (so far).

Still need to determine exactly what this function needs to do.

Parameters

<i>bus</i>	The buss to be affected.
<i>port</i>	The prot to be affected.

Returns

Returns -1 if no matching port is found, otherwise it returns the replacement-port number.

10.2.3 Field Documentation

10.2.3.1 m_container

```
std::vector<businfo> seq64::busarray::m_container [private]
```

10.3 seq64::businfo Class Reference

A new class to consolidate a number of bus-related arrays into one array.

Public Member Functions

- [businfo](#) ()
A new class to consolidate a number of bus-related arrays into one array.
- [businfo](#) ([midibus](#) *[bus](#))
Principal constructor.
- [businfo](#) (const [businfo](#) &rhs)
Copy constructor.
- [~businfo](#) ()
We can't destroy the bus pointer.
- void [remove](#) ()
Deletes and nullifies the m_bus pointer.
- const [midibus](#) * [bus](#) () const
'Getter' function for member m_bus pointer, const version
- [midibus](#) * [bus](#) ()
'Getter' function for member m_bus pointer
- bool [active](#) () const
'Getter' function for member m_active
- bool [initialize](#) ()
This function is called when the businfo object is added to the busarray.
- bool [initialized](#) () const
'Getter' function for member m_initialized
- [clock_e](#) [init_clock](#) () const
'Getter' function for member m_init_clock
- bool [init_input](#) () const

- *'Getter' function for member m_init_input*
- void `bus` (`midibus *b`)
- *'Setter' function for member m_bus*
- void `activate` ()
- *'Setter' function for member m_active and m_initialized*
- void `deactivate` ()
- *'Setter' function for member m_active and m_initialized*
- void `init_clock` (`clock_e` clocktype)
- *'Setter' function for member m_init_clock and bus clock*
- void `init_input` (bool flag)
- *'Setter' function for member m_init_input and bus input*

Private Member Functions

- void `start` ()
 - void `stop` ()
 - void `continue_from` (`midipulse` tick)
 - void `init_clock` (`midipulse` tick)
 - void `clock` (`midipulse` tick)
 - void `sysex` (`event *ev`)
 - void `print` () const
- Print some information about the MIDI bus.*

Private Attributes

- `midibus * m_bus`
- *Points to an existing midibus object.*
- bool `m_active`
- *Indicates if the existing bus is active.*
- bool `m_initialized`
- *Indicates if the existing bus is initialized.*
- `clock_e m_init_clock`
- *Clock initialization.*
- bool `m_init_input`
- *Input initialization?*

Friends

- class `busarray`

10.3.1 Detailed Description

There will be in input instance and an output instance of this object contained by mastermidibus.

10.3.2 Constructor & Destructor Documentation

10.3.2.1 businfo() [1/3]

```
seq64::businfo::businfo ( )
```

There will be in input instance and an output instance of this object contained by mastermidibus.

10.3.2.2 businfo() [2/3]

```
seq64::businfo::businfo (
    midibus * bus )
```

is_input_port():

Indicates if the midibus represents an input port (true) versus an output port (false).

is_virtual_port():

Indicates if the midibus represents a virtual port (true) versus a normal port (false).

Parameters

<i>bus</i>	Provides a pointer to the MIDI buss object to be represented by this object.
------------	--

10.3.2.3 businfo() [3/3]

```
seq64::businfo::businfo (
    const businfo & rhs )
```

Currently it does not replicate the pointed-to object.

Parameters

<i>rhs</i>	The source object to be copied.
------------	---------------------------------

10.3.2.4 ~businfo()

```
seq64::businfo::~~businfo ( ) [inline]
```

10.3.3 Member Function Documentation

10.3.3.1 remove()

```
void seq64::businfo::remove ( ) [inline]
```

10.3.3.2 bus() [1/3]

```
const midibus* seq64::businfo::bus ( ) const [inline]
```

10.3.3.3 bus() [2/3]

```
midibus* seq64::businfo::bus ( ) [inline]
```

10.3.3.4 active()

```
bool seq64::businfo::active ( ) const [inline]
```

10.3.3.5 initialize()

```
bool seq64::businfo::initialize ( )
```

It relies on the [perform::launch\(\)](#) function to actually [activate\(\)](#) all of the ports that have been flagged as "activated" here.

is_input_port():

Indicates if the midibus represents an input port (true) versus an output port (false). The way the mastermidibus currently works, it creates the API MIDI input objects there, so it does not need to be done here. This falls under the heading of "tricky code".

is_virtual_port():

Indicates if the midibus represents a manual/virtual port (true) versus a normal port (false).

The rules for port initialization follow those of seq24 for MIDI busses:

- Manual (virtual) input and output ports always get their init functions called. They are unconditionally marked as "active" and "initialized".
- Normal output ports are marked as "active" and "initialized" if `init_out()` succeeds.
- Normal input ports don't have `init_in()` called, but are marked as "active" and "initialized" anyway. The settings from the "rc" file determine which inputs will operate.

Returns

Returns true if the buss is value, and it could be initialized (as an output port or a virtual output port).

10.3.3.6 initialized()

```
bool seq64::businfo::initialized ( ) const [inline]
```

10.3.3.7 init_clock() [1/3]

```
clock_e seq64::businfo::init_clock ( ) const [inline]
```

10.3.3.8 init_input() [1/2]

```
bool seq64::businfo::init_input ( ) const [inline]
```

10.3.3.9 bus() [3/3]

```
void seq64::businfo::bus (
    midibus * b ) [inline]
```

10.3.3.10 activate()

```
void seq64::businfo::activate ( ) [inline]
```

10.3.3.11 deactivate()

```
void seq64::businfo::deactivate ( ) [inline]
```

10.3.3.12 init_clock() [2/3]

```
void seq64::businfo::init_clock (
    clock_e clocktype ) [inline]
```

10.3.3.13 `init_input()` [2/2]

```
void seq64::businfo::init_input (
    bool flag ) [inline]
```

10.3.3.14 `start()`

```
void seq64::businfo::start ( ) [inline], [private]
```

10.3.3.15 `stop()`

```
void seq64::businfo::stop ( ) [inline], [private]
```

10.3.3.16 `continue_from()`

```
void seq64::businfo::continue_from (
    midipulse tick ) [inline], [private]
```

10.3.3.17 `init_clock()` [3/3]

```
void seq64::businfo::init_clock (
    midipulse tick ) [inline], [private]
```

10.3.3.18 `clock()`

```
void seq64::businfo::clock (
    midipulse tick ) [inline], [private]
```

10.3.3.19 `sysex()`

```
void seq64::businfo::sysex (
    event * ev ) [inline], [private]
```

10.3.3.20 print()

```
void seq64::businfo::print ( ) const [private]
```

10.3.4 Friends And Related Function Documentation

10.3.4.1 busarray

```
friend class busarray [friend]
```

10.3.5 Field Documentation

10.3.5.1 m_bus

```
midibus* seq64::businfo::m_bus [private]
```

10.3.5.2 m_active

```
bool seq64::businfo::m_active [private]
```

10.3.5.3 m_initialized

```
bool seq64::businfo::m_initialized [private]
```

10.3.5.4 m_init_clock

```
clock_e seq64::businfo::m_init_clock [private]
```

10.3.5.5 m_init_input

```
bool seq64::businfo::m_init_input [private]
```

10.4 seq64::click Class Reference

Encapsulates any possible mouse click.

Public Member Functions

- [click](#) ()
The constructor for class click.
- [click](#) (int [x](#), int [y](#), int [button](#)=SEQ64_CLICK_BUTTON_LEFT, bool [press](#)=true, [seq_modifier_t](#) [modkey](#)=SEQ64_NO_MASK)
Principal constructor for class click.
- [click](#) (const [click](#) &[rhs](#))
Provides a stock copy constructor.
- [click](#) & [operator](#)= (const [click](#) &[rhs](#))
Provides a stock principal assignment operator.
- bool [is_press](#) () const
'Getter' function for member [m_is_press](#)
- bool [is_left](#) () const
'Getter' function for member [m_button](#) to test for the left button.
- bool [is_middle](#) () const
'Getter' function for member [m_button](#) to test for the middle button.
- bool [is_right](#) () const
'Getter' function for member [m_button](#) to test for the right button.
- int [x](#) () const
'Getter' function for member [m_x](#)
- int [y](#) () const
'Getter' function for member [m_y](#)
- int [button](#) () const
'Getter' function for member [m_button](#)
- [seq_modifier_t](#) [modifier](#) () const
'Getter' function for member [m_modifier](#)
- bool [mod_control](#) () const
'Getter' function for member [m_modifier](#) tested for Ctrl key.
- bool [mod_control_shift](#) () const
'Getter' function for member [m_modifier](#) tested for Ctrl and Shift key.
- bool [mod_super](#) () const
'Getter' function for member [m_modifier](#) tested for Mod4/Super/Windows key.

Private Attributes

- bool [m_is_press](#)
Determines if the click was a press or a release event.
- int [m_x](#)
The x-coordinate of the click.
- int [m_y](#)
The y-coordinate of the click.
- int [m_button](#)
The button that was pressed or released.
- [seq_modifier_t](#) [m_modifier](#)
The optional modifier value.

10.4.1 Detailed Description

Useful in passing more generic events to non-GUI classes.

10.4.2 Constructor & Destructor Documentation

10.4.2.1 `click()` [1/3]

```
seq64::click::click ( )
```

Sets all members to false, zero, or the lowest good value.

10.4.2.2 `click()` [2/3]

```
seq64::click::click (
    int x,
    int y,
    int button = SEQ64_CLICK_BUTTON_LEFT,
    bool press = true,
    seq_modifier_t modkey = SEQ64_NO_MASK )
```

This function is the only way to set value for the click members (other than the copy constructor and principal assignment operator).

Parameters

<i>x</i>	The putative x value of the button click.
<i>y</i>	The putative y value of the button click.
<i>button</i>	The value of the button that was clicked, set to 1, 2, or 3.
<i>press</i>	Set to true if the event was a button press, false if it was a button release.
<i>modkey</i>	Indicates which modifier key (such as Ctrl or Alt), if any, was pressed at the same time as the click action.

10.4.2.3 `click()` [3/3]

```
seq64::click::click (
    const click & rhs )
```

It is nice to be explicit about these kinds of functions, even if it gets tedious.

Parameters

<i>rhs</i>	Provies the source object to be copied.
------------	---

10.4.3 Member Function Documentation

10.4.3.1 operator=()

```
click & seq64::click::operator= (
    const click & rhs )
```

It is nice to be explicit about these kinds of functions, even if it gets tedious.

Parameters

<i>rhs</i>	Provies the source object to be assigned from. The assignment is not made if "this" has the same address as this parameter.
------------	---

Returns

Returns a reference to self for usage in a string of assignments.

10.4.3.2 is_press()

```
bool seq64::click::is_press ( ) const [inline]
```

10.4.3.3 is_left()

```
bool seq64::click::is_left ( ) const [inline]
```

10.4.3.4 is_middle()

```
bool seq64::click::is_middle ( ) const [inline]
```

10.4.3.5 is_right()

```
bool seq64::click::is_right ( ) const [inline]
```

10.4.3.6 x()

```
int seq64::click::x ( ) const [inline]
```

10.4.3.7 y()

```
int seq64::click::y ( ) const [inline]
```

10.4.3.8 button()

```
int seq64::click::button ( ) const [inline]
```

10.4.3.9 modifier()

```
seq_modifier_t seq64::click::modifier ( ) const [inline]
```

10.4.3.10 mod_control()

```
bool seq64::click::mod_control ( ) const [inline]
```

10.4.3.11 mod_control_shift()

```
bool seq64::click::mod_control_shift ( ) const [inline]
```

10.4.3.12 mod_super()

```
bool seq64::click::mod_super ( ) const [inline]
```

10.4.4 Field Documentation

10.4.4.1 m_is_press

```
bool seq64::click::m_is_press [private]
```

10.4.4.2 m_x

```
int seq64::click::m_x [private]
```

0 is the left-most coordinate.

10.4.4.3 m_y

```
int seq64::click::m_y [private]
```

0 is the top-most coordinate.

10.4.4.4 m_button

```
int seq64::click::m_button [private]
```

Left is 1, middle is 2, and right is 3. These numbers are defined via macros, and are Linux-specific and Gtk-specific.

10.4.4.5 m_modifier

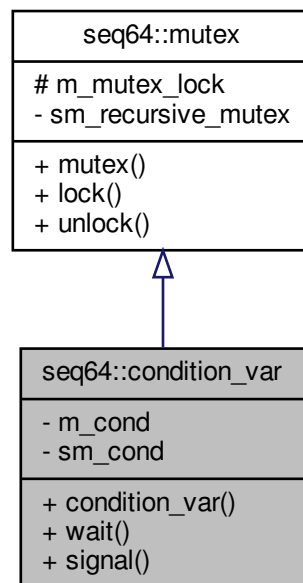
```
seq_modifier_t seq64::click::m_modifier [private]
```

Note that SEQ64_NO_MASK is our word for 0, meaning "no modifier".

10.5 seq64::condition_var Class Reference

A mutex works best in conjunction with a condition variable.

Inheritance diagram for seq64::condition_var:



Public Member Functions

- `condition_var ()`
Initialize the condition variable with the global variable.
- `void wait ()`
Waits for the condition variable.
- `void signal ()`
Signals the condition variable.

Private Attributes

- `pthread_cond_t m_cond`
Provides a class-specific condition variable.

Static Private Attributes

- `static const pthread_cond_t sm_cond`
Provides a "global" condition variable.

Additional Inherited Members

10.5.1 Detailed Description

Therefore this class derives from the mutex class. A "has-a" relationship might be more logical than this "is-a" relationship.

10.5.2 Constructor & Destructor Documentation

10.5.2.1 condition_var()

```
seq64::condition_var::condition_var ( )
```

10.5.3 Member Function Documentation

10.5.3.1 wait()

```
void seq64::condition_var::wait ( )
```

10.5.3.2 signal()

```
void seq64::condition_var::signal ( )
```

10.5.4 Field Documentation

10.5.4.1 sm_cond

```
const pthread_cond_t seq64::condition_var::sm_cond [static], [private]
```

Define the static condition variable used by all mutex locks.

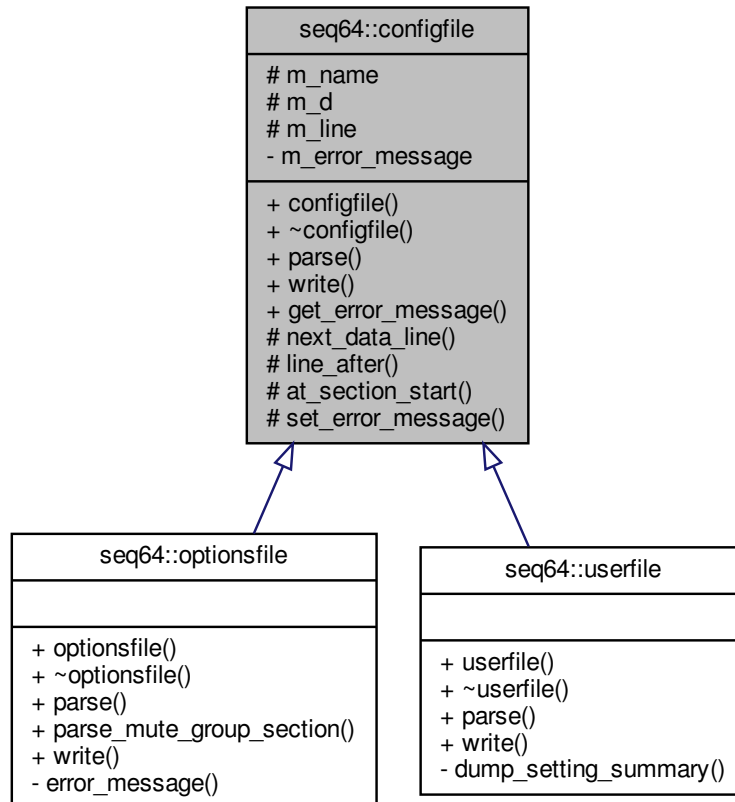
10.5.4.2 m_cond

```
pthread_cond_t seq64::condition_var::m_cond [private]
```

10.6 seq64::configfile Class Reference

This class is the abstract base class for optionsfile and userfile.

Inheritance diagram for seq64::configfile:



Public Member Functions

- [configfile](#) (const std::string &name)
Provides the string constructor for a configuration file.
- virtual [~configfile](#) ()
A rote destructor needed for a base class.
- virtual bool [parse](#) ([perform](#) &perf)=0
- virtual bool [write](#) (const [perform](#) &perf)=0
- const std::string & [get_error_message](#) () const

Protected Member Functions

- bool [next_data_line](#) (std::ifstream &file)
Gets the next line of data from an input stream.

- bool [line_after](#) (std::ifstream &file, const std::string &tag)
This function gets a specific line of text, specified as a tag.
- bool [at_section_start](#) () const
Sometimes we need to know if there are new data lines at the end of an existing section.
- void [set_error_message](#) (const std::string &msg)

Protected Attributes

- std::string [m_name](#)
Provides the name of the configuration file.
- char * [m_d](#)
Points to an allocated buffer that holds the data for the configuration file.
- char [m_line](#) [SEQ64_LINE_MAX]
The current line of text being processed.

Private Attributes

- std::string [m_error_message](#)
Holds the last error message, if any.

10.6.1 Constructor & Destructor Documentation

10.6.1.1 configfile()

```
seq64::configfile::configfile (
    const std::string & name )
```

Parameters

<i>name</i>	The name of the configuration file.
-------------	-------------------------------------

10.6.1.2 ~configfile()

```
virtual seq64::configfile::~~configfile ( ) [inline], [virtual]
```

10.6.2 Member Function Documentation

10.6.2.1 next_data_line()

```
bool seq64::configfile::next_data_line (
    std::ifstream & file ) [protected]
```

If the line starts with a number-sign, a space (!), or a null, it is skipped, to try the next line. This occurs until an EOF is encountered.

Member m_line is a "global" return value.

Parameters

<i>file</i>	Points to an input stream. We converted this item to a reference; pointers can be subject to problems. For example, what if someone passes a null pointer?
-------------	--

Returns

Returns true if a presumed data line was found. False is returned if not found before an EOF or a section marker ("[" is found. This is a new (ca 2016-02-14) feature of this function, to assist in adding new data to the file.

10.6.2.2 line_after()

```
bool seq64::configfile::line_after (
    std::ifstream & file,
    const std::string & tag ) [protected]
```

Then it gets the next non-blank line (i.e. data line) after that.

This function always starts from the beginning of the file. Therefore, it can handle reading Sequencer64 configuration files that have had their tagged sections arranged in a different order. This feature makes the configuration file a little more robust against errors.

Parameters

<i>file</i>	Points to the input file stream.
<i>tag</i>	Provides a tag to be found. Lines are read until a match occurs with this tag. Normally, the tag is a section marker, such as "[user-interface]". Best to assume an exact match is needed.

Returns

Returns true if the tag was found. Otherwise, false is returned.

10.6.2.3 at_section_start()

```
bool seq64::configfile::at_section_start ( ) const [inline], [protected]
```

One clue that there is not is that we're at the next section marker. This function tests for that condition.

Returns

Returns true if `m_line[0]` is the left-bracket character.

10.6.2.4 parse()

```
virtual bool seq64::configfile::parse (
    perform & perf ) [pure virtual]
```

Implemented in [seq64::userfile](#), and [seq64::optionsfile](#).

10.6.2.5 write()

```
virtual bool seq64::configfile::write (
    const perform & perf ) [pure virtual]
```

Implemented in [seq64::optionsfile](#), and [seq64::userfile](#).

10.6.2.6 get_error_message()

```
const std::string& seq64::configfile::get_error_message ( ) const [inline]
```

10.6.2.7 set_error_message()

```
void seq64::configfile::set_error_message (
    const std::string & msg ) [inline], [protected]
```

10.6.3 Field Documentation

10.6.3.1 m_error_message

```
std::string seq64::configfile::m_error_message [private]
```

Not a 100% foolproof yet.

10.6.3.2 m_name

```
std::string seq64::configfile::m_name [protected]
```

10.6.3.3 m_d

```
char* seq64::configfile::m_d [protected]
```

10.6.3.4 m_line

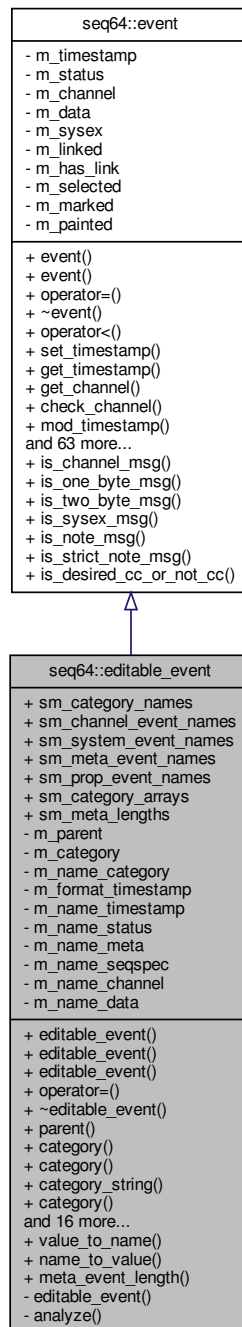
```
char seq64::configfile::m_line[SEQ64_LINE_MAX] [protected]
```

This member receives an input line, and so needs to be a character buffer.

10.7 seq64::editable_event Class Reference

Provides for the management of MIDI editable events.

Inheritance diagram for seq64::editable_event:



Data Structures

- struct [meta_length_t](#)

Provides a type that contains the pair of values needed to get the Meta event's data length.

- struct [name_value_t](#)

Provides a type that contains the pair of values needed for the various lookup maps that are needed to manage editable events.

Public Types

- enum `category_t` {
`category_name`,
`category_channel_message`,
`category_system_message`,
`category_meta_event`,
`category_prop_event` }

These values determine the major kind of event, which determines what types of events are possible for this editable event object.

- enum `timestamp_format_t` {
`timestamp_measures`,
`timestamp_time`,
`timestamp_pulses` }

Provides a code to indicate the desired timestamp format.

Public Member Functions

- `editable_event` (const `editable_events` &`parent`)

Principal constructor.

- `editable_event` (const `editable_events` &`parent`, const `event` &`ev`)

Event constructor.

- `editable_event` (const `editable_event` &`rhs`)

This copy constructor initializes most of the class members.

- `editable_event` & `operator=` (const `editable_event` &`rhs`)

- virtual `~editable_event` ()

This destructor current is a rote virtual function override.

- const `editable_events` & `parent` () const

'Getter' function for member `m_parent`

- `category_t` `category` () const

'Getter' function for member `m_category`

- void `category` (`category_t` `c`)

'Setter' function for member `m_category` by value Also keeps the `m_name_category` member in synchrony.

- const std::string & `category_string` () const

'Getter' function for member `m_category`

- void `category` (const std::string &`cs`)

'Setter' function for member `m_category` by name Also keeps the `m_name_category` member in synchrony, but looks up the name, rather than using the name parameter, to avoid storing abbreviations.

- const std::string & `timestamp_string` () const

'Getter' function for member `m_name_timestamp`

- `midipulse` `timestamp` () const

'Getter' function for member `event::get_timestamp()` Implemented to allow a uniform naming convention that is not slavish to the get/set crowd [this ain't Java or, chuckle, C#].

- void `timestamp` (`midipulse` `ts`)

'Setter' function for member `event::set_timestamp()` Implemented to allow a uniform naming convention that is not slavish to the get/set crowd [this ain't Java].

- void `timestamp` (const std::string &`ts_string`)

'Setter' function for member `event::set_timestamp()` [string version]

- std::string `time_as_pulses` ()

Converts the current time-stamp to a string representation in units of pulses.

- std::string `time_as_measures` ()

Converts the current time-stamp to a string representation in units of measures, beats, and divisions.

- `std::string time_as_minutes ()`
Converts the current time-stamp to a string representation in units of hours, minutes, seconds, and fraction.
- `void set_status_from_string (const std::string &ts, const std::string &s, const std::string &sd0, const std::string &sd1)`
Converts a string into an event status, along with timestamp and data bytes.
- `std::string format_timestamp ()`
Formats the current timestamp member as a string.
- `std::string stock_event_string ()`
Converts the event into a string describing the full event.
- `std::string ex_data_string () const`
Assuming the event is a Meta event or a SysEx, this function returns a short string representation of the event data, usable in the eventeditor class or elsewhere.
- `std::string status_string () const`
'Getter' function for member m_name_status
- `std::string meta_string () const`
'Getter' function for member m_name_meta
- `std::string seqspec_string () const`
'Getter' function for member m_name_seqspec
- `std::string channel_string () const`
'Getter' function for member m_name_channel
- `std::string data_string () const`
'Getter' function for member m_name_data

Static Public Member Functions

- `static std::string value_to_name (midibyte value, category_t cat)`
Provides a static lookup function that returns the name, if any, associated with a midibyte value.
- `static unsigned short name_to_value (const std::string &name, category_t cat)`
Provides a static lookup function that returns the value, if any, associated with a name string.
- `static unsigned short meta_event_length (midibyte value)`
Provides a static lookup function that takes a meta-event number and returns the expected length of the data for that event.

Static Public Attributes

- `static const name_value_t sm_category_names []`
An array of event categories and their names.
- `static const name_value_t sm_channel_event_names []`
An array of MIDI channel events and their names.
- `static const name_value_t sm_system_event_names []`
An array of MIDI system events and their names.
- `static const name_value_t sm_meta_event_names []`
An array of Meta events and their names.
- `static const name_value_t sm_prop_event_names []`
An array of Sequencer64-specific events and their names.
- `static const name_value_t *const sm_category_arrays []`
Provides for fast access (no ifs) to the correct name array for the given category.
- `static const meta_length_t sm_meta_lengths []`
Provides a list of meta-event numbers and their expected lengths (if any).

Private Member Functions

- [editable_event](#) ()
- void [analyze](#) ()
Analyzes an editable-event to make all the settings it needs.

Private Attributes

- const [editable_events](#) & [m_parent](#)
Provides a reference to the container that holds this event.
- [category_t](#) [m_category](#)
Indicates the overall category of this event, which will be `category_channel_message`, `category_system_message`, `category_meta_event`, and `category_prop_event`.
- std::string [m_name_category](#)
Holds the name of the event category for this event.
- [timestamp_format_t](#) [m_format_timestamp](#)
Indicates the format to display the time-stamp.
- std::string [m_name_timestamp](#)
Holds the string version of the MIDI pulses time-stamp.
- std::string [m_name_status](#)
Holds the name of the status value for this event.
- std::string [m_name_meta](#)
Holds the name of the meta message, if applicable.
- std::string [m_name_seqspect](#)
If we eventually implement the editing of the Seq24/Sequencer64 "proprietary" meta sequencer-specific events, the name of the SeqSpec will be stored here.
- std::string [m_name_channel](#)
Holds the channel description, if applicable.
- std::string [m_name_data](#)
Holds the data description, if applicable.

10.7.1 Detailed Description

It makes the following members of an event modifiable using human-readable strings:

```
- m_timestamp
- m_status
- m_channel
- m_data[]
```

Eventually, it would be nice to be able to edit, or at least view, the SysEx events and the Meta events. Those two will require extensions to make events out of them (SysEx is partly supported).

To the concepts of event, the [editable_event](#) class adds a category field and strings to represent all of these members.

10.7.2 Member Enumeration Documentation

10.7.2.1 category_t

```
enum seq64::editable_event::category_t
```

These tags are accompanied by category names in `sm_category_names[]`. The enum values are cast to midibyte values for the purposes of using the lookup infrastructure.

Enumerator

category_name	Indicates that the lookup needs to be done on the category names, as listed in sm_category_names[].
category_channel_message	Indicates a channel event, with a value ranging from 0x80 through 0xEF. Some examples are note on/off, control change, and program change. Values are looked up in sm_channel_event_names[].
category_system_message	Indicates a system event, with a value ranging from 0xF0 through 0xFF. Some examples are SysEx start/end, song position, and stop/start/continue/reset. Values are looked up in sm_system_event_names[]. These values are "real" only in MIDI data coming in "over the wire". In MIDI files, they represent Meta events.
category_meta_event	Indicates a meta event, and there is a second value that is used to look up the name of the meta event, in sm_meta_event_names[]. Meta messages are message that are stored in a MIDI file. Although they start with 0xFF, they are not to be confused with the 0xFF message that can be sent "over the wire", which denotes a Reset event.
category_prop_event	Indicates a "proprietary", Sequencer64 event. Indicates to look up the name of the event in sm_prop_event_names[]. Not sure if these kinds of events will be stored separately.

10.7.2.2 timestamp_format_t

```
enum seq64::editable_event::timestamp_format_t
```

Three are supported. All editable events will share the same timestamp format, but it seems good to make this a event class member, rather than something imposed from an outside static value. We shall see.

Enumerator

timestamp_measures	This format displays the time in "measures:beats:divisions" format, where measures and beats start at 1. Thus, "1:1:0" is equivalent to 0 pulses or to "0:0:0.0" in normal time values.
timestamp_time	This format displays the time in "hh:mm:second.fraction" format. The value displayed should not depend upon the internal timing parameters of the event.
timestamp_pulses	This format specifies a bare pulse format for the timestamp – a long integer ranging from 0 on up. Obviously, this representation depends on the PPQN value for the sequence holding this event.

10.7.3 Constructor & Destructor Documentation

10.7.3.1 editable_event() [1/4]

```
seq64::editable_event::editable_event ( ) [private]
```

10.7.3.2 `editable_event()` [2/4]

```
seq64::editable_event::editable_event (
    const editable_events & parent )
```

The default constructor is hidden and unimplemented. We will get the default controller name from the controllers module. We should also be able to look up the selected buss's entries for a sequence, and load up the CC/name pairs on the fly.

Parameters

<i>parent</i>	Provides the overall editable-events object that manages the whole set of editable-event.
---------------	---

10.7.3.3 `editable_event()` [3/4]

```
seq64::editable_event::editable_event (
    const editable_events & parent,
    const event & ev )
```

This function basically adds all of the extra `editable_event` stuff to a standard event, so that the resulting `editable_event` is container-ready.

10.7.3.4 `editable_event()` [4/4]

```
seq64::editable_event::editable_event (
    const editable_event & rhs )
```

This function is currently geared only toward support of the SMF 0 channel-splitting feature. Many of the members are not set to useful values when the MIDI file is read, so we don't handle them for now.

Warning

This function does not yet copy the SysEx data. The inclusion of SysEx `editable_events` was not complete in Seq24, and it is still not complete in Sequencer64. Nor does it currently bother with the links.

Parameters

<i>rhs</i>	Provides the <code>editable_event</code> object to be copied.
------------	---

10.7.3.5 `~editable_event()`

```
virtual seq64::editable_event::~~editable_event ( ) [inline], [virtual]
```


10.7.4 Member Function Documentation

10.7.4.1 value_to_name()

```
std::string seq64::editable_event::value_to_name (
    midibyte value,
    editable_event::category_t cat ) [static]
```

Parameters

<i>value</i>	The MIDI byte value to look up.
<i>cat</i>	The category of the MIDI byte. Each category calls a different name array into play.

Returns

Returns the name associated with the value. If there is no such name, then an empty string is returned.

10.7.4.2 name_to_value()

```
unsigned short seq64::editable_event::name_to_value (
    const std::string & name,
    editable_event::category_t cat ) [static]
```

The `string_match()` function, which can match abbreviations, case-insensitively, is used to make the string comparisons.

Parameters

<i>name</i>	The string value to look up.
<i>cat</i>	The category of the MIDI byte. Each category calls a different name array into play.

Returns

Returns the value associated with the name. If there is no such value, then `SEQ64_END_OF_MIDIBYTE_TABLE` is returned.

10.7.4.3 meta_event_length()

```
unsigned short seq64::editable_event::meta_event_length (
    midibyte value ) [static]
```

Parameters

<i>value</i>	The MIDI byte value to look up.
--------------	---------------------------------

Returns

Returns the length associated with the meta event. If the expected length is actually 0, or is variable, then 0 is returned.

10.7.4.4 operator=()

```

editable_event & seq64::editable_event::operator= (
    const editable_event & rhs )

```

10.7.4.5 parent()

```

const editable_events& seq64::editable_event::parent ( ) const [inline]

```

10.7.4.6 category() [1/3]

```

category_t seq64::editable_event::category ( ) const [inline]

```

10.7.4.7 category() [2/3]

```

void seq64::editable_event::category (
    category_t c )

```

Note that a bad value is translated to the enum value `category_name`.

Parameters

<i>c</i>	Provides the category value to set.
----------	-------------------------------------

10.7.4.8 category_string()

```

const std::string& seq64::editable_event::category_string ( ) const [inline]

```

10.7.4.9 category() [3/3]

```
void seq64::editable_event::category (
    const std::string & name )
```

Note that a bad value is translated to the value of category_name.

Parameters

<i>name</i>	Provides the category name for the category value to set.
-------------	---

10.7.4.10 timestamp_string()

```
const std::string& seq64::editable_event::timestamp_string ( ) const [inline]
```

10.7.4.11 timestamp() [1/3]

```
midipulse seq64::editable_event::timestamp ( ) const [inline]
```

10.7.4.12 timestamp() [2/3]

```
void seq64::editable_event::timestamp (
    midipulse ts )
```

Plus, we also have to set the string version at the same time.

The format of the string representation is of the format selected by the m_format_timestamp member and is set by the [format_timestamp\(\)](#) function.

Parameters

<i>ts</i>	Provides the timestamp in units of MIDI pulses.
-----------	---

10.7.4.13 timestamp() [3/3]

```
void seq64::editable_event::timestamp (
    const std::string & ts_string )
```

The format of the string representation is of the format selected by the `m_format_timestamp` member and is set by the `format_timestamp()` function.

Parameters

<i>ts_string</i>	Provides the timestamp in units of MIDI pulses.
------------------	---

10.7.4.14 `time_as_pulses()`

```
std::string seq64::editable_event::time_as_pulses ( ) [inline]
```

10.7.4.15 `time_as_measures()`

```
std::string seq64::editable_event::time_as_measures ( )
```

Cannot be inlined because of a circular dependency between the `editable_event` and `editable_events` classes.

10.7.4.16 `time_as_minutes()`

```
std::string seq64::editable_event::time_as_minutes ( )
```

Cannot be inlined because of a circular dependency between the `editable_event` and `editable_events` classes.

10.7.4.17 `set_status_from_string()`

```
void seq64::editable_event::set_status_from_string (
    const std::string & ts,
    const std::string & s,
    const std::string & sd0,
    const std::string & sd1 )
```

Currently, this function handles only the following two messages:

- `category_channel_message`
- `category_system_message`

After all of the numbering member items have been set, they are converted and assigned to the string versions via a call to the `analyze()` function.

Parameters

<i>ts</i>	Provides the time-stamp string of the event.
<i>s</i>	Provides the name of the event, such as "Program Change".
<i>sd0</i>	Provides the string defining the first data byte of the event. For Meta events, this might have multiple values, though we support only Set Tempo and Time Signature at present. Generated by Doxygen
<i>sd1</i>	Provides the string defining the second data byte of the event, if applicable to the event. Some meta event may provide multiple values in this string.

10.7.4.18 format_timestamp()

```
std::string seq64::editable_event::format_timestamp ( )
```

The format of the string representation is of the format selected by the `m_format_timestamp` member.

10.7.4.19 stock_event_string()

```
std::string seq64::editable_event::stock_event_string ( )
```

We get the time-stamp as a string, make sure the event is fully analyzed so that all items and strings are set correctly.

Returns

Returns a human-readable string describing this event. This string is displayed in an event list, such as in the `eventedit` module.

10.7.4.20 ex_data_string()

```
std::string seq64::editable_event::ex_data_string ( ) const
```

Most SysEx events will only show the first few bytes; we could make a SysEx viewer/editor for handling long events.

Returns

Returns the data string. If empty, the data is bad in some way, or the event is not a Meta event.

10.7.4.21 status_string()

```
std::string seq64::editable_event::status_string ( ) const [inline]
```

10.7.4.22 meta_string()

```
std::string seq64::editable_event::meta_string ( ) const [inline]
```

10.7.4.23 seqspec_string()

```
std::string seq64::editable_event::seqspec_string ( ) const [inline]
```

10.7.4.24 channel_string()

```
std::string seq64::editable_event::channel_string ( ) const [inline]
```

10.7.4.25 data_string()

```
std::string seq64::editable_event::data_string ( ) const [inline]
```

10.7.4.26 analyze()

```
void seq64::editable_event::analyze ( ) [private]
```

Used in the constructors. Some of the setters indirectly set the appropriate string representation, as well.

Category:

This function can figure out if the status byte implies a channel message or a system message, and set the category string as well. However, at this time, detection of Meta events (0xFF) or Proprietary/SeqSpec events (0xFF with 0x2424) doesn't work due to lack of context here (and due to the fact that currently such events are not yet stored in a Sequencer64 sequence/track, and the least-significant-byte gets masked off anyway.)

Status:

We distinguish between channel and system messages, and then one- and two-byte messages, but don't yet distinguish the data values fully.

Sysex and Meta events:

We are starting to support events with statuses ranging from 0xF0 to 0xFF, with a concentration on Set Tempo and Time Signature events. We want them to be full-fledged Sequencer64 events.

The 0xFF byte represents a Meta event, not a Reset event, when we're dealing with data from a MIDI file, as we are here. And we need to get the next byte after the status byte.

We want Set Tempo events to appear as "Tempo 120.0" and Time Signature events to appear as "Time Sig 4/4".

10.7.5 Field Documentation

10.7.5.1 sm_category_names

```
const editable_event::name_value_t seq64::editable_event::sm_category_names [static]
```

Initializes the array of event/name pairs for the MIDI events categories.

Terminated by an empty string, the latter being the preferred test, for consistency with the other arrays and because 0 is often a legitimate code value.

10.7.5.2 sm_channel_event_names

```
const editable_event::name_value_t seq64::editable_event::sm_channel_event_names [static]
```

Initializes the array of event/name pairs for the channel MIDI events.

We split channel and system messages into two arrays, for semantic reasons and for faster linear lookups.

Terminated by an empty string.

10.7.5.3 sm_system_event_names

```
const editable_event::name_value_t seq64::editable_event::sm_system_event_names [static]
```

Initializes the array of event/name pairs for the system MIDI events.

We split channel and system messages into two arrays, for semantic reasons and for faster linear lookups.

Terminated by an empty string.

10.7.5.4 sm_meta_event_names

```
const editable_event::name_value_t seq64::editable_event::sm_meta_event_names [static]
```

Initializes the array of event/name pairs for all of the Meta events.

Terminated only by the empty string.

10.7.5.5 sm_prop_event_names

```
const editable_event::name_value_t seq64::editable_event::sm_prop_event_names [static]
```

Initializes the array of event/name pairs for all of the seq24/sequencer64-specific events.

Terminated only by the empty string. Note that the numbers reflect the masking off of the high-order bits by 0x242400FF.

10.7.5.6 sm_category_arrays

```
const editable_event::name_value_t *const seq64::editable_event::sm_category_arrays [static]
```

Contains pointers (references cannot be stored in an array) to the desired array for a given category.

Too bad that an array of references is not possible.

This code could be considered a bit rococo.

10.7.5.7 sm_meta_lengths

```
const editable_event::meta_length_t seq64::editable_event::sm_meta_lengths [static]
```

Initializes the array of event/length pairs for all of the Meta events.

Terminated only by the empty string.

10.7.5.8 m_parent

```
const editable_events& seq64::editable_event::m_parent [private]
```

The container's "children" need to go to their "parent" to get certain items of information.

10.7.5.9 m_category

```
category_t seq64::editable_event::m_category [private]
```

The category_name value is not set here, since that category is used only for looking up the human-readable form of the category.

10.7.5.10 m_name_category

```
std::string seq64::editable_event::m_name_category [private]
```

10.7.5.11 m_format_timestamp

```
timestamp_format_t seq64::editable_event::m_format_timestamp [private]
```

The default is to display in timestamp_measures format.

10.7.5.12 m_name_timestamp

```
std::string seq64::editable_event::m_name_timestamp [private]
```


10.7.5.13 m_name_status

```
std::string seq64::editable_event::m_name_status [private]
```

It will include the names of the channel messages and the system messages. The latter includes SysEx and Meta messages.

10.7.5.14 m_name_meta

```
std::string seq64::editable_event::m_name_meta [private]
```

If not applicable, this name will be empty.

10.7.5.15 m_name_seqspect

```
std::string seq64::editable_event::m_name_seqspect [private]
```

10.7.5.16 m_name_channel

```
std::string seq64::editable_event::m_name_channel [private]
```

10.7.5.17 m_name_data

```
std::string seq64::editable_event::m_name_data [private]
```

10.8 seq64::editable_events Class Reference

Provides for the management of an ordered collection MIDI editable events.

Public Member Functions

- `editable_events` (`sequence` &seq, `midibpm` bpm)

This constructor hooks into the sequence object.
- `editable_events` (const `editable_events` &rhs)

This copy constructor initializes most of the class members.
- `editable_events` & `operator=` (const `editable_events` &rhs)

This principal assignment operator sets most of the class members.
- virtual `~editable_events` ()

This destructor current is a rote virtual function override.
- const `midi_timing` & `timing` () const

'Getter' function for member m_midi_parameters
- `midipulse string_to_pulses` (const std::string &ts_string) const

Calculates the MIDI pulses (divisions) from a string using one of the free functions of the calculations module.
- bool `load_events` ()

Accesses the sequence's event-list, iterating through it from beginning to end, wrapping each event in the list in an editable event and inserting it into the editable-event container.
- bool `save_events` ()

Erases the sequence's event container and recreates it using the edited container of editable events.
- `Events` & `events` ()

'Getter' function for member m_events
- `iterator begin` ()

'Getter' function for member m_events.begin(), non-constant version.
- `const_iterator begin` () const

'Getter' function for member m_events.begin(), constant version.
- `iterator end` ()

'Getter' function for member m_events.end(), non-constant version.
- `const_iterator end` () const

'Getter' function for member m_events.end(), constant version.
- int `count` () const

Returns the number of events stored in m_events.
- `midipulse get_length` () const

Provides the length of the events in MIDI pulses.
- bool `add` (const `event` &e)

Adds an event, converted to an `editable_event`, to the internal event list.
- bool `add` (const `editable_event` &e)

Adds an editable event to the internal event list.
- bool `replace` (`iterator` ie, const `editable_event` &e)

Provides a wrapper for the iterator form of erase(), which is the only one that the `editable_events` container uses.
- void `remove` (`iterator` ie)

Provides a wrapper for the iterator form of erase(), which is the only one that sequence uses.
- void `clear` ()

Provides a wrapper for `clear()`.
- void `sort` ()

Sorts the event list; active only for the std::list implementation.
- `iterator current_event` () const

'Getter' function for member m_current_event The caller must make sure the iterator is not Events::end().
- bool `is_valid_iterator` (`iterator` &cit) const

Validates the given iterator.
- void `print` () const

Prints a list of the currently-held events.

Static Public Member Functions

- static [editable_event](#) & [dref](#) ([iterator](#) ie)
Dereference access for list or map.
- static const [editable_event](#) & [dref](#) ([const_iterator](#) ie)
Dereference const access for list or map.

Private Types

- typedef [event_list::event_key](#) [Key](#)
Types to use to with the multimap implementation.
- typedef std::pair< [Key](#), [editable_event](#) > [EventsPair](#)
- typedef std::multimap< [Key](#), [editable_event](#) > [Events](#)
- typedef [Events::iterator](#) [iterator](#)
- typedef [Events::const_iterator](#) [const_iterator](#)
- typedef [Events::reverse_iterator](#) [reverse_iterator](#)
- typedef [Events::const_reverse_iterator](#) [const_reverse_iterator](#)

Private Member Functions

- [editable_events](#) ()
- void [current_event](#) ([iterator](#) cei)
'Setter' function for member m_current_event

Private Attributes

- [Events](#) [m_events](#)
Holds the [editable_events](#).
- [iterator](#) [m_current_event](#)
Points to the current event, which is the event that has just been inserted.
- [sequence](#) & [m_sequence](#)
Provides a reference to the sequence containing the events to be edited.
- [midi_timing](#) [m_midi_parameters](#)
Holds the current settings for the sequence (and usually for the whole MIDI tune as well).

Friends

- class [eventslots](#)

10.8.1 Member Typedef Documentation

10.8.1.1 Key

```
typedef event_list::event_key seq64::editable_events::Key [private]
```

These typenames are identical to those used in [event_list](#), but of course they are in the [editable_events](#) scope instead. See the [event_list](#) class; that class once again uses `std::list`, but still defines [event_list::event_key](#) for use here. We think the `std::list` implementation breaks editing, so we're going back to the `std::map` implementation for the event-editor.

10.8.1.2 EventsPair

```
typedef std::pair<Key, editable_event> seq64::editable_events::EventsPair [private]
```

10.8.1.3 Events

```
typedef std::multimap<Key, editable_event> seq64::editable_events::Events [private]
```

10.8.1.4 iterator

```
typedef Events::iterator seq64::editable_events::iterator [private]
```

10.8.1.5 const_iterator

```
typedef Events::const_iterator seq64::editable_events::const_iterator [private]
```

10.8.1.6 reverse_iterator

```
typedef Events::reverse_iterator seq64::editable_events::reverse_iterator [private]
```

10.8.1.7 const_reverse_iterator

```
typedef Events::const_reverse_iterator seq64::editable_events::const_reverse_iterator [private]
```

10.8.2 Constructor & Destructor Documentation

10.8.2.1 editable_events() [1/3]

```
seq64::editable_events::editable_events ( ) [private]
```

10.8.2.2 editable_events() [2/3]

```
seq64::editable_events::editable_events (
    sequence & seq,
    midibpm bpm )
```

Parameters

<i>seq</i>	Provides a reference to the sequence object, which provides the events and some of the MIDI timing parameters.
<i>bpm</i>	Provides the beats/minute value, which the caller figures out how to get and provides in this parameter.

10.8.2.3 `editable_events()` [3/3]

```
seq64::editable_events::editable_events (
    const editable\_events & rhs )
```

Note that we need to reconstitute the event links here, as well.

Parameters

<i>rhs</i>	Provides the editable_events object to be copied.
------------	---

10.8.2.4 `~editable_events()`

```
virtual seq64::editable_events::~~editable_events ( ) [inline], [virtual]
```

10.8.3 Member Function Documentation

10.8.3.1 `operator=()`

```
editable\_events & seq64::editable_events::operator= (
    const editable\_events & rhs )
```

Note that we need to reconstitute the event links here, as well.

Parameters

<i>rhs</i>	Provides the editable_events object to be assigned.
------------	---

Returns

Returns a reference to "this" object, to support the serial assignment of `editable_eventss`.

10.8.3.2 timing()

```
const midi_timing& seq64::editable_events::timing ( ) const [inline]
```

10.8.3.3 string_to_pulses()

```
midipulse seq64::editable_events::string_to_pulses (
    const std::string & ts_string ) const [inline]
```

10.8.3.4 load_events()

```
bool seq64::editable_events::load_events ( )
```

Note that the new events will not have valid links (actually, no links). These links are used for associating Note Off events with their respective Note On events. To be consistent, we must take the time to reconstitute these links, using [event_list::verify_and_link\(\)](#).

Returns

Returns true if the size of the final [editable_event](#) container matches the size of the original events container.

10.8.3.5 save_events()

```
bool seq64::editable_events::save_events ( )
```

Note that the old events are replaced only if the container of editable events is not empty. There are safer ways for the user to erase all the events.

Todo Consider what to do about the `sequence::m_is_modified` flag.

Returns

Returns true if the size of the final event container matches the size of the original [editable_events](#) container.

10.8.3.6 events()

```
Events& seq64::editable_events::events ( ) [inline]
```

10.8.3.7 begin() [1/2]

```
iterator seq64::editable_events::begin ( ) [inline]
```

10.8.3.8 begin() [2/2]

```
const_iterator seq64::editable_events::begin ( ) const [inline]
```

10.8.3.9 end() [1/2]

```
iterator seq64::editable_events::end ( ) [inline]
```

10.8.3.10 end() [2/2]

```
const_iterator seq64::editable_events::end ( ) const [inline]
```

10.8.3.11 dref() [1/2]

```
static editable_event& seq64::editable_events::dref (
    iterator ie ) [inline], [static]
```

Parameters

<i>ie</i>	Provides the iterator to the event to which to get a reference.
-----------	---

10.8.3.12 dref() [2/2]

```
static const editable_event& seq64::editable_events::dref (
    const_iterator ie ) [inline], [static]
```

Parameters

<i>ie</i>	Provides the iterator to the event to which to get a reference.
-----------	---

10.8.3.13 count()

```
int seq64::editable_events::count ( ) const [inline]
```

We like returning an integer instead of `size_t`, and rename the function so nobody is fooled.

10.8.3.14 get_length()

```
midipulse seq64::editable_events::get_length ( ) const
```

This function gets the iterator for the last element and returns its length value.

Returns

Returns the timestamp of the latest event in the container.

10.8.3.15 add() [1/2]

```
bool seq64::editable_events::add (
    const event & e )
```

Parameters

<i>e</i>	Provides the regular event to be added to the list of editable events.
----------	--

Returns

Returns true if the insertion succeeded, as evidenced by an increment in container size.

10.8.3.16 add() [2/2]

```
bool seq64::editable_events::add (
    const editable_event & e )
```

For the `std::multimap` implementation, this is an option if we want to make sure the insertion succeed:

```
*      std::pair<Events::iterator, bool> result = m_events.insert(p);
*      return result.second;
```

Parameters

<i>e</i>	Provides the regular event to be added to the list of editable events.
----------	--

Returns

Returns true if the insertion succeeded, as evidenced by an increment in container size.

Side-effect(s) Sets `m_current_event`, which can be used right-away in a single-threaded context to get an iterator to the event via the `current_event()` accessor.

10.8.3.17 replace()

```
bool seq64::editable_events::replace (
    iterator ie,
    const editable_event & e ) [inline]
```

10.8.3.18 remove()

```
void seq64::editable_events::remove (
    iterator ie ) [inline]
```

10.8.3.19 clear()

```
void seq64::editable_events::clear ( ) [inline]
```

10.8.3.20 sort()

```
void seq64::editable_events::sort ( ) [inline]
```

10.8.3.21 current_event() [1/2]

```
iterator seq64::editable_events::current_event ( ) const [inline]
```

10.8.3.22 is_valid_iterator()

```
bool seq64::editable_events::is_valid_iterator (
    iterator & cit ) const [inline]
```

10.8.3.23 print()

```
void seq64::editable_events::print ( ) const
```

Useful for debugging.

10.8.3.24 current_event() [2/2]

```
void seq64::editable_events::current_event (
    iterator cei ) [inline], [private]
```

Parameters

<code>cei</code>	Provide an iterator to the event to set as the current event.
------------------	---

10.8.4 Friends And Related Function Documentation

10.8.4.1 eventslots

```
friend class eventslots [friend]
```

10.8.5 Field Documentation

10.8.5.1 m_events

```
Events seq64::editable_events::m_events [private]
```

Just to be clear, since we currently do not define SEQ64_USE_EVENT_MAP, this is an std::list container, not a multimap. BELAY THAT! The multimap works better here.

10.8.5.2 m_current_event

```
iterator seq64::editable_events::m_current_event [private]
```

(From this event we can get the current time and other parameters.) If the container were a plain map, we could instead use a key to access it. But we can at least use an iterator, rather than a bare pointer.

10.8.5.3 m_sequence

```
sequence& seq64::editable_events::m_sequence [private]
```

Besides the events, this object also holds the beats/measure, beat-width, and the PPQN value. The beats/minute have to be obtained from the application's perform object, and passed to the [editable_events](#) constructor by the caller.

10.8.5.4 m_midi_parameters

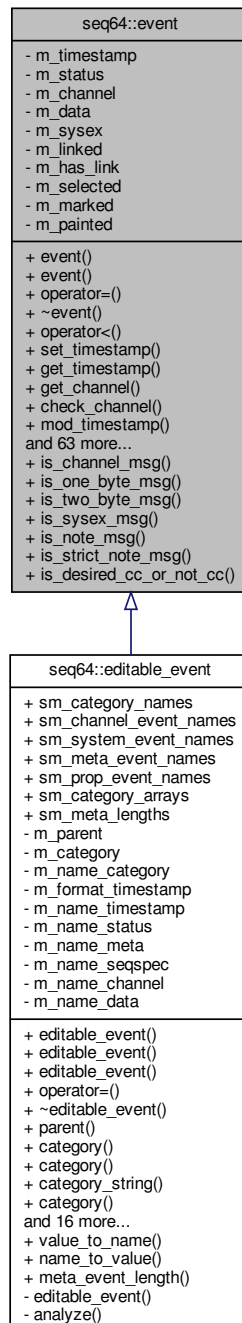
```
midi_timing seq64::editable_events::m_midi_parameters [private]
```

It holds the beats/minute, beats/measure, beat-width, and PPQN values needed to properly convert MIDI pulse timestamps to time and measure values.

10.9 seq64::event Class Reference

Provides events for management of MIDI events.

Inheritance diagram for seq64::event:



Public Types

- `typedef std::vector< midibyte > SysexContainer`

Provides a type definition for a vector of midibytes.

Public Member Functions

- [event](#) ()
This constructor simply initializes all of the class members.
- [event](#) (const [event](#) &rhs)
This copy constructor initializes most of the class members.
- [event](#) & [operator=](#) (const [event](#) &rhs)
This principal assignment operator sets most of the class members.
- virtual [~event](#) ()
This destructor explicitly deletes `m_sysex` and sets it to null.
- bool [operator<](#) (const [event](#) &rhsevent) const
If the current timestamp equal the event's timestamp, then this function returns true if the current rank is less than the event's rank.
- void [set_timestamp](#) ([midipulse](#) time)
'Setter' function for member `m_timestamp`
- [midipulse](#) [get_timestamp](#) () const
'Getter' function for member `m_timestamp`
- [midibyte](#) [get_channel](#) () const
'Getter' function for member `m_channel`
- bool [check_channel](#) (int channel) const
Checks the channel number to see if the event's channel matches it, or if the event has no channel.
- void [mod_timestamp](#) ([midipulse](#) modtick)
Calculates the value of the current timestamp modulo the given parameter.
- void [set_status](#) ([midibyte](#) status)
Sets the `m_status` member to the value of status.
- void [set_status](#) ([midibyte](#) eventcode, [midibyte](#) channel)
This overload is useful when synthesizing events, such as converting a Note On event with a velocity of zero to a Note Off event.
- void [set_meta_status](#) ([midibyte](#) metatype)
Sets a Meta event.
- void [set_status_keep_channel](#) ([midibyte](#) eventcode)
This function is used in recording to preserve the input channel information for deciding what to do with an incoming MIDI event.
- void [set_channel](#) ([midibyte](#) channel)
Sets the channel "nybble", without modifying the status "nybble".
- [midibyte](#) [get_status](#) () const
'Getter' function for member `m_status` Note that we have ensured that status ranges from 0x80 to 0xFF.
- bool [non_cc_match](#) ([midibyte](#) status)
Returns true if the event's status is not a control-change, but does match the given status.
- bool [cc_match](#) ([midibyte](#) st, [midibyte](#) cc)
Returns true if the event's status is a control-change that matches the given status, and has a control value matching the given control-change value.
- void [set_data](#) ([midibyte](#) d1)
Clears the most-significant-bit of the `d1` parameter, and sets it into the first byte of `m_data`.
- void [set_data](#) ([midibyte](#) d1, [midibyte](#) d2)
Clears the most-significant-bit of both parameters, and sets them into the first and second bytes of `m_data`.
- void [get_data](#) ([midibyte](#) &d0) const
Retrieves only the first data byte from `m_data[]` and copies it into the parameter.
- void [get_data](#) ([midibyte](#) &d0, [midibyte](#) &d1) const
Retrieves the two data bytes from `m_data[]` and copies each into its respective parameter.
- void [increment_data1](#) ()
Increments the first data byte (`m_data[0]`) and clears the most significant bit.

- void `decrement_data1` ()
Decrements the first data byte (`m_data[0]`) and clears the most significant bit.
- void `increment_data2` ()
Increments the second data byte (`m_data[1]`) and clears the most significant bit.
- void `decrement_data2` ()
Decrements the second data byte (`m_data[1]`) and clears the most significant bit.
- bool `append_sysex` (midibyte *data, int len)
Appends SYSEX data to a new buffer.
- bool `append_sysex` (midibyte data)
An overload for logging SYSEX data byte-by-byte.
- bool `append_meta_data` (midibyte metatype, midibyte *data, int len)
Appends Meta-event data to a new buffer.
- bool `append_meta_data` (midibyte metatype, const std::vector< midibyte > &data)
This overload appends Meta-event data from a vector to a new buffer.
- void `restart_sysex` ()
Deletes and clears out the SYSEX buffer.
- bool `set_sysex` (midibyte *data, int len)
Resets and adds ex data.
- `SysexContainer` & `get_sysex` ()
'Getter' function for member `m_sysex` from stazed, non-const version for use by midibus.
- const `SysexContainer` & `get_sysex` () const
'Getter' function for member `m_sysex` from stazed
- void `set_sysex_size` (int len)
'Setter' function for member `m_sysex` from stazed
- int `get_sysex_size` () const
'Getter' function for member `m_sysex.size()`
- void `link` (event *ev)
Sets `m_has_link` and sets `m_link` to the provided event pointer.
- event * `get_linked` () const
'Getter' function for member `m_linked`
- bool `is_linked` () const
'Getter' function for member `m_has_link`
- void `clear_link` ()
'Setter' function for member `m_has_link` and `m_linked`
- void `paint` ()
'Setter' function for member `m_painted`
- void `unpaint` ()
'Setter' function for member `m_painted`
- bool `is_painted` () const
'Getter' function for member `m_painted`
- void `mark` ()
'Setter' function for member `m_marked`
- void `unmark` ()
'Setter' function for member `m_marked`
- bool `is_marked` () const
'Getter' function for member `m_marked`
- void `select` ()
'Setter' function for member `m_selected`
- void `unselect` ()
'Setter' function for member `m_selected`
- bool `is_selected` () const

- *'Getter' function for member m_selected*
- void `make_clock ()`
Sets m_status to EVENT_MIDI_CLOCK;.
- `midibyte data (int index) const`
'Getter' function for member m_data[]
- `midibyte get_note () const`
Assuming m_data[] holds a note, get the note number, which is in the first data byte, m_data[0].
- void `set_note (midibyte note)`
Sets the note number, clearing off the most-significant-bit and assigning it to the first data byte, m_data[0].
- void `transpose_note (int tn)`
Transpose the note, if possible.
- `midibyte get_note_velocity () const`
'Getter' function for member m_data[1], the note velocity.
- void `set_note_velocity (int vel)`
Sets the note velocity, which is held in the second data byte, and clearing off the most-significant-bit, storing it in m_data[1].
- bool `is_note_on () const`
Check for the Note On value in m_status.
- bool `is_note_off () const`
Check for the Note Off value in m_status.
- bool `is_note () const`
Returns true if m_status is a Note On, Note Off, or Aftertouch message.
- bool `is_note_off_recorded () const`
Some keyboards send Note On with velocity 0 for Note Off, so we provide this function to test that during recording.
- void `adjust_note_off ()`
Some keyboards send Note On with velocity 0 for Note Off, so we take care of that situation here by creating a Note Off event, with the channel nybble preserved.
- bool `is_one_byte () const`
Indicates if the m_status value is a one-byte message (Program Change or Channel Pressure.
- bool `is_two_bytes () const`
Indicates if the m_status value is a two-byte message (everything except Program Change, Channel Pressure, and).
- bool `is_sysex () const`
Indicates if the event is a System Exclusive event or not.
- bool `is_meta () const`
Indicates if the event is a Meta event or not.
- bool `is_ex_data () const`
Indicates if we need to use extended data (SysEx or Meta).
- bool `is_tempo () const`
Indicates if the event is a tempo event.
- `midibpm tempo () const`
Calculates the tempo from the stored event bytes, if the event is a Tempo meta-event and has valid data.
- void `set_tempo (midibpm tempo)`
The inverse of tempo().
- bool `is_time_signature () const`
Indicates if the event is a Time Signature event.
- bool `is_key_signature () const`
Indicates if the event is a Key Signature event.
- void `print () const`
Prints out the timestamp, data size, the current status byte, channel (which is the type value for Meta events), any SysEx or Meta data if present, or the two data bytes for the status byte.
- int `get_rank () const`
This function is used in sorting MIDI status events (e.g.

Static Public Member Functions

- static bool `is_channel_msg` (midibyte m)
Static test for the channel message/status values: Note On, Note Off, Aftertouch, Control Change, Program Change, Channel Pressure, and Pitch Wheel.
- static bool `is_one_byte_msg` (midibyte m)
Static test for channel messages that have only one data byte: Program Change and Channel Pressure.
- static bool `is_two_byte_msg` (midibyte m)
Static test for channel messages that have two data bytes: Note On, Note Off, Control Change, Aftertouch, and Pitch Wheel.
- static bool `is_sysex_msg` (midibyte m)
Static test for a SysEx message.
- static bool `is_note_msg` (midibyte m)
Static test for messages that involve notes and velocity: Note On, Note Off, and Aftertouch.
- static bool `is_strict_note_msg` (midibyte m)
Static test for messages that involve notes only: Note On and Note Off.
- static bool `is_desired_cc_or_not_cc` (midibyte m, midibyte cc, midibyte datum)
Static test for channel messages that are either not control-change messages, or are and match the given controller value.

Private Attributes

- `midipulse m_timestamp`
Provides the MIDI timestamp in ticks, otherwise known as the "pulses" in "pulses per quarter note" (PPQN).
- `midibyte m_status`
This is the status byte without the channel.
- `midibyte m_channel`
In order to be able to handle MIDI channel-splitting of an SMF 0 file, we need to store the channel, even if we override it when playing the MIDI data.
- `midibyte m_data` [SEQ64_MIDI_DATA_BYTE_COUNT]
The two bytes of data for the MIDI event.
- `SysexContainer m_sysex`
The data buffer for SYSEX messages.
- `event * m_linked`
This event is used to link Note Ons and Offs together.
- `bool m_has_link`
Indicates that a link has been made.
- `bool m_selected`
Answers the question "is this event selected in editing.".
- `bool m_marked`
Answers the question "is this event marked in processing.".
- `bool m_painted`
Answers the question "is this event being painted.".

10.9.1 Detailed Description

A MIDI event consists of 3 bytes:

```
-# Status byte, 1sssnnn, where the sss bits specify the type of
   message, and the nnnn bits denote the channel number.
   The status byte always starts with 0.
-# The first data byte, 0xxxxxxx, where the data byte always
   start with 0, and the xxxxxxx values range from 0 to 127.
-# The second data byte, 0xxxxxxx.
```

This class may have too many member functions.

10.9.2 Member Typedef Documentation

10.9.2.1 SysexContainer

```
typedef std::vector<midibyte> seq64::event::SysexContainer
```

This type will also hold the generally small amounts of data needed for Meta events, but doesn't help us encapsulate derived values, such as tempo.

10.9.3 Constructor & Destructor Documentation

10.9.3.1 event() [1/2]

```
seq64::event::event ( )
```

10.9.3.2 event() [2/2]

```
seq64::event::event (
    const event & rhs )
```

This function is currently geared only toward support of the SMF 0 channel-splitting feature. Many of the members are not set to useful values when the MIDI file is read, so we don't handle them for now.

Note that now events are also copied when creating the [editable_events](#) container, so this function is even more important. The event links, for linking Note Off events to their respective Note On events, are dropped. Generally, they will need to be reconstituted by calling the [event_list::verify_and_link\(\)](#) function.

Warning

This function does not yet copy the SysEx data. The inclusion of SysEx events was not complete in Seq24, and it is still not complete in Sequencer64. Nor does it currently bother with the links, as noted above.

Parameters

<i>rhs</i>	Provides the event object to be copied.
------------	---

10.9.3.3 ~event()

```
seq64::event::~~event ( ) [virtual]
```

The `restart_sysex()` function does what we need. But now that `m_sysex` is a vector, no action is needed.

10.9.4 Member Function Documentation

10.9.4.1 `operator=()`

```
event & seq64::event::operator= (
    const event & rhs )
```

This function is currently geared only toward support of the SMF 0 channel-splitting feature. Many of the member are not set to useful value when the MIDI file is read, so we don't handle them for now.

Warning

This function now copies the SysEx data, but the inclusion of SysEx events was not complete in Seq24, and it is still not complete in Sequencer64. Nor does it currently bother with the link the event might have.

Parameters

<i>rhs</i>	Provides the event object to be assigned.
------------	---

Returns

Returns a reference to "this" object, to support the serial assignment of events.

10.9.4.2 `operator<()`

```
bool seq64::event::operator< (
    const event & rhs ) const
```

Otherwise, it returns true if the current timestamp is less than the event's timestamp.

Warning

The less-than operator is supposed to support a "strict weak ordering", and is supposed to leave equivalent values in the same order they were before the sort. However, every time we load and save our sample MIDI file, events get reversed. Here are program-changes that get reversed:

```
Save N:      0070: 6E 00 C4 48 00 C4 0C 00  C4 57 00 C4 19 00 C4 26
Save N+1:    0070: 6E 00 C4 26 00 C4 19 00  C4 57 00 C4 0C 00 C4 48
```

The 0070 is the offset within the versions of the b4uacuse-seq24.midi file.

Because of this mis-feature, and the very slow speed of loading a MIDI file when Sequencer64 is built for debugging, we are exploring using an `std::multimap` instead of an `std::list`. Search for occurrences of the `SEQ64_USE_EVENT_MAP` macro. (This actually works better than a list, for loading MIDI event, we have found, but may cause the upper limit of the number of playing sequences to drop a little, due to the overhead of incrementing multimap iterators versus list iterators).

Parameters

<i>rhs</i>	The object to be compared against.
------------	------------------------------------

Returns

Returns true if the time-stamp and "rank" are less than those of the comparison object.

10.9.4.3 set_timestamp()

```
void seq64::event::set_timestamp (
    midipulse time ) [inline]
```

Parameters

<i>time</i>	Provides the time value, in ticks, to set as the timestamp.
-------------	---

10.9.4.4 get_timestamp()

```
midipulse seq64::event::get_timestamp ( ) const [inline]
```

10.9.4.5 get_channel()

```
midibyte seq64::event::get_channel ( ) const [inline]
```

10.9.4.6 check_channel()

```
bool seq64::event::check_channel (
    int channel ) const [inline]
```

Used in the SMF 0 track-splitting code.

Parameters

<i>channel</i>	The channel to check.
----------------	-----------------------

Returns

Returns true if the given channel matches the event's channel.

10.9.4.7 is_channel_msg()

```
static bool seq64::event::is_channel_msg (
    midibyte m ) [inline], [static]
```

This function requires that the channel data have already been masked off.

Parameters

<i>m</i>	The channel status or message byte to be tested, with the channel bits masked off.
----------	--

We could add an optional boolean to cause the channel nybble to be explicitly cleared.

Returns

Returns true if the byte represents a MIDI channel message.

10.9.4.8 is_one_byte_msg()

```
static bool seq64::event::is_one_byte_msg (
    midibyte m ) [inline], [static]
```

The rest of the channel messages have two data bytes. This function requires that the channel data have already been masked off.

Parameters

<i>m</i>	The channel status or message byte to be tested, with the channel bits masked off.
----------	--

We could add an optional boolean to cause the channel nybble to be explicitly cleared.

Returns

Returns true if the byte represents a MIDI channel message that has only one data byte. However, if this function returns false, it might not be a channel message at all, so be careful.

10.9.4.9 is_two_byte_msg()

```
static bool seq64::event::is_two_byte_msg (
    midibyte m ) [inline], [static]
```

This function requires that the channel data have already been masked off.

Parameters

<i>m</i>	The channel status or message byte to be tested, with the channel bits masked off.
----------	--

We could add an optional boolean to cause the channel nybble to be explicitly cleared.

Returns

Returns true if the byte represents a MIDI channel message that has two data bytes. However, if this function returns false, it might not be a channel message at all, so be careful.

10.9.4.10 is_sysex_msg()

```
static bool seq64::event::is_sysex_msg (  
    midibyte m ) [inline], [static]
```

Parameters

<i>m</i>	The status/message byte to test, with the channel bits masked off.
----------	--

10.9.4.11 is_note_msg()

```
static bool seq64::event::is_note_msg (  
    midibyte m ) [inline], [static]
```

This function requires that the channel nybble has already been masked off.

Parameters

<i>m</i>	The channel status or message byte to be tested, with the channel bits masked off.
----------	--

We could add an optional boolean to cause the channel nybble to be explicitly cleared.

Returns

Returns true if the byte represents a MIDI note message.

10.9.4.12 is_strict_note_msg()

```
static bool seq64::event::is_strict_note_msg (  
    midibyte m ) [inline], [static]
```

Parameters

<i>m</i>	The channel status or message byte to be tested, with the channel bits masked off.
----------	--

Returns

Returns true if the byte represents a MIDI note on/off message.

10.9.4.13 is_desired_cc_or_not_cc()

```
static bool seq64::event::is_desired_cc_or_not_cc (
    midibyte m,
    midibyte cc,
    midibyte datum ) [inline], [static]
```

Note

The old logic was the first line, but can be simplified to the second line; the third line shows the abstract representation. Also made sure of this using a couple truth tables.

```
(m != EVENT_CONTROL_CHANGE) || (m == EVENT_CONTROL_CHANGE && d == cc)
(m != EVENT_CONTROL_CHANGE) || (d == cc)
a || (! a && b) => a || b
```

Parameters

<i>m</i>	The channel status or message byte to be tested, with the channel bits masked off.
<i>cc</i>	The desired cc value, which the datum must match, if the message is a control-change message.
<i>datum</i>	The current datum, to be compared to cc, if the message is a control-change message.

Returns

Returns true if the message is not a control-change, or if it is and the cc and datum parameters match.

10.9.4.14 mod_timestamp()

```
void seq64::event::mod_timestamp (
    midipulse modtick ) [inline]
```

Parameters

<i>modtick</i>	The tick value to mod the timestamp against.
----------------	--

Returns

Returns a value ranging from 0 to `_mod-1`.

10.9.4.15 `set_status()` [1/2]

```
void seq64::event::set_status (
    midibyte status )
```

If `a_status` is a channel event, then the channel portion of the status is cleared using a bitwise AND against `EVENT_CLEAR_CHAN_MASK`. This version is basically the Seq24 version with the additional setting of the Sequencer64-specific `m_channel` member.

Found in yet another fork of seq24: // ORL fait de la merde He also provided a very similar routine: `set_status_midibus()`.

Stazed:

The record parameter, if true, does not clear channel portion on record for channel specific recording. The channel portion is cleared in `sequence::stream_event()` by calling `set_status()` (`a_record = false`) after the matching channel is determined. Otherwise, we use a bitwise AND to clear the channel portion of the status. All events will be stored without the channel nybble. This is necessary since the channel is appended by `midibus::play()` based on the track.

Instead of adding a "record" parameter to `set_status()`, we provide a more specific function, `set_status_keep_channel()`, for use in the mastermidibus class. This usage also has the side-effect of allowing the usage of channel in the MIDI-control feature.

Parameters

<i>status</i>	The status byte, perhaps read from a MIDI file or edited in the sequencer's event editor. Sometime, this byte will have the channel nybble masked off. If that is the case, the eventcode/channel overload of this function is more appropriate.
---------------	--

10.9.4.16 `set_status()` [2/2]

```
void seq64::event::set_status (
    midibyte eventcode,
    midibyte channel )
```

See its usage around line 681 of `midifile.cpp`.

Parameters

<i>eventcode</i>	The status byte, perhaps read from a MIDI file. This byte is assumed to have already had its low nybble cleared by masking against <code>EVENT_CLEAR_CHAN_MASK</code> .
<i>channel</i>	The channel byte. Combined with the event-code, this makes a valid MIDI "status" byte. This byte is assumed to have already had its high nybble cleared by masking against <code>EVENT_GET_CHAN_MASK</code> .

10.9.4.17 set_meta_status()

```
void seq64::event::set_meta_status (
    midibyte metatype )
```

Meta events have a status byte of EVENT_MIDI_META == 0xff and a channel value that reflects the type of Meta event (e.g. 0x51 for a "Set Tempo" event).

Note that the data bytes (if any) for this event will still need to be added to the event via (for example) the [append_sysex\(\)](#) or [set_sysex\(\)](#) function.

Parameters

<i>metatype</i>	Indicates the type of meta event.
-----------------	-----------------------------------

10.9.4.18 set_status_keep_channel()

```
void seq64::event::set_status_keep_channel (
    midibyte eventcode )
```

It replaces stazed's [set_status\(\)](#) that had an optional "record" parameter. This call allows channel to be detected and used in MIDI control events. It "keeps" the channel in the status byte.

Todo THIS function WILL set a BOGUS CHANNEL on events >= SYSEX !!!!!

Parameters

<i>eventcode</i>	The status byte, generally read from the MIDI buss.
------------------	---

10.9.4.19 set_channel()

```
void seq64::event::set_channel (
    midibyte channel ) [inline]
```

It actually just sets the m_channel member. Note that the sequence channel generally overrides this value in the usage of the event.

Parameters

<i>channel</i>	The channel byte to be set. It is masked to ensure the value ranges from 0x0 to 0xF. This update should be safe, but we could allow EVENT_NULL_CHANNEL if issues are uncovered.
----------------	---

10.9.4.20 get_status()

```
midibyte seq64::event::get_status ( ) const [inline]
```

10.9.4.21 non_cc_match()

```
bool seq64::event::non_cc_match (
    midibyte status ) [inline]
```

Parameters

<i>status</i>	The status to be checked.
---------------	---------------------------

10.9.4.22 cc_match()

```
bool seq64::event::cc_match (
    midibyte st,
    midibyte cc ) [inline]
```

Parameters

<i>st</i>	The status to be checked.
<i>cc</i>	The control-change value to be checked against the events current "d0" value.

10.9.4.23 set_data() [1/2]

```
void seq64::event::set_data (
    midibyte d1 ) [inline]
```

The second byte of data is zeroed. The data bytes are in a two-byte array member, `m_data`. This setter is useful for Program Change and Channel Pressure events.

Parameters

<i>d1</i>	The byte value to set as the first data byte.
-----------	---

10.9.4.24 set_data() [2/2]

```
void seq64::event::set_data (
    midibyte d1,
    midibyte d2 ) [inline]
```

Parameters

<i>d1</i>	The first byte value to set.
<i>d2</i>	The second byte value to set.

10.9.4.25 get_data() [1/2]

```
void seq64::event::get_data (
    midibyte & d0 ) const [inline]
```

Parameters

<i>d0</i>	[out] The return reference for the first byte.
-----------	--

10.9.4.26 get_data() [2/2]

```
void seq64::event::get_data (
    midibyte & d0,
    midibyte & d1 ) const [inline]
```

Parameters

<i>d0</i>	[out] The return reference for the first byte.
<i>d1</i>	[out] The return reference for the first byte.

10.9.4.27 increment_data1()

```
void seq64::event::increment_data1 ( ) [inline]
```

10.9.4.28 decrement_data1()

```
void seq64::event::decrement_data1 ( ) [inline]
```

10.9.4.29 increment_data2()

```
void seq64::event::increment_data2 ( ) [inline]
```

10.9.4.30 decrement_data2()

```
void seq64::event::decrement_data2 ( ) [inline]
```

10.9.4.31 append_sysex() [1/2]

```
bool seq64::event::append_sysex (
    midibyte * data,
    int dsize )
```

We now use a vector instead of an array, so there is no need for reallocation and copying of the current SYSEX data. The data represented by data and dsize is appended to that data buffer.

Parameters

<i>data</i>	Provides the additional SysEx/Meta data. If not provided, nothing is done, and false is returned.
<i>dsize</i>	Provides the size of the additional SYSEX data. If not provided, nothing is done.

Returns

Returns false if there was an EVENT_MIDI_SYSEX_END byte in the appended data, or if an error occurred, and the caller needs to stop trying to process the data. We're not quite sure what to do with any extra data remains.

10.9.4.32 append_sysex() [2/2]

```
bool seq64::event::append_sysex (
    midibyte data )
```

Parameters

<i>data</i>	A single MIDI byte of data, assumed to be part of a SYSEX message event.
-------------	--

10.9.4.33 append_meta_data() [1/2]

```
bool seq64::event::append_meta_data (
```

```

    midibyte metatype,
    midibyte * data,
    int dsize )

```

Similar to [append_sysex\(\)](#), but useful for holding the data for a Meta event. Please note that Meta events and SysEx events shared the same "extended" data buffer that originated to support SysEx.

Also see [set_meta_status\(\)](#), which, like this function, sets the event::m_channel_member to indicate the type of Meta event, but, unlike this function, leaves the data alone. Also note that the [set_status\(\)](#) call in midifile flags the event as a Meta event. The handling of Meta events is not yet uniform between all the modules.

Warning

Currently does not clear the "sysex" buffer first.

Parameters

<i>metatype</i>	Provides the type of the Meta event, which is stored in the m_channel member.
<i>data</i>	Provides the Meta event's data. If not provided, nothing is done, and false is returned.
<i>dsize</i>	Provides the size of the data. If not provided, nothing is done.

Returns

Returns false if an error occurred, and the caller needs to stop trying to process the data.

10.9.4.34 [append_meta_data\(\)](#) [2/2]

```

bool seq64::event::append_meta_data (
    midibyte metatype,
    const std::vector< midibyte > & data )

```

Parameters

<i>metatype</i>	Provides the type of the Meta event, which is stored in the m_channel member.
<i>data</i>	Provides the Meta event's data as a vector.

Returns

Returns false if an error occurred, and the caller needs to stop trying to process the data.

10.9.4.35 [restart_sysex\(\)](#)

```

void seq64::event::restart_sysex ( )

```

(The m_sysex member used to be a pointer.)

10.9.4.36 set_sysex()

```
bool seq64::event::set_sysex (
    midibyte * data,
    int len ) [inline]
```

Parameters

<i>data</i>	Provides the SysEx/Meta data. If not provided, nothing is done, and false is returned.
<i>len</i>	The number of bytes to set.

Returns

Returns true if the function succeeded.

10.9.4.37 get_sysex() [1/2]

```
SysexContainer& seq64::event::get_sysex ( ) [inline]
```

10.9.4.38 get_sysex() [2/2]

```
const SysexContainer& seq64::event::get_sysex ( ) const [inline]
```

10.9.4.39 set_sysex_size()

```
void seq64::event::set_sysex_size (
    int len ) [inline]
```

10.9.4.40 get_sysex_size()

```
int seq64::event::get_sysex_size ( ) const [inline]
```

10.9.4.41 link()

```
void seq64::event::link (
    event * ev ) [inline]
```

Parameters

ev	Provides a pointer to the event value to set. If null, then m_has_link is set to false, to guarantee that is_linked() is correct.
----	---

10.9.4.42 get_linked()

```
event* seq64::event::get_linked ( ) const [inline]
```

10.9.4.43 is_linked()

```
bool seq64::event::is_linked ( ) const [inline]
```

10.9.4.44 clear_link()

```
void seq64::event::clear_link ( ) [inline]
```

10.9.4.45 paint()

```
void seq64::event::paint ( ) [inline]
```

10.9.4.46 unpaint()

```
void seq64::event::unpaint ( ) [inline]
```

10.9.4.47 is_painted()

```
bool seq64::event::is_painted ( ) const [inline]
```

10.9.4.48 mark()

```
void seq64::event::mark ( ) [inline]
```

10.9.4.49 unmark()

```
void seq64::event::unmark ( ) [inline]
```

10.9.4.50 is_marked()

```
bool seq64::event::is_marked ( ) const [inline]
```

10.9.4.51 select()

```
void seq64::event::select ( ) [inline]
```

10.9.4.52 unselect()

```
void seq64::event::unselect ( ) [inline]
```

10.9.4.53 is_selected()

```
bool seq64::event::is_selected ( ) const [inline]
```

10.9.4.54 make_clock()

```
void seq64::event::make_clock ( ) [inline]
```

10.9.4.55 data()

```
midibyte seq64::event::data (
    int index ) const [inline]
```

10.9.4.56 get_note()

```
midibyte seq64::event::get_note ( ) const [inline]
```

10.9.4.57 set_note()

```
void seq64::event::set_note (
    midibyte note ) [inline]
```

Parameters

<i>note</i>	Provides the note value to set.
-------------	---------------------------------

10.9.4.58 transpose_note()

```
void seq64::event::transpose_note (
    int tn )
```

Parameters

<i>tn</i>	The amount (positive or negative) to transpose a note. If the result is out of range, the transposition is not performed.
-----------	---

10.9.4.59 get_note_velocity()

```
midibyte seq64::event::get_note_velocity ( ) const [inline]
```

10.9.4.60 set_note_velocity()

```
void seq64::event::set_note_velocity (
    int vel ) [inline]
```

Parameters

<i>vel</i>	Provides the velocity value to set.
------------	-------------------------------------

10.9.4.61 is_note_on()

```
bool seq64::event::is_note_on ( ) const [inline]
```

Currently assumes that the channel nybble has already been stripped.

Returns

Returns true if m_status is EVENT_NOTE_ON.

10.9.4.62 is_note_off()

```
bool seq64::event::is_note_off ( ) const [inline]
```

Currently assumes that the channel nybble has already been stripped.

Returns

Returns true if m_status is EVENT_NOTE_OFF.

10.9.4.63 is_note()

```
bool seq64::event::is_note ( ) const [inline]
```

All of these are notes, associated with a MIDI key value. Uses the static function [is_note_msg\(\)](#).

Returns

The return value of [is_note_msg\(\)](#) is returned.

10.9.4.64 is_note_off_recorded()

```
bool seq64::event::is_note_off_recorded ( ) const [inline]
```

The channel nybble is masked off before the test, but this is unnecessary since the velocity byte doesn't contain the channel! And it is a nasty bug!

"(m_data[1] & EVENT_CLEAR_CHAN_MASK) == 0" is wrong!

"m_status == EVENT_NOTE_ON && m_data[1] == 0" replaced by an inline function call for robustness.

Returns

Returns true if the event is a Note On event with velocity of 0.

10.9.4.65 adjust_note_off()

```
void seq64::event::adjust_note_off ( )
```

Note that we call [event::set_status_keep_channel\(\)](#) instead of using stazed's set_status function with the "record" parameter. Also, we have to mask in the actual channel number.

Encapsulates some common code. This function assumes we have already set the status and data bytes.

10.9.4.66 is_one_byte()

```
bool seq64::event::is_one_byte ( ) const [inline]
```

Channel is stripped, because sometimes we keep the channel.

10.9.4.67 is_two_bytes()

```
bool seq64::event::is_two_bytes ( ) const [inline]
```

Channel is stripped, because sometimes we keep the channel.

10.9.4.68 is_sysex()

```
bool seq64::event::is_sysex ( ) const [inline]
```

We're overloading the SysEx support to handle Meta events as well. Perhaps we need to split this support out at some point.

10.9.4.69 is_meta()

```
bool seq64::event::is_meta ( ) const [inline]
```

We're overloading the SysEx support to handle Meta events as well.

10.9.4.70 is_ex_data()

```
bool seq64::event::is_ex_data ( ) const [inline]
```

If true, then the m_channel byte is used to encode the type of meta event.

10.9.4.71 is_tempo()

```
bool seq64::event::is_tempo ( ) const [inline]
```

See sm_meta_event_names[].

10.9.4.72 tempo()

```
midibpm seq64::event::tempo ( ) const
```

Remember that we are overloading the SysEx support to hold Meta-event data. Also note that we're treating the vector like an array, which is supposed to work.

Returns

Returns the result of calculating the tempo from the three data bytes. If an error occurs, 0.0 is returned.

10.9.4.73 set_tempo()

```
void seq64::event::set_tempo (
    midibpm tempo )
```

First, we convert beats/minute to a tempo microseconds value. Then we convert the microseconds to three tempo bytes.

10.9.4.74 is_time_signature()

```
bool seq64::event::is_time_signature ( ) const [inline]
```

See `sm_meta_event_names[]`.

10.9.4.75 is_key_signature()

```
bool seq64::event::is_key_signature ( ) const [inline]
```

See `sm_meta_event_names[]`.

10.9.4.76 print()

```
void seq64::event::print ( ) const
```

10.9.4.77 get_rank()

```
int seq64::event::get_rank ( ) const
```

The ranking, from high to low, is note off, note on, aftertouch, channel pressure, and pitch wheel, control change, and program changes.

note on/off, aftertouch, control change, etc.) The sort order is not determined by the actual status values.

The lower the ranking the more upfront an item comes in the sort order.

Returns

Returns the rank of the current `m_status` byte.

10.9.5 Field Documentation

10.9.5.1 m_timestamp

```
midipulse seq64::event::m_timestamp [private]
```

10.9.5.2 m_status

```
midibyte seq64::event::m_status [private]
```

The channel is included when recording MIDI, but, once a sequence with a matching channel is found, the channel nybble is cleared for storage. The channel will be added back on the MIDI bus upon playback. The high nybble = type of event; The low nybble = channel. Bit 7 is present in all status bytes.

Note that, for status values of 0xF0 (Sysex) or 0xFF (Meta), special handling of the event can occur. We would like to eventually use inheritance to keep the event class simple. For now, search for "tempo" and "sysex" to tease out their implementations. Sigh.

10.9.5.3 m_channel

```
midibyte seq64::event::m_channel [private]
```

This member adds another 4 bytes to the event object, most likely.

Overload: For Meta events, where [is_meta\(\)](#) is true, this value holds the type of Meta event. See the [editable ↗ event::sm_meta_event_names\[\]](#) array. Note that EVENT_META_ILLEGAL (0xFF) indicates an illegal Meta event.

10.9.5.4 m_data

```
midibyte seq64::event::m_data[SEQ64_MIDI_DATA_BYTE_COUNT] [private]
```

Remember that the most-significant bit of a data byte is always 0. A one-byte message uses only the 0th index.

10.9.5.5 m_sysex

```
SysexContainer seq64::event::m_sysex [private]
```

Adapted from Stazed's Seq32 project on GitHub. This object will also hold the generally small amounts of data needed for Meta events. Compare [is_sysex\(\)](#) to [is_meta\(\)](#) and [is_ex_data\(\)](#) [which tests for both].

10.9.5.6 m_linked

```
event* seq64::event::m_linked [private]
```

10.9.5.7 m_has_link

```
bool seq64::event::m_has_link [private]
```

This item is used [via the `get_link()` and [link\(\)](#) accessors] in the sequence class.

10.9.5.8 m_selected

```
bool seq64::event::m_selected [private]
```

10.9.5.9 m_marked

```
bool seq64::event::m_marked [private]
```

10.9.5.10 m_painted

```
bool seq64::event::m_painted [private]
```

10.10 seq64::event_list::event_key Class Reference

Provides a key value for an event map.

Public Member Functions

- [event_key](#) (midipulse tstamp, int rank)
Principal [event_key](#) constructor.
- [event_key](#) (const [event](#) &e)
Event-based constructor.
- bool [operator<](#) (const [event_key](#) &rhs) const
Provides the minimal operator needed to sort events using an [event_key](#).

Private Attributes

- [midipulse m_timestamp](#)
The primary key-value for the key.
- int [m_rank](#)
The sub-key-value for the key.

10.10.1 Detailed Description

Its types match the `m_timestamp` and `get_rank()` function of this event class.

10.10.2 Constructor & Destructor Documentation

10.10.2.1 `event_key()` [1/2]

```
seq64::event_list::event_key::event_key (
    midipulse tstamp,
    int rank )
```

Parameters

<i>tstamp</i>	The time-stamp is the primary part of the key. It is the most important key item.
<i>rank</i>	Rank is an arbitrary number used to prioritize events that have the same time-stamp. See the event::get_rank() function for more information.

10.10.2.2 `event_key()` [2/2]

```
seq64::event_list::event_key::event_key (
    const event & rhs )
```

This constructor makes it even easier to create an [event_key](#). Note that the call to [event::get_rank\(\)](#) makes a simple calculation based on the status of the event.

Parameters

<i>rhs</i>	Provides the event key to be copied.
------------	--------------------------------------

10.10.3 Member Function Documentation

10.10.3.1 `operator<()`

```
bool seq64::event_list::event_key::operator< (
    const event\_key & rhs ) const
```

Parameters

<i>rhs</i>	Provides the event key to be compared against.
------------	--

Returns

Returns true if the rank and timestamp of the current object are less than those of *rhs*.

10.10.4 Field Documentation

10.10.4.1 m_timestamp

`midipulse seq64::event_list::event_key::m_timestamp [private]`

10.10.4.2 m_rank

`int seq64::event_list::event_key::m_rank [private]`

10.11 seq64::event_list Class Reference

The `event_list` class is a receptable for MIDI events.

Data Structures

- class `event_key`

Provides a key value for an event map.

Public Types

- typedef std::multimap< `event_key`, `event` > `Events`
Types to use to swap between list and multimap implementations.
- typedef std::pair< `event_key`, `event` > `EventsPair`
- typedef Events::iterator `iterator`
- typedef Events::const_iterator `const_iterator`
- typedef Events::reverse_iterator `reverse_iterator`
- typedef Events::const_reverse_iterator `const_reverse_iterator`

Public Member Functions

- [event_list](#) ()
Principal constructor.
- [event_list](#) (const [event_list](#) &a_rhs)
Copy constructor.
- [event_list](#) & [operator=](#) (const [event_list](#) &a_rhs)
Principal assignment operator.
- [~event_list](#) ()
A rote destructor.
- [iterator begin](#) ()
'Getter' function for member m_events.begin(), non-constant version.
- [const_iterator begin](#) () const
'Getter' function for member m_events.begin(), constant version.
- [iterator end](#) ()
'Getter' function for member m_events.end(), non-constant version.
- [const_iterator end](#) () const
'Getter' function for member m_events.end(), constant version.
- int [count](#) () const
Returns the number of events stored in m_events.
- [midipulse get_length](#) () const
Provides the length of the events in MIDI pulses.
- bool [empty](#) () const
Returns true if there are no events.
- bool [add](#) (const [event](#) &e)
Adds an event to the internal event list in an optionally sorted manner.
- bool [append](#) (const [event](#) &e)
Adds an event to the internal event list without sorting.
- void [push_back](#) (const [event](#) &)
The multimap version of this function does nothing.
- bool [is_modified](#) () const
'Getter' function for member m_is_modified
- bool [has_tempo](#) () const
'Getter' function for member m_has_tempo
- bool [has_time_signature](#) () const
'Getter' function for member m_has_time_signature
- void [unmodify](#) ()
'Setter' function for member m_is_modified This function may be needed by some of the sequence editors.
- void [remove](#) ([iterator](#) ie)
Provides a wrapper for the iterator form of erase(), which is the only one that sequence uses.
- void [clear](#) ()
Provides a wrapper for clear().
- void [merge](#) ([event_list](#) &el, bool presort=true)
Provides a merge operation for the event multimap analogous to the merge operation for the event list.
- void [sort](#) ()
Sorts the event list; active only for the std::list implementation.

Static Public Member Functions

- static [event](#) & [dref](#) ([iterator](#) ie)
Dereference access for list or map.
- static const [event](#) & [dref](#) (const [iterator](#) ie)
Dereference const access for list or map.

Private Member Functions

- void [link_new](#) ()
Links a new event.
- void [clear_links](#) ()
Clears all event links and unmarks them all.
- void [verify_and_link](#) (midipulse slength)
This function verifies state: all note-ons have an off, and it links note-offs with their note-ons.
- void [link_tempos](#) ()
This function tries to link tempo events.
- void [clear_tempo_links](#) ()
Clears all tempo event links.
- bool [mark_selected](#) ()
Marks all selected events.
- void [mark_out_of_range](#) (midipulse slength)
Marks all events that have a time-stamp that is out of range.
- void [mark_all](#) ()
Marks all events.
- void [unmark_all](#) ()
Unmarks all events.
- bool [remove_marked](#) ()
Removes marked events.
- void [unpaint_all](#) ()
Unpaints all list-events.
- int [count_selected_notes](#) () const
Counts the selected note-on events in the event list.
- bool [any_selected_notes](#) () const
Indicates that at least one note is selected.
- int [count_selected_events](#) (midibyte status, midibyte cc) const
Counts the selected events, with the given status, in the event list.
- void [select_all](#) ()
Selects all events, unconditionally.
- void [unselect_all](#) ()
Deselects all events, unconditionally.
- void [print](#) () const
Prints a list of the currently-held events.
- const [Events](#) & [events](#) () const
'Getter' function for member m_events

Private Attributes

- [Events](#) [m_events](#)
This list holds the current pattern/sequence events.
- bool [m_is_modified](#)
A flag to indicate if an event was added or removed.
- bool [m_has_tempo](#)
A new flag to indicate that a tempo event has been added.
- bool [m_has_time_signature](#)
A new flag to indicate that a time-signature event has been added.

Friends

- class [editable_events](#)
- class [midifile](#)
- class [sequence](#)

10.11.1 Detailed Description

Two implementations, an `std::multimap`, and the original, an `std::list`, are provided for comparison, and are selected at build time, by manually defining the `SEQ64_USE_EVENT_MAP` macro near the top of this module.

10.11.2 Member Typedef Documentation

10.11.2.1 Events

```
typedef std::multimap<event_key, event> seq64::event_list::Events
```

10.11.2.2 EventsPair

```
typedef std::pair<event_key, event> seq64::event_list::EventsPair
```

10.11.2.3 iterator

```
typedef Events::iterator seq64::event_list::iterator
```

10.11.2.4 const_iterator

```
typedef Events::const_iterator seq64::event_list::const_iterator
```

10.11.2.5 reverse_iterator

```
typedef Events::reverse_iterator seq64::event_list::reverse_iterator
```

10.11.2.6 const_reverse_iterator

```
typedef Events::const_reverse_iterator seq64::event_list::const_reverse_iterator
```

10.11.3 Constructor & Destructor Documentation

10.11.3.1 event_list() [1/2]

```
seq64::event_list::event_list ( )
```

10.11.3.2 event_list() [2/2]

```
seq64::event_list::event_list (
    const event_list & rhs )
```

Parameters

<i>rhs</i>	Provides the event list to be copied.
------------	---------------------------------------

10.11.3.3 ~event_list()

```
seq64::event_list::~~event_list ( )
```

10.11.4 Member Function Documentation

10.11.4.1 operator=()

```
event_list & seq64::event_list::operator= (
    const event_list & rhs )
```

Follows the stock rules for such an operator, just assigning member values.

Parameters

<i>rhs</i>	Provides the event list to be assigned.
------------	---

10.11.4.2 begin() [1/2]

```
iterator seq64::event_list::begin ( ) [inline]
```

10.11.4.3 begin() [2/2]

```
const_iterator seq64::event_list::begin ( ) const [inline]
```

10.11.4.4 end() [1/2]

```
iterator seq64::event_list::end ( ) [inline]
```

10.11.4.5 end() [2/2]

```
const_iterator seq64::event_list::end ( ) const [inline]
```

10.11.4.6 count()

```
int seq64::event_list::count ( ) const [inline]
```

We like returning an integer instead of `size_t`, and rename the function so nobody is fooled.

10.11.4.7 get_length()

```
midipulse seq64::event_list::get_length ( ) const
```

This function gets the iterator for the last element and returns its length value.

Returns

Returns the timestamp of the latest event in the container.

10.11.4.8 empty()

```
bool seq64::event_list::empty ( ) const [inline]
```

```
return m_events.size() == 0;
```

10.11.4.9 add()

```
bool seq64::event_list::add (
    const event & e ) [inline]
```

Parameters

<code>e</code>	Provides the event to be added to the list.
----------------	---

Returns

Returns true. We assume the insertion succeeded, and no longer care about an increment in container size. It's a multimap, so it always inserts, and if we don't have memory left, all bets are off anyway.

10.11.4.10 `append()`

```
bool seq64::event_list::append (
    const event & e )
```

It is a wrapper, wrapper for `insert()` or `push_front()`, with an option to call `sort()`.

The `add()` function without sorting, useful to speed up the initial container loading into the event-list.

For the `std::multimap` implementation, This is an option if we want to make sure the insertion succeed.

If the `std::list` implementation has been built in, then the event list is not sorted after the addition. This is a time-consuming operation.

We also have to raise some new flags if the event is a Set Tempo or Time Signature event, so that we do not force the current tempo and time-signature when writing the MIDI file.

Warning

This pushing (and, in writing the MIDI file, the popping), causes events with identical timestamps to be written in reverse order. Doesn't affect functionality, but it's puzzling until one understands what is happening. That's why we're now preferring to use a multimap as the container.

Parameters

<code>e</code>	Provides the event to be added to the list.
----------------	---

Returns

Returns true. We assume the insertion succeeded, and no longer care about an increment in container size. It's a multimap, so it always inserts, and if we don't have memory left, all bets are off anyway.

10.11.4.11 `push_back()`

```
void seq64::event_list::push_back (
    const event & ) [inline]
```

10.11.4.12 is_modified()

```
bool seq64::event_list::is_modified ( ) const [inline]
```

10.11.4.13 has_tempo()

```
bool seq64::event_list::has_tempo ( ) const [inline]
```

10.11.4.14 has_time_signature()

```
bool seq64::event_list::has_time_signature ( ) const [inline]
```

10.11.4.15 unmodify()

```
void seq64::event_list::unmodify ( ) [inline]
```

But use it with great caution.

10.11.4.16 remove()

```
void seq64::event_list::remove (
    iterator ie ) [inline]
```

Currently, no check on removal is performed. Sets the modified-flag.

Parameters

<i>ie</i>	Provides the iterator to the event to be removed.
-----------	---

10.11.4.17 clear()

```
void seq64::event_list::clear ( ) [inline]
```

Sets the modified-flag.

10.11.4.18 merge()

```
void seq64::event_list::merge (
    event_list & el,
    bool presort = true )
```

We have certain constraints to preserve, as the following discussion shows.

For `std::list`, sequence merges list T into list A by first calling `T.sort()`, and then `A.merge(T)`. The `merge()` operation merges T into A by transferring all of its elements, at their respective ordered positions, into A. Both containers must already be ordered.

The merge effectively removes all the elements in T (which becomes empty), and inserts them into their ordered position within container (which expands in size by the number of elements transferred). The operation is performed without constructing nor destroying any element, whether T is an lvalue or an rvalue, or whether the value-type supports move-construction or not.

Each element of T is inserted at the position that corresponds to its value according to the strict weak ordering defined by operator `<`. The resulting order of equivalent elements is stable (i.e. equivalent elements preserve the relative order they had before the call, and existing elements precede those equivalent inserted from x). The function does nothing if `(&x == this)`.

For `std::multimap`, sorting is automatic. However, unless move-construction is supported, merging will be less efficient than for the list version. Also, we need a way to include duplicates of each event, so we need to use a multimap. Once all this setup, merging is really just insertion. And, since sorting isn't needed, the multimap actually turns out to be faster.

Parameters

<i>el</i>	Provides the event list to be merged into the current event list.
<i>presort</i>	If true, the events are presorted. This is a requirement for merging an <code>std::list</code> , but is a no-op for the <code>std::multimap</code> implementation.

10.11.4.19 sort()

```
void seq64::event_list::sort ( ) [inline]
```

10.11.4.20 dref() [1/2]

```
static event& seq64::event_list::dref (
    iterator ie ) [inline], [static]
```

Parameters

<i>ie</i>	Provides the iterator to the event to which to get a reference.
-----------	---

10.11.4.21 dref() [2/2]

```
static const event& seq64::event_list::dref (
    const_iterator ie ) [inline], [static]
```

Parameters

<i>ie</i>	Provides the iterator to the event to which to get a reference.
-----------	---

10.11.4.22 link_new()

```
void seq64::event_list::link_new ( ) [private]
```

This function checks for a note on, then look for its note off. This function is provided in the [event_list](#) because it does not depend on any external data. Also note that any desired thread-safety must be provided by the caller.

10.11.4.23 clear_links()

```
void seq64::event_list::clear_links ( ) [private]
```

10.11.4.24 verify_and_link()

```
void seq64::event_list::verify_and_link (
    midipulse slength ) [private]
```

Stazed (seq32):

This function now deletes any notes that are \geq m_length, so any resize or move of notes must modify for wrapping if Note Off is \geq m_length.

Not threadsafe As in most case, the caller will use an automutex to call this function safely.

Parameters

<i>slength</i>	Provides the length beyond which events will be pruned.
----------------	---

10.11.4.25 link_tempos()

```
void seq64::event_list::link_tempos ( ) [private]
```


Native support for tempo tracks is a new feature of [seq64](#). These links are only in one direction: forward in time, to the next tempo event, if any.

Also, at present, tempo events are not markable.

Not threadsafe As in most case, the caller will use an automutex to call this function safely.

10.11.4.26 clear_tempo_links()

```
void seq64::event_list::clear_tempo_links ( ) [private]
```

10.11.4.27 mark_selected()

```
bool seq64::event_list::mark_selected ( ) [private]
```

Returns

Returns true if there was even one event selected and marked.

10.11.4.28 mark_out_of_range()

```
void seq64::event_list::mark_out_of_range (
    midipulse slength ) [private]
```

Used for killing (pruning) those events not in range. If the current time-stamp is greater than the length, then the event is marked for pruning.

Note

This code was comparing the timestamp as greater than or equal to the sequence length. However, being equal is fine. This may explain why the midifile code would add one tick to the length of the last note when processing the end-of-track.

Parameters

<i>slength</i>	Provides the length beyond which events will be pruned.
----------------	---

10.11.4.29 mark_all()

```
void seq64::event_list::mark_all ( ) [private]
```

Not yet used, but might come in handy with the event editor dialog.

10.11.4.30 unmark_all()

```
void seq64::event_list::unmark_all ( ) [private]
```

10.11.4.31 remove_marked()

```
bool seq64::event_list::remove_marked ( ) [private]
```

Note how this function handles removing a value to avoid incrementing a now-invalid iterator.

Threadsafe

Returns

Returns true if at least one event was removed.

10.11.4.32 unpaint_all()

```
void seq64::event_list::unpaint_all ( ) [private]
```

10.11.4.33 count_selected_notes()

```
int seq64::event_list::count_selected_notes ( ) const [private]
```

10.11.4.34 any_selected_notes()

```
bool seq64::event_list::any_selected_notes ( ) const [private]
```

Acts like [event_list::count_selected_notes\(\)](#), but stops after finding a selected note.

Returns

Returns true if at least one note is selected.

10.11.4.35 count_selected_events()

```
int seq64::event_list::count_selected_events (
    midibyte status,
    midibyte cc ) const [private]
```

If the event is a control change (CC), then it must also match the given CC value. The one exception is tempo events, which are always selectable.

Parameters

<i>status</i>	The desired status value to count.
<i>cc</i>	The desired control-change to count. Used only if the status parameter indicates a control-change event.

Returns

Returns the number of selected events.

10.11.4.36 select_all()

```
void seq64::event_list::select_all ( ) [private]
```

10.11.4.37 unselect_all()

```
void seq64::event_list::unselect_all ( ) [private]
```

10.11.4.38 print()

```
void seq64::event_list::print ( ) const [private]
```

Useful for debugging.

10.11.4.39 events()

```
const Events& seq64::event_list::events ( ) const [inline], [private]
```

10.11.5 Friends And Related Function Documentation

10.11.5.1 editable_events

```
friend class editable_events [friend]
```

10.11.5.2 midifile

```
friend class midifile [friend]
```

10.11.5.3 sequence

```
friend class sequence [friend]
```

10.11.6 Field Documentation

10.11.6.1 m_events

```
Events seq64::event_list::m_events [private]
```

10.11.6.2 m_is_modified

```
bool seq64::event_list::m_is_modified [private]
```

We may need to give client code a way to reload the sequence. This is currently an issue when a seqroll and an eventedit/eventslots are active for the same sequence.

10.11.6.3 m_has_tempo

```
bool seq64::event_list::m_has_tempo [private]
```

Legacy behavior forces the tempo to be written to the track-0 sequence, but we don't want to do that if the MIDI file (or the current event list) contains a tempo event.

10.11.6.4 m_has_time_signature

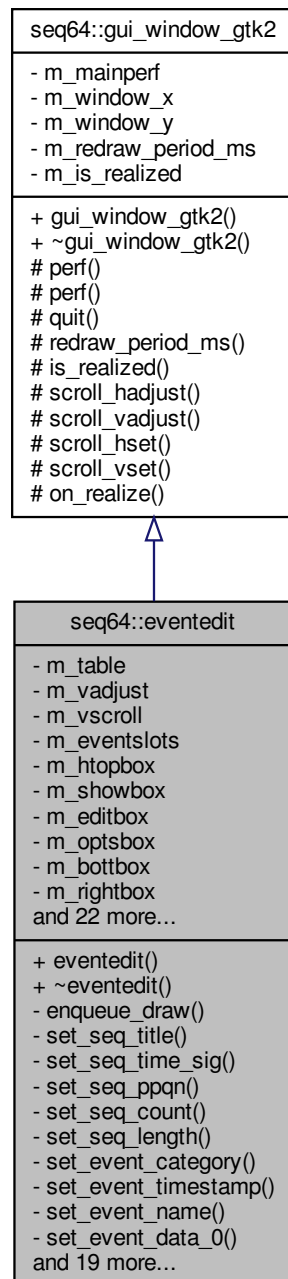
```
bool seq64::event_list::m_has_time_signature [private]
```

Legacy behavior forces the time-signature to be written to the track-0 sequence, but we don't want to do that if the MIDI file (or the current event list) contains a time-signature event.

10.12 seq64::eventedit Class Reference

This class supports an Event Editor that is used to tweak the details of events and get a better idea of the mix of events in a sequence.

Inheritance diagram for seq64::eventedit:



Public Member Functions

- [eventedit](#) ([perform](#) &p, [sequence](#) &seq)

Principal constructor, has a reference to a perform object.

- virtual [~eventedit](#) ()

This rote constructor does nothing.

Private Member Functions

- void [enqueue_draw](#) ()
Helper wrapper for calling `eventslots::queue_draw()`.
- void [set_seq_title](#) (const std::string &title)
Sets `m_label_seq_name` to the title.
- void [set_seq_time_sig](#) (const std::string &sig)
Sets `m_label_time_sig` to the time-signature string.
- void [set_seq_ppqn](#) (const std::string &p)
Sets `m_label_ppqn` to the parts-per-quarter-note string.
- void [set_seq_count](#) ()
Sets `m_label_ev_count` to the number-of-events string.
- void [set_seq_length](#) ()
Sets `m_label_seq_length` to the number-of-events string.
- void [set_event_category](#) (const std::string &c)
Sets `m_label_category` to the category string.
- void [set_event_timestamp](#) (const std::string &ts)
Sets `m_entry_ev_timestamp` to the time-stamp string.
- void [set_event_name](#) (const std::string &n)
Sets `m_entry_ev_name` to the name-of-event string.
- void [set_event_data_0](#) (const std::string &d)
Sets `m_entry_ev_data_0` to the first data byte string.
- void [set_event_data_1](#) (const std::string &d)
Sets `m_entry_data_1` to the second data byte string.
- void [perf_modify](#) ()
Provides a way to mark the perform object as modified, when the modified sequence is saved.
- void [set_dirty](#) (bool flag=true)
Sets the "modified" status of the user-interface.
- void [v_adjustment](#) (int value)
Sets the parameters for the vertical scroll-bar, using only the value parameter.
- void [v_adjustment](#) (int value, int lower, int upper)
Sets the parameters for the vertical scroll-bar that is associated with the eventslots event-list user-interface.
- void [change_focus](#) (bool set_it=true)
Changes what perform and mainwid see as the "current sequence".
- void [close_out](#) ()
Handles closing the sequence editor, common code for [handle_cancel\(\)](#) and [handle_close\(\)](#).
- void [handle_close](#) ()
Handles closing the sequence editor.
- void [handle_delete](#) ()
Initiates the deletion of the current editable event.
- void [handle_insert](#) ()
Initiates the insertion of a new editable event.
- void [handle_modify](#) ()
Passes the edited fields to the current editable event in the eventslot.
- void [handle_save](#) ()
Handles saving the edited data back to the original sequence.

- void `handle_cancel` ()
Cancels the edits and closes the dialog box.
- void `on_realize` ()
This callback function calls the base-class `on_realize()` function.
- void `on_set_focus` (Widget *focus)
On receiving focus, attempt to tell mainwid that this sequence is now the current sequence.
- bool `on_focus_in_event` (GdkEventFocus *)
Implements the on-focus event handling.
- bool `on_focus_out_event` (GdkEventFocus *)
Implements the on-unfocus event handling.
- bool `on_key_press_event` (GdkEventKey *ev)
This function is the callback for a key-press event.
- bool `on_delete_event` (GdkEventAny *event)
Handles an on-delete event.

Private Attributes

- Gtk::Table * `m_table`
A whole horde of GUI elements.
- Gtk::Adjustment * `m_vadjust`
Vertical paging for event list.
- Gtk::VScrollbar * `m_vscroll`
Vertical scroll for event list.
- `eventslots` * `m_eventslots`
Drawing area for events.
- Gtk::HBox * `m_htopbox`
Padding at the top of the dialog.
- Gtk::VBox * `m_showbox`
Area for sequence information.
- Gtk::VBox * `m_editbox`
Text-edits and buttons for data.
- Gtk::VBox * `m_optsbox`
Reserved for future options.
- Gtk::HBox * `m_bottbox`
Holds the Save and Close buttons.
- Gtk::VBox * `m_rightbox`
Used for padding on right side.
- Gtk::Button * `m_button_del`
"Delete Current Event ()" button.*
- Gtk::Button * `m_button_ins`
"Insert New Event" button.
- Gtk::Button * `m_button_modify`
"Modify New Event" button.
- Gtk::Button * `m_button_save`
"Save to Sequence" button.
- Gtk::Button * `m_button_cancel`
"Close" button.
- Gtk::Label * `m_label_seq_name`
Items for the inside of the m_showbox member.
- Gtk::Label * `m_label_time_sig`

- Shows time signature for pattern.*

 - Gtk::Label * [m_label_ppqn](#)

Shows the parts per quarter note.
- Gtk::Label * [m_label_channel](#)

Shows channel number of pattern.
- Gtk::Label * [m_label_ev_count](#)

Shows the count of pattern events.
- Gtk::Label * [m_label_seq_length](#)

Shows the length of the pattern.
- Gtk::Label * [m_label_spacer](#)

Spacer for the showbox elements.
- Gtk::Label * [m_label_modified](#)

Shows "[Modified]" if edited.
- Gtk::Label * [m_label_category](#)

Items for the inside of the m_editbox member.
- Gtk::Entry * [m_entry_ev_timestamp](#)

Text edit for event time-stamp.
- Gtk::Entry * [m_entry_ev_name](#)

Text edit for MIDI event name.
- Gtk::Entry * [m_entry_ev_data_0](#)

Text edit for first event datum.
- Gtk::Entry * [m_entry_ev_data_1](#)

Text edit for second event datum.
- Gtk::Label * [m_label_time_fmt](#)

Optsbox item, only "Sequencer64".
- Gtk::Label * [m_label_right](#)

Padding at the right of dialog.
- [sequence](#) & [m_seq](#)

A reference to the sequence being edited, to control its editing flag.
- bool [m_have_focus](#)

Indicates that the focus has already been changed to this sequence.

Friends

- class [eventslots](#)

Additional Inherited Members

10.12.1 Constructor & Destructor Documentation

10.12.1.1 eventedit()

```
seq64::eventedit::eventedit (
    perform & p,
    sequence & seq )
```

We've reordered the pointer members and put them in the initializer list to make the constructor a bit cleaner.

Adjustment parameters:

value	initial value
lower	minimum value
upper	maximum value
step_increment	step increment
page_increment	page increment
page_size	page size

Table constructor parameters:

```
rows
columns
homogenous
```

Table attach() parameters:

child	widget to add.
left_attach	column number to attach left side of a child widget
right_attach	column number to attach right side of a child widget
top_attach	row number to attach the top of a child widget
bottom_attach	row number to attach the bottom of a child widget
xoptions	properties of the child widget when table resized
yoptions	same as xoptions, except vertical.
xpadding	padding on L and R of widget added to table
ypadding	amount of padding above and below the child widget

Layout:

```

0          1  2          3  4          0
-----
htop | (OLD LAYOUT)          :  :          |
e'slots | 1-120:0:192 Program Change | ^ | "Sequence name"          | r | 1
|-----| 4/4 PPQN 192          | r | 2
| 2-120:1:0   Program Change | s | 9999 events          | i | 3
|-----| ----- editbox -----| g | 4
| ...   ...          ...   | r | Channel Event: Ch. 5   | h |
| ...   ...          ...   | o | -----| t | 6
| ...   ...          ...   | l | [Edit field: Note On ] | |
| ...   ...          ...   | l | -----| b | 7
| ...   ...          ...   |   | [Edit field: Key #   ] | o |
| ...   ...          ...   | b | -----| x | 8
| ...   ...          ...   | a | [Edit field: Vel #   ] | |
| ...   ...          ...   | r | -----| | 9
| ...   ...          ...   |   | [Optional more data? ] | |
| ...   ...          ...   |   | -----| | 10
| ...   ...          ...   |   | o Pulses          | |
| ...   ...          ...   |   | o Measures          | |
| ...   ...          ...   | v | o Time          | |
|-----| ----- bottbox -----| | 13
| 56-136:3:133 Program Change | v | | Save | | Close | | |
-----|-----|-----|-----| 14
```

Parameters

<i>p</i>	Refers to the main performance object.
<i>seq</i>	Refers to the sequence holding the event data to be edited.

The seqedit class indirectly sets the sequence dirty flags, and this allows the sequence's pattern slot to be updated, which, for example, allows the new optional in-edit-highlight feature to work. To get the eventedit to also show the in-edit highlighting, we can make the [sequence::set_dirty_mp\(\)](#) call. This call does not cause a prompt for saving the file when exiting.

10.12.1.2 ~eventedit()

```
seq64::eventedit::~~eventedit ( ) [virtual]
```

We're going to have to run the application through valgrind to make sure that nothing is left behind.

10.12.2 Member Function Documentation**10.12.2.1 enqueue_draw()**

```
void seq64::eventedit::enqueue_draw ( ) [private]
```

10.12.2.2 set_seq_title()

```
void seq64::eventedit::set_seq_title (
    const std::string & title ) [private]
```

Parameters

<i>title</i>	The name of the sequence.
--------------	---------------------------

10.12.2.3 set_seq_time_sig()

```
void seq64::eventedit::set_seq_time_sig (
    const std::string & sig ) [private]
```

Parameters

<i>sig</i>	The time signature of the sequence.
------------	-------------------------------------

10.12.2.4 set_seq_ppqn()

```
void seq64::eventedit::set_seq_ppqn (
    const std::string & p ) [private]
```

Parameters

<i>p</i>	The parts-per-quarter-note string for the sequence.
----------	---

10.12.2.5 set_seq_count()

```
void seq64::eventedit::set_seq_count ( ) [private]
```

10.12.2.6 set_seq_length()

```
void seq64::eventedit::set_seq_length ( ) [private]
```

```
m_eventslots->seq().calculate_measures()
```

10.12.2.7 set_event_category()

```
void seq64::eventedit::set_event_category (
    const std::string & c ) [private]
```

Parameters

<i>c</i>	The category string for the current event.
----------	--

10.12.2.8 set_event_timestamp()

```
void seq64::eventedit::set_event_timestamp (
    const std::string & ts ) [private]
```

Parameters

<i>ts</i>	The time-stamp string for the current event.
-----------	--

10.12.2.9 `set_event_name()`

```
void seq64::eventedit::set_event_name (
    const std::string & n ) [private]
```

Parameters

<i>n</i>	The name-of-event string for the current event.
----------	---

10.12.2.10 `set_event_data_0()`

```
void seq64::eventedit::set_event_data_0 (
    const std::string & d ) [private]
```

Parameters

<i>d</i>	The first data byte string for the current event.
----------	---

10.12.2.11 `set_event_data_1()`

```
void seq64::eventedit::set_event_data_1 (
    const std::string & d ) [private]
```

Parameters

<i>d</i>	The second data byte string for the current event.
----------	--

10.12.2.12 `perf_modify()`

```
void seq64::eventedit::perf_modify ( ) [private]
```

10.12.2.13 `set_dirty()`

```
void seq64::eventedit::set_dirty (
    bool flag = true ) [private]
```

This includes changing a label, enabling/disabling the Save button, and modifying the event count and sequence length (in measures).

Parameters

<i>flag</i>	If true, the modified status is indicated, otherwise it is cleared.
-------------	---

10.12.2.14 v_adjustment() [1/2]

```
void seq64::eventedit::v_adjustment (
    int value ) [private]
```

This function overload provides a common use case.

Parameters

<i>value</i>	The new current value to be indicated by the scroll-bar.
--------------	--

10.12.2.15 v_adjustment() [2/2]

```
void seq64::eventedit::v_adjustment (
    int value,
    int lower,
    int upper ) [private]
```

It keeps the frame scroll-bar in sync with the frame movement actions. Some of the parameters are obtained from the eventslots object:

- Page size comes from eventslots::line_maximum().
- Page increment is a little less than the page-size value.

Parameters

<i>value</i>	The current value to be indicated by the scroll-bar. It will lie between the lower and upper parameter.
<i>lower</i>	The lowest value to be indicated by the scroll-bar.
<i>upper</i>	The highest value to be indicated by the scroll-bar.

10.12.2.16 change_focus()

```
void seq64::eventedit::change_focus (
    bool set_it = true ) [private]
```

Similar to the same function in seqedit.

Parameters

<code>set_it</code>	If true (the default value), indicates we want focus, otherwise we want to give up focus.
---------------------	---

10.12.2.17 close_out()

```
void seq64::eventedit::close_out ( ) [private]
```

10.12.2.18 handle_close()

```
void seq64::eventedit::handle_close ( ) [private]
```

Simply calls [close_out\(\)](#).

10.12.2.19 handle_delete()

```
void seq64::eventedit::handle_delete ( ) [private]
```

10.12.2.20 handle_insert()

```
void seq64::eventedit::handle_insert ( ) [private]
```

The event's location will be determined by the timestamp and existing events. Note that we have to recalibrate the scroll-bar when we insert/delete events by calling [v_adjustment\(\)](#).

As a new feature, we will allow events to extend the official length of the sequence.

10.12.2.21 handle_modify()

```
void seq64::eventedit::handle_modify ( ) [private]
```

Note that there are two cases to worry about. If the timestamp has not changed, then we can simply modify the existing current event in place. Otherwise, we need to delete the old event and insert the new one. But that is done for us by [eventslots::modify_current_event\(\)](#).

10.12.2.22 handle_save()

```
void seq64::eventedit::handle_save ( ) [private]
```

The event list in the original sequence is cleared, and the editable events are converted to plain events, and added to the container, one by one.

Todo Could also support writing the events to a new sequence, for added flexibility.

10.12.2.23 handle_cancel()

```
void seq64::eventedit::handle_cancel ( ) [private]
```

In order for removing the current-highlighting in the mainwd or perfedit windows, some of the work of [handle_close\(\)](#) needs to be done here as well.

10.12.2.24 on_realize()

```
void seq64::eventedit::on_realize ( ) [private]
```

Then it sets the vertical adjustment to account for the number of events in the eventslot.

10.12.2.25 on_set_focus()

```
void seq64::eventedit::on_set_focus (
    Widget * focus ) [private]
```

Only works in certain circumstances.

Parameters

<i>focus</i>	The widget that has the focus. Merely passed on to gui_window_gtk2 's version of this function.
--------------	---

10.12.2.26 on_focus_in_event()

```
bool seq64::eventedit::on_focus_in_event (
    GdkEventFocus * ) [private]
```

It sets the focus flag and calls [change_focus\(\)](#).

10.12.2.27 on_focus_out_event()

```
bool seq64::eventedit::on_focus_out_event (
    GdkEventFocus * ) [private]
```

It resets the focus flag and calls [change_focus\(\)](#).

10.12.2.28 on_key_press_event()

```
bool seq64::eventedit::on_key_press_event (
    GdkEventKey * ev ) [private]
```

If the Up or Down arrow is pressed (later, k and j :-), then we tell the eventslots object to move the "current event" highlighting up or down. In Gtkmm, these arrows also cause movement from one edit field to the next, so we disable that process if the event was handled here.

Note that the Delete key is needed for the edit fields. For now, we replace it with the asterisk, which is easy to access from the numeric pad of a keyboard, and allows for rapid deletion. The Insert key also causes confusing effects in the edit fields, so we replaced it by the slash, but that didn't work. Note that the asterisk and slash should not be required in any of the edit fields. HOWEVER, "/" still gets passed the edit fields (!), so you'll just have to click the button to insert an event. Let's try the backslash! No go there, either.

Parameters

<i>ev</i>	The key event to process.
-----------	---------------------------

Returns

Returns true if the event got handled somewhere along the line.

10.12.2.29 on_delete_event()

```
bool seq64::eventedit::on_delete_event (
    GdkEventAny * event ) [private]
```

It sets the sequence object's editing flag to false, and deletes "this". This function is called if the "Close" ("X") button in the window's title bar is clicked. That is a different action from clicking the Close button.

Returns

Always returns false.

10.12.3 Friends And Related Function Documentation

10.12.3.1 eventslots

```
friend class eventslots [friend]
```

10.12.4 Field Documentation

10.12.4.1 m_table

```
Gtk::Table* seq64::eventedit::m_table [private]
```

Provides the layout table for UI.

10.12.4.2 m_vadjust

```
Gtk::Adjustment* seq64::eventedit::m_vadjust [private]
```

10.12.4.3 m_vscroll

```
Gtk::VScrollbar* seq64::eventedit::m_vscroll [private]
```

10.12.4.4 m_eventslots

```
eventslots* seq64::eventedit::m_eventslots [private]
```

10.12.4.5 m_htopbox

```
Gtk::HBox* seq64::eventedit::m_htopbox [private]
```

10.12.4.6 m_showbox

```
Gtk::VBox* seq64::eventedit::m_showbox [private]
```

10.12.4.7 m_editbox

Gtk::VBox* seq64::eventedit::m_editbox [private]

10.12.4.8 m_optsbox

Gtk::VBox* seq64::eventedit::m_optsbox [private]

10.12.4.9 m_bottbox

Gtk::HBox* seq64::eventedit::m_bottbox [private]

10.12.4.10 m_rightbox

Gtk::VBox* seq64::eventedit::m_rightbox [private]

10.12.4.11 m_button_del

Gtk::Button* seq64::eventedit::m_button_del [private]

10.12.4.12 m_button_ins

Gtk::Button* seq64::eventedit::m_button_ins [private]

10.12.4.13 m_button_modify

Gtk::Button* seq64::eventedit::m_button_modify [private]

10.12.4.14 m_button_save

Gtk::Button* seq64::eventedit::m_button_save [private]

10.12.4.15 m_button_cancel

Gtk::Button* seq64::eventedit::m_button_cancel [private]

10.12.4.16 m_label_seq_name

Gtk::Label* seq64::eventedit::m_label_seq_name [private]

Shows the name of the pattern.

10.12.4.17 m_label_time_sig

Gtk::Label* seq64::eventedit::m_label_time_sig [private]

10.12.4.18 m_label_ppqn

Gtk::Label* seq64::eventedit::m_label_ppqn [private]

10.12.4.19 m_label_channel

Gtk::Label* seq64::eventedit::m_label_channel [private]

10.12.4.20 m_label_ev_count

Gtk::Label* seq64::eventedit::m_label_ev_count [private]

10.12.4.21 m_label_seq_length

Gtk::Label* seq64::eventedit::m_label_seq_length [private]

10.12.4.22 m_label_spacer

Gtk::Label* seq64::eventedit::m_label_spacer [private]

10.12.4.23 m_label_modified

Gtk::Label* seq64::eventedit::m_label_modified [private]

10.12.4.24 m_label_category

Gtk::Label* seq64::eventedit::m_label_category [private]

Shows the type of MIDI event.

10.12.4.25 m_entry_ev_timestamp

Gtk::Entry* seq64::eventedit::m_entry_ev_timestamp [private]

10.12.4.26 m_entry_ev_name

Gtk::Entry* seq64::eventedit::m_entry_ev_name [private]

10.12.4.27 m_entry_ev_data_0

Gtk::Entry* seq64::eventedit::m_entry_ev_data_0 [private]

10.12.4.28 m_entry_ev_data_1

Gtk::Entry* seq64::eventedit::m_entry_ev_data_1 [private]

10.12.4.29 m_label_time_fmt

Gtk::Label* seq64::eventedit::m_label_time_fmt [private]

10.12.4.30 m_label_right

Gtk::Label* seq64::eventedit::m_label_right [private]

10.12.4.31 m_seq

```
sequence& seq64::eventedit::m_seq [private]
```

10.12.4.32 m_have_focus

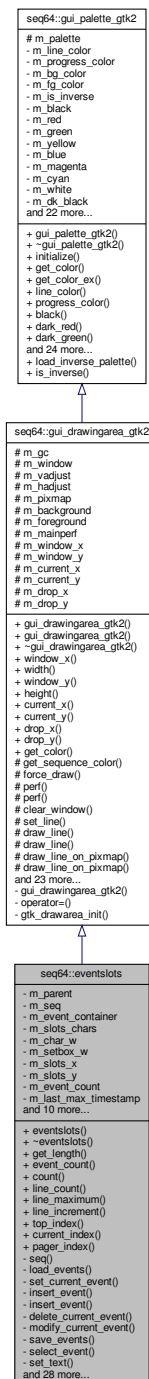
```
bool seq64::eventedit::m_have_focus [private]
```

This item is to modify the mainwid and perfedit "edit-sequence" value in order to highlight pattern slot of the pattern/event editor that currently has the user-input focus.

10.13 seq64::eventslots Class Reference

This class implements the left-side list of events in the pattern event-edit window.

Inheritance diagram for seq64::eventslots:



Public Member Functions

- [eventslots](#) (perform &p, [eventedit](#) &parent, [sequence](#) &seq, Gtk::Adjustment &vadjust)

Principal constructor for this user-interface object.

- virtual [~eventslots](#) ()

Let's provide a do-nothing virtual destructor.

- [midipulse get_length](#) () const

- *'Getter' function for member m_event_container.get_length()*
- int `event_count` () const
'Getter' function for member m_event_count Returns the number of total events in the sequence represented by the eventslots object.
- int `count` () const
'Getter' function for member m_event_count
- int `line_count` () const
'Getter' function for member m_line_count Returns the current number of rows (events) in the eventslots's display.
- int `line_maximum` () const
'Getter' function for member m_line_maximum Returns the maximum number of rows (events) in the eventslots's display.
- int `line_increment` () const
Provides the "page increment" or "line increment" of the frame, This value is the current line-maximum of the frame minus its overlap value.
- int `top_index` () const
'Getter' function for member m_top_index
- int `current_index` () const
'Getter' function for member m_current_index
- int `pager_index` () const
'Getter' function for member m_pager_index

Private Member Functions

- const `sequence & seq` () const
'Getter' function for member m_seq const version
- bool `load_events` ()
Grabs the event list from the sequence and uses it to fill the editable-event list.
- void `set_current_event` (const `editable_events::iterator` ei, int index, bool full_redraw=true)
Set the current event, which is the event that is highlighted.
- bool `insert_event` (const `editable_event` &edev)
Inserts an event.
- bool `insert_event` (const std::string &evtimestamp, const std::string &evname, const std::string &evdata0, const std::string &evdata1)
Inserts an event based on the setting provided, which the eventedit object gets from its Entry fields.
- bool `delete_current_event` ()
Deletes the current event, and makes adjustments due to that deletion.
- bool `modify_current_event` (const std::string &evtimestamp, const std::string &evname, const std::string &evdata0, const std::string &evdata1)
Modifies the data in the currently-selected event.
- bool `save_events` ()
Writes the events back to the sequence.
- void `select_event` (int event_index=SEQ64_NULL_EVENT_INDEX, bool full_redraw=true)
Selects and highlights the event that is located in the frame at the given event index.
- void `set_text` (const std::string &evcategory, const std::string &evtimestamp, const std::string &evname, const std::string &evdata0, const std::string &evdata1)
Sets the text in the parent dialog, eventedit.
- void `enqueue_draw` ()
Wraps queue_draw().
- int `convert_y` (int y)
Converts a y-value into an event index relative to 0 (the top of the eventslots window/pixmap) and returns it.
- void `draw_event` (`editable_events::iterator` ei, int index)

- Draw the given slot/event.*

 - void `draw_events` ()

Draws all of the events in the current eventslots frame.
- void `change_vert` ()

Change the vertical offset of events.
- void `page_movement` (int new_value)

Adjusts the vertical position of the frame according to the given new scrollbar/vadjust value.
- void `page_topper` (editable_events::iterator newcurrent)

Adjusts the vertical position of the frame according to the given new bottom iterator.
- int `decrement_top` ()

Decrements the top iterator, if possible.
- int `increment_top` ()

Increments the top iterator, if possible.
- int `decrement_current` ()

Decrements the current iterator, if possible.
- int `increment_current` ()

Increments the current iterator, if possible.
- int `decrement_bottom` ()

Decrements the bottom iterator, if possible.
- int `increment_bottom` ()

Increments the bottom iterator, if possible.
- int `calculate_measures` () const

This function basically duplicates `sequence::calculate_measures()`.
- void `on_realize` ()

Handles the callback when the window is realized.
- bool `on_expose_event` (GdkEventExpose *ev)

Handles an on-expose event.
- bool `on_button_press_event` (GdkEventButton *ev)

Provides the callback for a button press, and it handles only a left mouse button.
- bool `on_button_release_event` (GdkEventButton *ev)

Currently does nothing.
- bool `on_focus_in_event` (GdkEventFocus *ev)

This callback is an attempt to get keyboard focus into the eventslots pixmap area.
- bool `on_focus_out_event` (GdkEventFocus *ev)

This callback handles an out-of-focus event by resetting the flag `HAS_FOCUS`.
- bool `on_scroll_event` (GdkEventScroll *ev)

Handle the scrolling of the window.
- void `on_size_allocate` (Gtk::Allocation &)

Handles a size-allocation event.
- void `on_move_up` ()

Move to the previous event.
- void `on_move_down` ()

Move to the next event.
- void `on_frame_up` ()

Move to the previous frame.
- void `on_frame_down` ()

Move to the next frame.
- void `on_frame_home` ()

Move to the first frame.
- void `on_frame_end` ()

Move to the last frame.

Private Attributes

- [eventedit](#) & [m_parent](#)
Provides a link to the eventedit that created this object.
- [sequence](#) & [m_seq](#)
Provides a reference to the sequence that this dialog is meant to view or modify.
- [editable_events](#) [m_event_container](#)
Holds the editable events for this sequence.
- [int](#) [m_slots_chars](#)
Provides the number of the characters in the name box.
- [int](#) [m_char_w](#)
Provides the "real" width of a character.
- [int](#) [m_setbox_w](#)
Provides the width of the "set number" box.
- [int](#) [m_slots_x](#)
Provides the width of the names box, which is the width of a character for 24 characters.
- [int](#) [m_slots_y](#)
Provides the height of the names box, which is hardwired to 24 pixels.
- [int](#) [m_event_count](#)
The current number of events in the edited container.
- [midipulse](#) [m_last_max_timestamp](#)
Holds the previous length of the edited sequence, in MIDI pulses, so that we can detect changes in the length of the sequence.
- [int](#) [m_measures](#)
Holds the current number of measures, for display purposes.
- [int](#) [m_line_count](#)
Counts the number of displayed events, which depends on how many events there are ([m_event_count](#)) and the size of the event list ([m_line_maximum](#)).
- [int](#) [m_line_maximum](#)
Counts the maximum number of displayed events, which depends on the size of the event list (and thus the size of the dialog box for the event editor).
- [int](#) [m_line_overlap](#)
Provides a little overlap for paging through the frame.
- [int](#) [m_top_index](#)
The index of the event that is 0th in the visible list of events.
- [int](#) [m_current_index](#)
Indicates the index of the current event within the frame.
- [editable_events::iterator](#) [m_top_iterator](#)
Provides the top "pointer" to the start of the editable-events section that is being shown in the user-interface.
- [editable_events::iterator](#) [m_bottom_iterator](#)
Provides the bottom "pointer" to the end of the editable-events section that is being shown in the user-interface.
- [editable_events::iterator](#) [m_current_iterator](#)
Provides the "pointer" to the event currently in focus.
- [int](#) [m_pager_index](#)
Indicates the event index that matches the index value of the vertical pager.

Friends

- class [eventedit](#)

Additional Inherited Members

10.13.1 Constructor & Destructor Documentation

10.13.1.1 eventslots()

```
seq64::eventslots::eventslots (
    perform & p,
    eventedit & parent,
    sequence & seq,
    Gtk::Adjustment & vadjust )
```

Parameters

<i>p</i>	The parent perform object.
<i>parent</i>	The parent event-editor object.
<i>seq</i>	Provides the sequence represented by this event-editing object.
<i>vadjust</i>	Provides the vertical-adjustment object, used in updating the window that represents the events being edited.

10.13.1.2 ~eventslots()

```
virtual seq64::eventslots::~~eventslots ( ) [inline], [virtual]
```

10.13.2 Member Function Documentation

10.13.2.1 get_length()

```
midipulse seq64::eventslots::get_length ( ) const [inline]
```

10.13.2.2 event_count()

```
int seq64::eventslots::event_count ( ) const [inline]
```

10.13.2.3 count()

```
int seq64::eventslots::count ( ) const [inline]
```

10.13.2.4 line_count()

```
int seq64::eventslots::line_count ( ) const [inline]
```

10.13.2.5 line_maximum()

```
int seq64::eventslots::line_maximum ( ) const [inline]
```

10.13.2.6 line_increment()

```
int seq64::eventslots::line_increment ( ) const [inline]
```

10.13.2.7 top_index()

```
int seq64::eventslots::top_index ( ) const [inline]
```

10.13.2.8 current_index()

```
int seq64::eventslots::current_index ( ) const [inline]
```

10.13.2.9 pager_index()

```
int seq64::eventslots::pager_index ( ) const [inline]
```

10.13.2.10 seq()

```
const sequence& seq64::eventslots::seq ( ) const [inline], [private]
```

10.13.2.11 load_events()

```
bool seq64::eventslots::load_events ( ) [private]
```

Determines how many events can be shown in the GUI [later] and adjusts the top and bottom editable-event iterators to show the first page of events.

Returns

Returns true if the event iterators were able to be set up as valid.

10.13.2.12 set_current_event()

```
void seq64::eventslots::set_current_event (
    const editable_events::iterator ei,
    int index,
    bool full_redraw = true ) [private]
```

Note in the `snprintf()` calls that the first digit is part of the data byte, so that translation is easier.

Parameters

<i>ei</i>	The iterator that points to the event.
<i>index</i>	The index (re 0) of the event, starting at the top line of the frame. It is a frame index, not a container index.
<i>full_redraw</i>	If true (the default) does a full redraw of the frame. Otherwise, only the current event is drawn. Generally, the only time a single event (actually, two adjacent events) is convenient to draw is when using the arrow keys, where the speed of keystroke auto-repeats makes the full-frame update scrolling very flickery and disconcerting.

10.13.2.13 insert_event() [1/2]

```
bool seq64::eventslots::insert_event (
    const editable_event & edev ) [private]
```

What actually happens here depends if the new event is before the frame, within the frame, or after the frame, based on the timestamp. Also, we want to allow the lengthening of a sequence by inserting an event past its current length. Especially useful for the tempo track.

If before the frame: To keep the previous events visible, we do not need to increment the iterators (insertion does not affect multimap iterators), but we do need to increment their indices. The contents shown in the frame should not change.

If at the frame top: The new timestamp equals the top timestamp. We don't know exactly where the new event goes in the multimap, but we do have a new event.

If at the frame bottom: TODO

If after the frame: No action needed if the bottom event is actually at the bottom of the frame. But if the frame is not yet filled, we need to increment the bottom iterator, and its index.

Note

Actually, it is far easier to just adjust all the counts and iterators and redraw the screen, as done by the [page_topper\(\)](#) function.

Parameters

<i>edev</i>	The event to insert, prebuilt.
-------------	--------------------------------

Returns

Returns true if the event was inserted.

10.13.2.14 insert_event() [2/2]

```
bool seq64::eventslots::insert_event (
    const std::string & evtimestamp,
    const std::string & evname,
    const std::string & evdata0,
    const std::string & evdata1 ) [private]
```

It calls the other [insert_event\(\)](#) overload.

Note that we need to qualify the temporary event class object we create below, with the [seq64](#) namespace, otherwise the compiler thinks we're trying to access some Gtkmm thing.

Parameters

<i>evtimestamp</i>	The time-stamp of the new event, as obtained from the event-edit timestamp field.
<i>evname</i>	The type name (status name) of the new event, as obtained from the event-edit event-name field.
<i>evdata0</i>	The first data byte of the new event, as obtained from the event-edit data 1 field.
<i>evdata1</i>	The second data byte of the new event, as obtained from the event-edit data 2 field. Used only for two-parameter events.

Returns

Returns true if the event was inserted.

10.13.2.15 delete_current_event()

```
bool seq64::eventslots::delete_current_event ( ) [private]
```

To delete the current event, this function moves the current iterator to the next event, deletes the previously-current iterator, adjusts the event count and the bottom iterator, and redraws the pixmap. The exact changes depend upon whether the deleted event was at the top of the visible frame, within the visible frame, or at the bottom the visible frame. Note that only visible events can be the current event, and thus get deleted.

```

Event Index
0
1
2           Top
3 <----- Top case: The new top iterator, index becomes 2
4
.
.           Inside of Visible Frame
.
43
44          Bottom
45 <----- Top case: The new bottom iterator, index becomes 44
46          Bottom case: Same result

```

Basically, when an event is deleted, the frame (delimited by the event-index members) stays in place, while the frame iterators move to the previous event. If the top of the frame would move to before the first event, then the frame must shrink.

Top case: If the current iterator is the top (of the frame) iterator, then the top iterator needs to be incremented. The new top event has the same index as the now-gone top event. The index of the bottom event is decremented, since an event before it is now gone. The bottom iterator moves to the next event, which is now at the bottom of the frame. The current event is treated like the top event.

Inside case: If the current iterator is in the middle of the frame, the top iterator and index remain unchanged. The current iterator is incremented, but its index is now the same as the old bottom index. Same for the bottom iterator.

Bottom case: If the current iterator (and bottom iterator) point to the last event in the frame, then both of them need to be decremented. The frame needs to be moved up by one event, so that the current event remains at the bottom (it's just simpler to manage that way).

If there is no event after the bottom of the frame, the iterators that now point to end() must backtrack one event. If the container becomes empty, then everything is invalidated.

Returns

Returns true if the delete was possible. If the container was empty or became empty, then false is returned.

10.13.2.16 modify_current_event()

```

bool seq64::eventslots::modify_current_event (
    const std::string & evtimestamp,
    const std::string & evname,
    const std::string & evdata0,
    const std::string & evdata1 ) [private]

```

If the timestamp has changed, however, we can't just modify the event in place. Instead, we finish modifying the event, but tell the caller to delete and reinsert the new event (in its proper new location based on timestamp).

This function always copies the original event, modifies the copy, deletes the original event, and inserts the "new" event into the editable-event container. The insertion takes care of updating any length increase of the sequence.

Parameters

<i>evtimestamp</i>	Provides the new event time-stamp as edited by the user.
<i>evname</i>	Provides the event name as edited by the user.
<i>evdata0</i>	Provides the first data byte as edited by the user.
<i>evdata1</i>	Provides the second data byte as edited by the user.

Returns

Returns true simply if the event-count is greater than 0.

10.13.2.17 save_events()

```
bool seq64::eventslots::save_events ( ) [private]
```

Also sets the dirty flag for the sequence, via the [sequence::add_event\(\)](#) function, but this doesn't seem to set the perform dirty flag. So now we pass the modification buck to the parent, who passes it to the perform object.

We added a `copy_events()` function in the sequence class to replace `add_event()` for the purpose of reconstructing the events container for the sequence. It is locked by a mutex, and so will not draw until all is done, preventing a nasty segfault (all segfaults are nasty).

We create a new plain event container here, and then passing it to the new locked/threadsafe [sequence::copy_events\(\)](#) function that clears the sequence container and copies the events from the parameter container.

Note that this code will operate event if all events were deleted.

Returns

Returns true if the operations succeeded.

10.13.2.18 select_event()

```
void seq64::eventslots::select_event (
    int event_index = SEQ64_NULL_EVENT_INDEX,
    bool full_redraw = true ) [private]
```

The event index is provided by converting the y-coordinate of the mouse pointer into a slot number, and then an event index (actually the slot-distance from the `m_top_iterator`. Confusing, yes no?

Note that, if the event index is negative, then we just queue up a draw operation, which should paint an empty frame – the event container is empty.

Parameters

<i>event_index</i>	Provides the numeric index of the event in the event frame, or <code>SEQ64_NULL_EVENT</code> if there is no event to draw.
<i>full_redraw</i>	Defaulting to true, this parameter can be set to false in some case to reduce the flickering of the event frame under fast movement.

10.13.2.19 set_text()

```
void seq64::eventslots::set_text (
    const std::string & evcategory,
    const std::string & evtimestamp,
    const std::string & evname,
    const std::string & evdata0,
    const std::string & evdata1 ) [private]
```

Parameters

<i>evcategory</i>	The category of event to be set in the parent.
<i>evtimestamp</i>	The event time-stamp to be set in the parent.
<i>evname</i>	The event name to be set in the parent.
<i>evdata0</i>	The first event data byte to be set in the parent.
<i>evdata1</i>	The second event data byte to be set in the parent.

10.13.2.20 enqueue_draw()

```
void seq64::eventslots::enqueue_draw ( ) [private]
```

10.13.2.21 convert_y()

```
int seq64::eventslots::convert_y (
    int y ) [private]
```

Parameters

<i>y</i>	The y coordinate of the position of the mouse click in the eventslot window/pixmap.
----------	---

Returns

Returns the index of the event position in the user-interface, which should range from 0 to `m_line_count`.

10.13.2.22 draw_event()

```
void seq64::eventslots::draw_event (
    editable_events::iterator ei,
    int index ) [private]
```


The slot contains the event details in (so far) one line of text in the box:

```
| timestamp | event kind | channel | data 0 name + value | data 1 name + value
```

Currently, this view shows only events that get copied to the sequence's event list. This rules out the following items from the view:

- MThd (song header)
- MTrk and Meta TrkEnd (track marker, a sequence has only one track)
- SeqNr (sequence number)
- SeqSpec (but there are three that might appear, see below)
- Meta TrkName

The events that are shown in this view are:

- One-data-value events:
 - Program Change
 - Channel Pressure
- Two-data-value events:
 - Note Off
 - Note On
 - Aftertouch
 - Control Change
 - Pitch Wheel
- Other:
 - SysEx events, with partial show of data bytes
 - SeqSpec events (TBD):
 - Key
 - Scale
 - Background sequence

The index of the event is shown in the editor portion of the eventedit dialog.

10.13.2.23 draw_events()

```
void seq64::eventslots::draw_events ( ) [private]
```

It first clears the whole bitmap to white, so that no artifacts from the previous state of the frame are left behind.

Need to figure out how to calculate the number of displayable events.

```
m_line_maximum = ???
```

10.13.2.24 change_vert()

```
void seq64::eventslots::change_vert ( ) [private]
```

Note that `m_vadjust` is the `Gtk::Adjustment` object that the `eventedit` parent passes to the [gui_drawingarea_gtk2](#) constructor.

The top-event and bottom-event indices (and their corresponding editable-event iterators) delimit the part of the event container that is displayed in the eventslots user-interface. The top-event index starts at 0, and the bottom-event is larger (initially, by 42 slots).

When the scroll-bar thumb moves up or down, we need to change both event indices and both event iterators by the corresponding amount. Luckily, the `std::multimap` iterator is bidirectional.

Note that we may need to reduce the movement of events to a value less than a page; it can be limited backwards by the value of the top index, and forward by the value of the bottom index.

10.13.2.25 page_movement()

```
void seq64::eventslots::page_movement (
    int new_value ) [private]
```

The adjustment is done via movement from the current position.

Do we even need a way to detect excess movement? The scrollbar, if properly set up, should never move the frame too high or too low. Verified by testing.

Parameters

<i>new_value</i>	Provides the new value of the scrollbar position.
------------------	---

10.13.2.26 page_topper()

```
void seq64::eventslots::page_topper (
    editable_events::iterator newcurrent ) [private]
```

The adjustment is done "from scratch". We've found page movement to be an insoluble problem in some editing circumstances. So now we move to the inserted event, and make it the top event.

However, always moving an inserted event to the top is a bit annoying. So now we backtrack so that the inserted event is at the bottom.

Parameters

<i>newcurrent</i>	Provides the iterator to the event to be shown at the bottom of the frame.
-------------------	--

10.13.2.27 decrement_top()

```
int seq64::eventslots::decrement_top ( ) [private]
```

Returns

Returns 0, or SEQ64_NULL_EVENT_INDEX if the iterator could not be decremented.

10.13.2.28 increment_top()

```
int seq64::eventslots::increment_top ( ) [private]
```

Also handles the top-event index, so that the GUI can display the proper event numbers.

Returns

Returns the top index, or SEQ64_NULL_EVENT_INDEX if the iterator could not be incremented, or would increment to the end of the container.

10.13.2.29 decrement_current()

```
int seq64::eventslots::decrement_current ( ) [private]
```

Returns

Returns the decremented index, or SEQ64_NULL_EVENT_INDEX if the iterator could not be decremented. Remember that the index ranges only from 0 to m_line_count-1, and that is enforced here.

10.13.2.30 increment_current()

```
int seq64::eventslots::increment_current ( ) [private]
```

Returns

Returns the incremented index, or SEQ64_NULL_EVENT_INDEX if the iterator could not be incremented. Remember that the index ranges only from 0 to m_line_count-1, and that is enforced here.

10.13.2.31 decrement_bottom()

```
int seq64::eventslots::decrement_bottom ( ) [private]
```

Returns

Returns 0, or SEQ64_NULL_EVENT_INDEX if the iterator could not be decremented.

10.13.2.32 increment_bottom()

```
int seq64::eventslots::increment_bottom ( ) [private]
```

There is an issue in paging down using the scrollbar where, at the bottom of the scrolling, the bottom iterator ends up bad. Not yet sure how this happens, so for now we backtrack one event if this happens.

Returns

Returns the incremented index, or SEQ64_NULL_EVENT_INDEX if the iterator could not be incremented.

10.13.2.33 calculate_measures()

```
int seq64::eventslots::calculate_measures ( ) const [private]
```

Returns

Returns the number of measures needed to cover the full length of the current events in the container. The lowest valid measure is 1.

10.13.2.34 on_realize()

```
void seq64::eventslots::on_realize ( ) [private]
```

It first calls the base-class version of [on_realize\(\)](#). Then it allocates any additional resources needed.

10.13.2.35 on_expose_event()

```
bool seq64::eventslots::on_expose_event (
    GdkEventExpose * ev ) [private]
```

It draws all of the sequences.

10.13.2.36 on_button_press_event()

```
bool seq64::eventslots::on_button_press_event (
    GdkEventButton * ev ) [private]
```

10.13.2.37 on_button_release_event()

```
bool seq64::eventslots::on_button_release_event (
    GdkEventButton * ev ) [private]
```

10.13.2.38 on_focus_in_event()

```
bool seq64::eventslots::on_focus_in_event (
    GdkEventFocus * ev ) [private]
```

See the same function in the `perfroll` module.

10.13.2.39 on_focus_out_event()

```
bool seq64::eventslots::on_focus_out_event (
    GdkEventFocus * ev ) [private]
```

10.13.2.40 on_scroll_event()

```
bool seq64::eventslots::on_scroll_event (
    GdkEventScroll * ev ) [private]
```

10.13.2.41 on_size_allocate()

```
void seq64::eventslots::on_size_allocate (
    Gtk::Allocation & a ) [private]
```

It first calls the base-class version of this function.

10.13.2.42 on_move_up()

```
void seq64::eventslots::on_move_up ( ) [private]
```

We must scroll up if the event is now before the frame, and should be made the new top event of the frame. Note that this function isn't really an event-response callback. It is called by [eventedit::on_key_press_event\(\)](#).

10.13.2.43 on_move_down()

```
void seq64::eventslots::on_move_down ( ) [private]
```

We must scroll down if the event is now after the frame. Note that this function isn't really an event-response callback. It is called by [eventedit::on_key_press_event\(\)](#).

10.13.2.44 on_frame_up()

```
void seq64::eventslots::on_frame_up ( ) [private]
```

10.13.2.45 on_frame_down()

```
void seq64::eventslots::on_frame_down ( ) [private]
```

10.13.2.46 on_frame_home()

```
void seq64::eventslots::on_frame_home ( ) [private]
```

10.13.2.47 on_frame_end()

```
void seq64::eventslots::on_frame_end ( ) [private]
```

10.13.3 Friends And Related Function Documentation

10.13.3.1 eventedit

```
friend class eventedit [friend]
```

10.13.4 Field Documentation

10.13.4.1 m_parent

```
eventedit& seq64::eventslots::m_parent [private]
```

10.13.4.2 m_seq

```
sequence& seq64::eventslots::m_seq [private]
```

10.13.4.3 m_event_container

```
editable_events seq64::eventslots::m_event_container [private]
```

This container is what is edited, and any changes made to it are not saved to the container until the user pushes the "save" button.

10.13.4.4 m_slots_chars

```
int seq64::eventslots::m_slots_chars [private]
```

Pretty much hardwired to 64 at present. It helps determine the m_slots_x value (the width of the eventslots list).

10.13.4.5 m_char_w

```
int seq64::eventslots::m_char_w [private]
```

This value is obtained from a font-renderer accessor function.

10.13.4.6 m_setbox_w

```
int seq64::eventslots::m_setbox_w [private]
```

This used to be hardwired to $6 * 2$ (character-width times two).

10.13.4.7 m_slots_x

```
int seq64::eventslots::m_slots_x [private]
```

10.13.4.8 m_slots_y

```
int seq64::eventslots::m_slots_y [private]
```

This value was once 22 pixels, but we need a little extra room for our new font. This extra room is compatible enough with the old font, as well.

10.13.4.9 m_event_count

```
int seq64::eventslots::m_event_count [private]
```

10.13.4.10 m_last_max_timestamp

```
midipulse seq64::eventslots::m_last_max_timestamp [private]
```

10.13.4.11 m_measures

```
int seq64::eventslots::m_measures [private]
```

10.13.4.12 m_line_count

```
int seq64::eventslots::m_line_count [private]
```

10.13.4.13 m_line_maximum

```
int seq64::eventslots::m_line_maximum [private]
```

10.13.4.14 m_line_overlap

```
int seq64::eventslots::m_line_overlap [private]
```

10.13.4.15 m_top_index

```
int seq64::eventslots::m_top_index [private]
```

It is used in numbering the events that are shown in the event-slot frame. Do not confuse it with `m_current_index`, which is relative to the frame, not the container-beginning.

10.13.4.16 m_current_index

```
int seq64::eventslots::m_current_index [private]
```

This event will also be pointed to by the `m_current_event` iterator. Do not confuse it with `m_top_index`, which is relative to the container-beginning, not the frame.

10.13.4.17 m_top_iterator

```
editable_events::iterator seq64::eventslots::m_top_iterator [private]
```


10.13.4.18 m_bottom_iterator

```
editable_events::iterator seq64::eventslots::m_bottom_iterator [private]
```

10.13.4.19 m_current_iterator

```
editable_events::iterator seq64::eventslots::m_current_iterator [private]
```

10.13.4.20 m_pager_index

```
int seq64::eventslots::m_pager_index [private]
```

10.14 seq64::font Class Reference

This class provides a wrapper for rendering fonts that are encoded as a 16 x 16 pixmap file in XPM format.

Public Types

- enum [Color](#) {
[BLACK](#),
[WHITE](#),
[BLACK_ON_YELLOW](#),
[YELLOW_ON_BLACK](#),
[BLACK_ON_CYAN](#),
[CYAN_ON_BLACK](#) }

A simple enumeration to describe the basic colors used in writing text.

Public Member Functions

- [font](#) ()
Rote default constructor, except that it does add 1 to the cf_text_h or co_text_h values to use in m_padded_h.
- void [init](#) (Glib::RefPtr< Gdk::Window > windo)
Initialization function for a window on which fonts will be drawn.
- void [render_string_on_drawable](#) (Glib::RefPtr< Gdk::GC > m_gc, int x, int y, Glib::RefPtr< Gdk::Drawable > drawable, const char *str, [font::Color](#) col, bool invert=false) const
Draws a text string.
- int [char_width](#) () const
'Getter' function for member m_font_w
- int [char_height](#) () const
'Getter' function for member m_font_h
- int [padded_height](#) () const
'Getter' function for member m_padded_h

Private Attributes

- bool [m_use_new_font](#)
If true, use the new font, which is a little bit more modern looking, and is also thicker, and thus a little easier to see.
- int [m_cell_w](#)
Specifies the cell width of the whole character cell.
- int [m_cell_h](#)
Specifies the cell height of the whole character cell.
- int [m_font_w](#)
Specifies the exact width of a character cell, in pixels.
- int [m_font_h](#)
Specifies the exact height of a character cell, in pixels.
- int [m_offset](#)
Provides an ad hoc small horizontal or vertical offset for printing strings.
- int [m_padded_h](#)
Provides a common constant used by much of the drawing code, but only marginally related to the padded character height.
- const Glib::RefPtr< Gdk::Pixmap > * [m_pixmap](#)
Points to the current pixmap ([m_black_pixmap](#) or [m_white_pixmap](#)) to use to render a string.
- Glib::RefPtr< Gdk::Pixmap > [m_black_pixmap](#)
The pixmap in the file `src/pixmaps/font_b.xpm` is loaded into this object.
- Glib::RefPtr< Gdk::Pixmap > [m_white_pixmap](#)
The pixmap in the file `src/pixmaps/font_w.xpm` is loaded into this object.
- Glib::RefPtr< Gdk::Pixmap > [m_b_on_y_pixmap](#)
The pixmap in the file `src/pixmaps/font_y.xpm` is loaded into this object.
- Glib::RefPtr< Gdk::Pixmap > [m_y_on_b_pixmap](#)
The pixmap in the file `src/pixmaps/font_yb.xpm` is loaded into this object.
- Glib::RefPtr< Gdk::Pixmap > [m_b_on_c_pixmap](#)
The pixmap in the file `src/pixmaps/cyan_wenfont_y.xpm` is loaded into this object.
- Glib::RefPtr< Gdk::Pixmap > [m_c_on_b_pixmap](#)
The pixmap in the file `src/pixmaps/cyan_wenfont_yb.xpm` is loaded into this object.
- Glib::RefPtr< Gdk::Bitmap > [m_clip_mask](#)
This object is instantiated as a default object.

10.14.1 Member Enumeration Documentation

10.14.1.1 Color

enum [seq64::font::Color](#)

Basically, these two values cause the selection of one or another pixmap ([font_b_xpm](#) and [font_w_xpm](#)). We've added two more pixmaps to draw black text on a yellow background ([font_y.xpm](#)) and yellow text on a black background ([font_yb.xpm](#)). Oh, and couple more for cyan and black text-blitting.

Enumerator

BLACK	The first supported color. A black font on a white background.
WHITE	The second supported color. A white font on a black background.
BLACK_ON_YELLOW	A new color, for drawing black text on a yellow background.
YELLOW_ON_BLACK	A new color, for drawing yellow text on a black background.
BLACK_ON_CYAN	A new color, for drawing black text on a cyan background.
CYAN_ON_BLACK	A new color, for drawing cyan text on a black background.

10.14.2 Constructor & Destructor Documentation

10.14.2.1 font()

```
seq64::font::font ( )
```

10.14.3 Member Function Documentation

10.14.3.1 init()

```
void seq64::font::init (
    Glib::RefPtr< Gdk::Window > wp )
```

This function loads four pixmaps that contain the characters to be used to draw text strings.

One pixmap has white characters on a black background, one has black characters on a white background, one has yellow characters on a black background, and one has black characters on a yellow background.

Todo Can we scale these images via `scale_simple(newwidth, newheight, Gtk::INTERP_BILINEAR)`

Parameters

<code>wp</code>	Provides the windows pointer for the window that holds the color map.
-----------------	---

10.14.3.2 render_string_on_drawable()

```
void seq64::font::render_string_on_drawable (
    Glib::RefPtr< Gdk::GC > gc,
    int x,
    int y,
    Glib::RefPtr< Gdk::Drawable > adraw,
    const char * str,
    font::Color col,
    bool invert = false ) const
```

This function grabs the proper font bitmap, extracts the current character pixmap from it, and slaps it down where it needs to be to render the character in the string.

Parameters

<code>gc</code>	Provides the graphics context for drawing the text using GTK+.
-----------------	--

Parameters

<i>x</i>	The horizontal location of the text.
<i>y</i>	The vertical location of the text.
<i>adraw</i>	The drawable object on which to draw the text.
<i>str</i>	The string to draw. Should use a constant string reference instead.
<i>col</i>	The font color to use to draw the string. The supported values are font::BLACK , font::WHITE , font::BLACK_ON_YELLOW , font::YELLOW_ON_BLACK . The actual correct colors are provided by selecting one of four font pixmaps, as described in the init() function.
<i>invert</i>	If true, apply color inversion, if specified.

10.14.3.3 `char_width()`

```
int seq64::font::char_width ( ) const [inline]
```

10.14.3.4 `char_height()`

```
int seq64::font::char_height ( ) const [inline]
```

10.14.3.5 `padded_height()`

```
int seq64::font::padded_height ( ) const [inline]
```

10.14.4 Field Documentation

10.14.4.1 `m_use_new_font`

```
bool seq64::font::m_use_new_font [private]
```

10.14.4.2 `m_cell_w`

```
int seq64::font::m_cell_w [private]
```

10.14.4.3 m_cell_h

```
int seq64::font::m_cell_h [private]
```

10.14.4.4 m_font_w

```
int seq64::font::m_font_w [private]
```

Currently defaults to `cf_text_w = 6`. Note that a lot of stuff depends on this being 6 at present, even with our new, slightly wider, font.

10.14.4.5 m_font_h

```
int seq64::font::m_font_h [private]
```

Currently defaults to `cf_text_h = 10`. Note that a lot of stuff depends on this being 10 at present, even with our new, slightly wider, font. But some of the drawing code doesn't use the character height, but the padded character height.

10.14.4.6 m_offset

```
int seq64::font::m_offset [private]
```

10.14.4.7 m_padded_h

```
int seq64::font::m_padded_h [private]
```

10.14.4.8 m_pixmap

```
const Glib::RefPtr<Gdk::Pixmap>* seq64::font::m_pixmap [mutable], [private]
```

This member used to be an object, but it's probably a bit faster to just use a pointer (or a reference).

10.14.4.9 m_black_pixmap

```
Glib::RefPtr<Gdk::Pixmap> seq64::font::m_black_pixmap [private]
```

It contains a black font on a white background. The new-style font, if selected, is in the `resources/pixmaps/wenfont↔_b.xmp` pixmap.

10.14.4.10 m_white_pixmap

```
Glib::RefPtr<Gdk::Pixmap> seq64::font::m_white_pixmap [private]
```

It contains a black font on a white background. The new-style font, if selected, is in the `resources/pixmaps/wenfont↵_w.xmp` pixmap.

10.14.4.11 m_b_on_y_pixmap

```
Glib::RefPtr<Gdk::Pixmap> seq64::font::m_b_on_y_pixmap [private]
```

It contains a black font on a yellow background. The new-style font, if selected, is in the `resources/pixmaps/wenfont↵_y.xmp` pixmap.

10.14.4.12 m_y_on_b_pixmap

```
Glib::RefPtr<Gdk::Pixmap> seq64::font::m_y_on_b_pixmap [private]
```

It contains a yellow font on a black background. The new-style font, if selected, is `resources/pixmaps/wenfont↵_yb.xmp` pixmap.

10.14.4.13 m_b_on_c_pixmap

```
Glib::RefPtr<Gdk::Pixmap> seq64::font::m_b_on_c_pixmap [private]
```

It contains a black font on a cyan background. It is available only for the new font-style.

10.14.4.14 m_c_on_b_pixmap

```
Glib::RefPtr<Gdk::Pixmap> seq64::font::m_c_on_b_pixmap [private]
```

It contains a cyan font on a black background. It is available only for the new font-style.

10.14.4.15 m_clip_mask

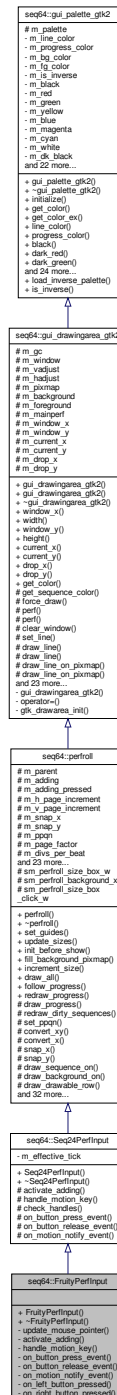
```
Glib::RefPtr<Gdk::Bitmap> seq64::font::m_clip_mask [private]
```

All we know is it seems to be a requirement for creating a pixmap object from an XMP file.

10.15 seq64::FruityPerfInput Class Reference

Implements the performance input of that certain fruity sequencer that people seem to like.

Inheritance diagram for seq64::FruityPerfInput:



Public Member Functions

- [FruityPerfInput](#) ([perform](#) &[perf](#), [perfedit](#) &parent, [Gtk::Adjustment](#) &hadjust, [Gtk::Adjustment](#) &vadjust, int ppqn=SEQ64_USE_DEFAULT_PPQN)

Principal constructor.

- `~FruityPerfInput ()`

virtual destructor

Private Member Functions

- void `update_mouse_pointer ()`
Updates the mouse pointer, implementing a context-sensitive mouse.
- virtual void `activate_adding (bool)`
- virtual bool `handle_motion_key (bool)`
- virtual bool `on_button_press_event (GdkEventButton *ev)`
Handles a button-press event in the Fruity manner.
- virtual bool `on_button_release_event (GdkEventButton *ev)`
Handles a button-release event.
- virtual bool `on_motion_notify_event (GdkEventMotion *ev)`
Handles a Fruity motion-notify event.
- virtual bool `on_left_button_pressed (GdkEventButton *ev)`
Handles the left button of the mouse.
- virtual bool `on_right_button_pressed (GdkEventButton *ev)`
Handles the right button of the mouse.

Additional Inherited Members

10.15.1 Constructor & Destructor Documentation

10.15.1.1 FruityPerfInput()

```
seq64::FruityPerfInput::FruityPerfInput (
    perform & p,
    perfedit & parent,
    Gtk::Adjustment & hadjust,
    Gtk::Adjustment & vadjust,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

Parameters

<i>p</i>	The perform object that this class will affect via user interaction.
<i>parent</i>	The perfedit object that holds this user-interface class.
<i>hadjust</i>	A horizontal adjustment object, passed along to the perffroll class.
<i>vadjust</i>	A vertical adjustment object, passed along to the perffroll class.
<i>ppqn</i>	The "resolution" of the MIDI file, used in zooming and scaling.

10.15.1.2 ~FruityPerfInput()

```
seq64::FruityPerfInput::~~FruityPerfInput ( ) [inline]
```

10.15.2 Member Function Documentation

10.15.2.1 update_mouse_pointer()

```
void seq64::FruityPerfInput::update_mouse_pointer ( ) [private]
```

Note that `perform::convert_xy()` returns its values via side-effects on the last two parameters.

10.15.2.2 activate_adding()

```
virtual void seq64::FruityPerfInput::activate_adding (
    bool ) [inline], [private], [virtual]
```

Reimplemented from [seq64::Seq24PerfInput](#).

10.15.2.3 handle_motion_key()

```
virtual bool seq64::FruityPerfInput::handle_motion_key (
    bool ) [inline], [private], [virtual]
```

Reimplemented from [seq64::Seq24PerfInput](#).

10.15.2.4 on_button_press_event()

```
bool seq64::FruityPerfInput::on_button_press_event (
    GdkEventButton * ev ) [private], [virtual]
```

Parameters

<i>ev</i>	The button-press event to process.
-----------	------------------------------------

Returns

Returns true if a modification occurred.

Reimplemented from [seq64::Seq24PerfInput](#).

10.15.2.5 on_button_release_event()

```
bool seq64::FruityPerfInput::on_button_release_event (
    GdkEventButton * ev ) [private], [virtual]
```

Why is m_adding_pressed modified conditionally when the same modification is then made unconditionally?

Parameters

ev	The button-release event to process.
----	--------------------------------------

Returns

Returns true if a modification occurred.

Reimplemented from [seq64::Seq24PerfInput](#).

10.15.2.6 on_motion_notify_event()

```
bool seq64::FruityPerfInput::on_motion_notify_event (
    GdkEventMotion * ev ) [private], [virtual]
```

Parameters

ev	The motion-notify event to process.
----	-------------------------------------

Returns

Returns true if a modification occurred, and sets the perform modified flag based on that result.

Reimplemented from [seq64::Seq24PerfInput](#).

10.15.2.7 on_left_button_pressed()

```
bool seq64::FruityPerfInput::on_left_button_pressed (
    GdkEventButton * ev ) [private], [virtual]
```

It can handle splitting triggers (?), adding notes, and the following clicks to resize the event, or move it, depending on where clicked:

- clicked left side: begin a grow/shrink for the left side
- clicked right side: grow/shrink the right side
- clicked in the middle - move it

I don't get it, though... all three buttons are handled in the generic button-press callback. Oh, this is just a helper function.

Parameters

<i>ev</i>	The left-button-press event to process.
-----------	---

Returns

Now returns true if a modification occurred.

10.15.2.8 on_right_button_pressed()

```
bool seq64::FruityPerfInput::on_right_button_pressed (  
    GdkEventButton * ev ) [private], [virtual]
```

I don't get it, though... all three buttons are handled in the generic button-press callback. Oh, this is a helper function.

Parameters

<i>ev</i>	The right-button-press event to process.
-----------	--

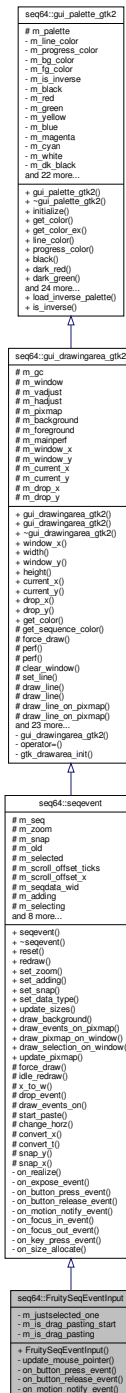
Returns

Returns true if a modification occurred.

10.16 seq64::FruitySeqEventInput Class Reference

This class implements the interaction methods for the "fruity" mode of operation in the event panel of the seqroll.

Inheritance diagram for seq64::FruitySeqEventInput:



Public Member Functions

- [FruitySeqEventInput](#) ([perform](#) &p, [sequence](#) &seq, int zoom, int snap, [seqdata](#) &seqdata_wid, [Gtk::Adjustment](#) &hadjust)

Private Member Functions

- virtual void [update_mouse_pointer](#) ()

Provides support for a context-sensitive mouse.

- virtual bool [on_button_press_event](#) (GdkEventButton *ev)
Implements the on-button-press event callback.
- virtual bool [on_button_release_event](#) (GdkEventButton *ev)
Implements the on-button-release callback.
- virtual bool [on_motion_notify_event](#) (GdkEventMotion *ev)
Implements the on-motion-notify callback.

Private Attributes

- bool [m_justselected_one](#)
Indicates that the left mouse button was click to start a selection.
- bool [m_is_drag_pasting_start](#)
Set to true when the mouse button is pressed and we're starting to drag some notes to move them and paste them to a different location.
- bool [m_is_drag_pasting](#)
Set to true when the left mouse button is pressed for dragging and pasting, set to false when the mouse button is released to drop the pasted items.

Additional Inherited Members

10.16.1 Constructor & Destructor Documentation

10.16.1.1 FruitySeqEventInput()

```
seq64::FruitySeqEventInput::FruitySeqEventInput (
    perform & p,
    sequence & seq,
    int zoom,
    int snap,
    seqdata & seqdata_wid,
    Gtk::Adjustment & hadjust )
```

10.16.2 Member Function Documentation

10.16.2.1 update_mouse_pointer()

```
void seq64::FruitySeqEventInput::update_mouse_pointer ( ) [private], [virtual]
```

10.16.2.2 on_button_press_event()

```
bool seq64::FruitySeqEventInput::on_button_press_event (
    GdkEventButton * ev ) [private], [virtual]
```

Handles dragging and other actions.

The first thing is to set the values for dragging, then reset the box that holds the dirty redraw spot. If pasting, undo the clipboard, and paste the selected events.

Otherwise, process the mouse actions. The current steps shown below are my initial guesses, to be verified at some point.

1. Left button:

(a) Click:

- i. A click and release without a drag, or without a Ctrl-Shift, deselects the events.
- ii. A direct click on an event selects only that event.

(b) Click-drag:

- i. If events already selected, adds note and length to the selected notes.
- ii. Otherwise, select the notes and events.
- iii. If no events selected in the end, undo the selection.

• Ctrl-left button:

- TODO.

The opening part of this function matches that of [Seq24SeqEventInput::on_button_press_event\(\)](#).

Parameters

<i>ev</i>	The button event for the press of a mouse button.
-----------	---

Returns

Returns true if a modification was made. It used to return true all the time.

Reimplemented from [seq64::segevent](#).

10.16.2.3 on_button_release_event()

```
bool seq64::FruitySeqEventInput::on_button_release_event (
    GdkEventButton * ev ) [private], [virtual]
```

Parameters

<i>ev</i>	The button event for the press of a mouse button.
-----------	---

Returns

Returns true if a modification was made. It used to return true all the time.

Reimplemented from [seq64::sequevent](#).

10.16.2.4 on_motion_notify_event()

```
bool seq64::FruitySeqEventInput::on_motion_notify_event (
    GdkEventMotion * ev ) [private], [virtual]
```

Parameters

<code>ev</code>	The button event for the press of a mouse button.
-----------------	---

Returns

Returns true if a modification occurred, and sets the perform modified flag based on that result.

Reimplemented from [seq64::sequevent](#).

10.16.3 Field Documentation

10.16.3.1 m_justselected_one

```
bool seq64::FruitySeqEventInput::m_justselected_one [private]
```

10.16.3.2 m_is_drag_pasting_start

```
bool seq64::FruitySeqEventInput::m_is_drag_pasting_start [private]
```

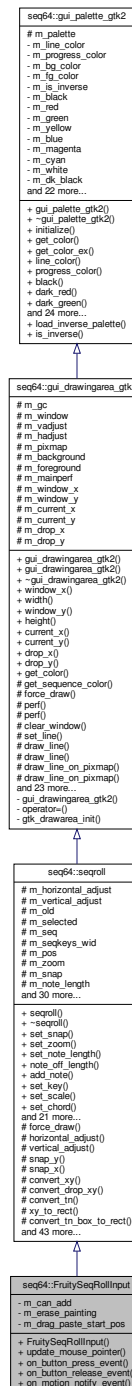
10.16.3.3 m_is_drag_pasting

```
bool seq64::FruitySeqEventInput::m_is_drag_pasting [private]
```

10.17 seq64::FruitySeqRollInput Class Reference

Implements the fruity mouse interaction paradigm for the seqroll.

Inheritance diagram for seq64::FruitySeqRollInput:



Public Member Functions

- [FruitySeqRollInput](#) ([perform](#) &perf, [sequence](#) &seq, int zoom, int snap, [seqkeys](#) &seqkeys_wid, int pos, Gtk↔::Adjustment &hadjust, Gtk::Adjustment &vadjust)

Default constructor.

- virtual void [update_mouse_pointer](#) (bool isadding)
Updates the mouse pointer, implementing a context-sensitive mouse.
- virtual bool [on_button_press_event](#) (GdkEventButton *ev)
Implements the fruity on-button-press callback.
- virtual bool [on_button_release_event](#) (GdkEventButton *ev)
Implements the fruity handling for the on-button-release event.
- virtual bool [on_motion_notify_event](#) (GdkEventMotion *ev)
Implements the fruity handling for the on-motion-notify event.

Private Attributes

- bool [m_can_add](#)
Indicates adding status peculiar to the fruity mode.
- bool [m_erase_painting](#)
Set to true if we hold the right mouse button down (in "fruity" mode) and start to drag the mouse around, erasing notes.
- int [m_drag_paste_start_pos](#) [2]
Holds the original position of the mouse when ctrl-left-click-drag is done, and is used to make sure that the action doesn't occur until a movement of at least 6 pixels has occurred, to avoid unintended actions caused by minimal jitter in the user's hands.

Additional Inherited Members

10.17.1 Constructor & Destructor Documentation

10.17.1.1 FruitySeqRollInput()

```
seq64::FruitySeqRollInput::FruitySeqRollInput (
    perform & perf,
    sequence & seq,
    int zoom,
    int snap,
    seqkeys & seqkeys_wid,
    int pos,
    Gtk::Adjustment & hadjust,
    Gtk::Adjustment & vadjust )
```

10.17.2 Member Function Documentation

10.17.2.1 update_mouse_pointer()

```
void seq64::FruitySeqRollInput::update_mouse_pointer (
    bool isadding ) [virtual]
```

This code is very similar to the base class version.

10.17.2.2 on_button_press_event()

```
bool seq64::FruitySeqRollInput::on_button_press_event (
    GdkEventButton * ev ) [virtual]
```

This function now uses the needs_update flag to determine if the perform object should modify().

Parameters

ev	The button event.
----	-------------------

Returns

Returns the value of needs_update. It used to return only true.

Reimplemented from [seq64::seqroll](#).

10.17.2.3 on_button_release_event()

```
bool seq64::FruitySeqRollInput::on_button_release_event (
    GdkEventButton * ev ) [virtual]
```

Parameters

ev	The button event.
----	-------------------

Returns

Returns the value of needs_update. It used to return only true.

If in moving mode, adjust for snap and convert deltas into screen coordinates. Since delta_note was from delta_y, it will be flipped (delta_y[0] = note[127], etc.), so we have to adjust.

Reimplemented from [seq64::seqroll](#).

10.17.2.4 on_motion_notify_event()

```
bool seq64::FruitySeqRollInput::on_motion_notify_event (
    GdkEventMotion * ev ) [virtual]
```

Why not just inherit and save all these indirect accesses to the seqroll? Well, that would make it more difficult to change the mode of interaction, in the Options menu, on the fly. We did it anyway. One will hardly ever change the mouse interaction mode.

Parameters

<i>ev</i>	The motion event.
-----------	-------------------

Returns

Returns the value of `needs_update`.

In "fruity" interaction mode, ctrl-left-click-drag on selected note(s) starts a copy/unselect/paste. Doesn't begin the paste until the mouse moves a few pixels, to filter out the unsteady hand.

Reimplemented from [seq64::seqroll](#).

10.17.3 Field Documentation

10.17.3.1 `m_can_add`

```
bool seq64::FruitySeqRollInput::m_can_add [private]
```

Currently always true.

10.17.3.2 `m_erase_painting`

```
bool seq64::FruitySeqRollInput::m_erase_painting [private]
```

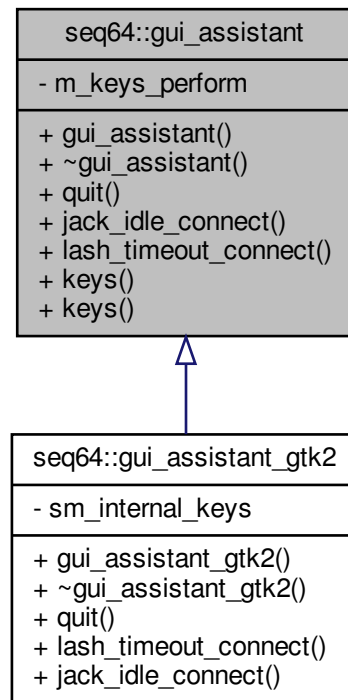
10.17.3.3 `m_drag_paste_start_pos`

```
int seq64::FruitySeqRollInput::m_drag_paste_start_pos[2] [private]
```

10.18 seq64::gui_assistant Class Reference

This class provides an interface for some of the GUI support needed in Sequencer64.

Inheritance diagram for seq64::gui_assistant:



Public Member Functions

- `gui_assistant (keys_perform &kp)`
This constructor wires in some externally (for now) created objects.
- `virtual ~gui_assistant ()`
Stock base-class implementation of a virtual destructor.
- `virtual void quit ()`
Handles the GUI's exiting.
- `virtual void jack_idle_connect (jack_assistant &)`
Handles connecting the "idle" signal to the session-event function.
- `virtual void lash_timeout_connect (lash *)`
Handles connecting the "timeout" signal to the process-event function.
- `const keys_perform & keys () const`
'Getter' function for member m_keys_perform The const getter.
- `keys_perform & keys ()`
'Getter' function for member m_keys_perform The un-const getter.

Private Attributes

- [keys_perform](#) & [m_keys_perform](#)

Provides a reference to the app-specific GUI-specific keys_perform-derived object that an application is going to use for handling sequence-control keys.

10.18.1 Detailed Description

It also contain a number of helper objects that all kind of go together; only this assistant object will need to be passed around (by non-GUI code).

10.18.2 Constructor & Destructor Documentation

10.18.2.1 gui_assistant()

```
seq64::gui_assistant::gui_assistant (
    keys\_perform & kp )
```

Parameters

<i>kp</i>	Provides a set of key codes to be used by the perform object to control patterns and their performance.
-----------	---

10.18.2.2 ~gui_assistant()

```
virtual seq64::gui_assistant::~~gui_assistant ( ) [inline], [virtual]
```

10.18.3 Member Function Documentation

10.18.3.1 quit()

```
virtual void seq64::gui_assistant::quit ( ) [inline], [virtual]
```

However, with the new command-line support, we now need a default implementation that does nothing.

Reimplemented in [seq64::gui_assistant_gtk2](#).

10.18.3.2 jack_idle_connect()

```
virtual void seq64::gui_assistant::jack_idle_connect (
    jack_assistant & ) [inline], [virtual]
```

But we now need a default implementation that does nothing.

Reimplemented in [seq64::gui_assistant_gtk2](#).

10.18.3.3 lash_timeout_connect()

```
virtual void seq64::gui_assistant::lash_timeout_connect (
    lash * ) [inline], [virtual]
```

But we now need a default implementation that does nothing.

Reimplemented in [seq64::gui_assistant_gtk2](#).

10.18.3.4 keys() [1/2]

```
const keys_perform& seq64::gui_assistant::keys ( ) const [inline]
```

10.18.3.5 keys() [2/2]

```
keys_perform& seq64::gui_assistant::keys ( ) [inline]
```

10.18.4 Field Documentation

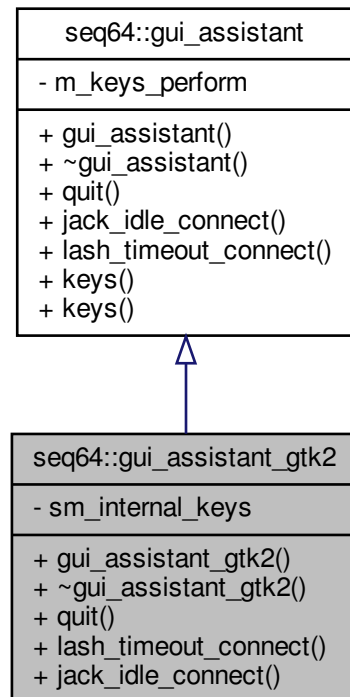
10.18.4.1 m_keys_perform

```
keys_perform& seq64::gui_assistant::m_keys_perform [private]
```

10.19 seq64::gui_assistant_gtk2 Class Reference

This class provides an interface for some of the Gtk/Gdk/Glib support needed in Sequencer64.

Inheritance diagram for seq64::gui_assistant_gtk2:



Public Member Functions

- [gui_assistant_gtk2 \(\)](#)
This class provides an interface for some of the Gtk/Gdk/Glib support needed in Sequencer64.
- virtual [~gui_assistant_gtk2 \(\)](#)
Virtual classes require a virtual destructor.
- virtual void [quit \(\)](#)
Calls the Glib Main object's [quit\(\)](#) function.
- virtual void [lash_timeout_connect \(lash *lashobject\)](#)
Connects the LASH timeout-event callback to the Glib timeout object.
- virtual void [jack_idle_connect \(jack_assistant &jack\)](#)
Connects the JACK session-event callback to the Glib idle object.

Static Private Attributes

- static [keys_perform_gtk2 sm_internal_keys](#)
Provides a pre-made [keys_perform](#) object.

10.19.1 Constructor & Destructor Documentation

10.19.1.1 `gui_assistant_gtk2()`

```
seq64::gui_assistant_gtk2::gui_assistant_gtk2 ( )
```

10.19.1.2 `~gui_assistant_gtk2()`

```
virtual seq64::gui_assistant_gtk2::~~gui_assistant_gtk2 ( ) [inline], [virtual]
```

10.19.2 Member Function Documentation

10.19.2.1 `quit()`

```
void seq64::gui_assistant_gtk2::quit ( ) [virtual]
```

Reimplemented from [seq64::gui_assistant](#).

10.19.2.2 `lash_timeout_connect()`

```
void seq64::gui_assistant_gtk2::lash_timeout_connect (
    lash * lashobject ) [virtual]
```

The time-out value is set to 250 ms.

Reimplemented from [seq64::gui_assistant](#).

10.19.2.3 `jack_idle_connect()`

```
void seq64::gui_assistant_gtk2::jack_idle_connect (
    jack\_assistant & jack ) [virtual]
```

If JACK session support is not enabled, we might emit a message. This mainly prevents a compiler warning about an unused parameter.

Reimplemented from [seq64::gui_assistant](#).

10.19.3.1 sm_internal_keys

This object is set into the reference provided in the `gui_assistant` base class.

Implements the basic drawing areas of the application.

[illegible]

Public Member Functions

- [gui_drawingarea_gtk2](#) ([perform](#) &p, int [window_x](#)=0, int [window_y](#)=0)
Perform-only constructor.
- [gui_drawingarea_gtk2](#) ([perform](#) &a_perf, Gtk::Adjustment &a_hadjust, Gtk::Adjustment &a_vadjust, int [window_x](#)=0, int [window_y](#)=0)
Principal constructor.
- virtual [~gui_drawingarea_gtk2](#) ()
Provides a destructor to delete allocated objects.
- int [window_x](#) () const
'Getter' function for member m_window_x
- int [width](#) () const
'Getter' function for member m_window_x This one matches what Qt 5 calls it.
- int [window_y](#) () const
'Getter' function for member m_window_y
- int [height](#) () const
'Getter' function for member m_window_y This one matches what Qt 5 calls it.
- int [current_x](#) () const
'Getter' function for member m_current_x
- int [current_y](#) () const
'Getter' function for member m_current_y
- int [drop_x](#) () const
'Getter' function for member m_drop_x
- int [drop_y](#) () const
'Getter' function for member m_drop_y
- const [Color](#) & [get_color](#) (PaletteColor c) const

Protected Member Functions

- const [gui_palette_gtk2::Color](#) & [get_sequence_color](#) (int seqnum) const
Color get_color (PaletteColor c, double h, double s = 0.65, double v = 1.0) const { return m_palette.get_color(c, h, s, v); }.
- virtual void [force_draw](#) ()
Provides a common function for redrawing.
- [perform](#) & [perf](#) ()
'Getter' function for member m_mainperf
- const [perform](#) & [perf](#) () const
'Getter' function for member m_mainperf, const version
- void [clear_window](#) ()
Clears the main window.
- void [set_line](#) (Gdk::LineStyle ls, int [width](#)=1)
A small wrapper function for readability in line-drawing.
- void [draw_line](#) (int x1, int y1, int x2, int y2)
A small wrapper function to draw a line on the window.
- void [draw_line](#) (const [Color](#) &c, int x1, int y1, int x2, int y2)
A small wrapper function to draw a line on the window after setting the given foreground color.
- void [draw_line_on_pixmap](#) (int x1, int y1, int x2, int y2)
A small wrapper function to draw a line on the pixmap.
- void [draw_line_on_pixmap](#) (const [Color](#) &c, int x1, int y1, int x2, int y2)
A small wrapper function to draw a line on the pixmap after setting the given foreground color.
- void [draw_line](#) (Glib::RefPtr< Gdk::Pixmap > &pixmap, int x1, int y1, int x2, int y2)

- A small wrapper function to draw a line on any pixmap (not a drawable, though, due to a compiler error after setting the given foreground color.*

 - void [draw_line](#) (Glib::RefPtr< Gdk::Pixmap > &pixmap, const [Color](#) &c, int x1, int y1, int x2, int y2)

A small wrapper function to draw a line on the pixmap after setting the given foreground color.

 - void [draw_line](#) (Glib::RefPtr< Gdk::Drawable > &drawable, int x1, int y1, int x2, int y2)

A small wrapper function to draw a line on any pixmap (not a drawable, though, due to a compiler error after setting the given foreground color.

 - void [draw_line](#) (Glib::RefPtr< Gdk::Drawable > &drawable, const [Color](#) &c, int x1, int y1, int x2, int y2)

A small wrapper function to draw a line on the drawable after setting the given foreground color.

 - void [render_string](#) (int x, int y, const std::string &s, [font::Color](#) color, bool invert=false)

A small wrapper function for readability in string-drawing to the window.

 - void [render_string_on_pixmap](#) (int x, int y, const std::string &s, [font::Color](#) color, bool invert=false)

A small wrapper function for readability in string-drawing to the pixmap.

 - void [draw_rectangle](#) (int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing on the window.

 - void [draw_rectangle](#) (const [Color](#) &c, int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing.

 - void [draw_rectangle](#) (Glib::RefPtr< Gdk::Drawable > &drawable, int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing on a "drawable" context, where the foreground color has already been specified.

 - void [draw_rectangle](#) (Glib::RefPtr< Gdk::Drawable > &drawable, const [Color](#) &c, int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing on any drawable context.

 - void [draw_rectangle](#) (Glib::RefPtr< Gdk::Pixmap > &pixmap, int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing on a "pixmap" context, where the foreground color has already been specified.

 - void [draw_rectangle](#) (Glib::RefPtr< Gdk::Pixmap > &pixmap, const [Color](#) &c, int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing on any pixmap context.

 - void [draw_rectangle_on_pixmap](#) (int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing on the pixmap.

 - void [draw_rectangle_on_pixmap](#) (const [Color](#) &c, int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing on the pixmap.

 - void [draw_normal_rectangle_on_pixmap](#) (int x, int y, int lx, int ly, bool fill=true)

A small wrapper function for readability in box-drawing on the pixmap.

 - void [draw_drawable](#) (int xsrc, int ysrc, int xdest, int ydest, int [width](#), int [height](#))

Provides the most common use case for redrawing.

 - void [scroll_hadjust](#) (Gtk::Adjustment &hadjust, double step)

This function provides optimization for the [on_scroll_event\(\)](#) functions, and should provide support for having the [seqedit/seqroll/seqtime/seqdata](#) panes follow the scrollbar, in a future upgrade (now partly in place).

 - void [scroll_vadjust](#) (Gtk::Adjustment &vadjust, double step)

This function is the vertical version of the [scroll_hadjust\(\)](#) function, intended for adding keystroke vertical scrolling using the Page-Up and Page-Down keys, as a new feature of Sequencer64.

 - void [scroll_hset](#) (Gtk::Adjustment &hadjust, double value)
 - void [scroll_vset](#) (Gtk::Adjustment &vadjust, double value)
 - void [set_current_drop_x](#) (int x)

Sets the current x value and the drop x value.

 - void [set_current_drop_y](#) (int y)

Sets the current y value and the drop y value.

 - void [on_realize](#) ()

For this GTK callback, on realization of window, initialize the shiz.

Protected Attributes

- `Glib::RefPtr< Gdk::GC > m_gc`
The graphics context, which is required for ever drawing and rendering operation.
- `Glib::RefPtr< Gdk::Window > m_window`
Provides the default "window".
- `Gtk::Adjustment & m_vadjust`
Provides an object for vertical "adjustments".
- `Gtk::Adjustment & m_hadjust`
Provides an object for horizontal "adjustments".
- `Glib::RefPtr< Gdk::Pixmap > m_pixmap`
Provides the default "pixmap".
- `Glib::RefPtr< Gdk::Pixmap > m_background`
Another pixmap, used for backgrounds.
- `Glib::RefPtr< Gdk::Pixmap > m_foreground`
Another pixmap, used for foregrounds.
- `perform & m_mainperf`
A frequent hook into the main perform object.
- `int m_window_x`
Window sizes.
- `int m_window_y`
Window height value.
- `int m_current_x`
The x and y value of the current location of the mouse (during dragging?)
- `int m_current_y`
Current mouse y value.
- `int m_drop_x`
These values are used when roping and highlighting a bunch of events.
- `int m_drop_y`
Current mouse y-drop value.

Private Member Functions

- `gui_drawingarea_gtk2 (const gui_drawingarea_gtk2 &)`
- `gui_drawingarea_gtk2 & operator= (const gui_drawingarea_gtk2 &)`
- `void gtk_drawarea_init ()`
Does basic initialization for each of the constructors.

Additional Inherited Members

10.20.1 Detailed Description

Note that this class really "isn't" a `gui_palette_gtk2`; it should simply "have" one. But that base class must be derived from `Gtk::DrawingArea`. We don't want to waste some space by using a "has-a" relationship, and also put up with having to access the palette indirectly. So, in this case, we tolerate the less strict implementation.

10.20.2 Constructor & Destructor Documentation

10.20.2.1 gui_drawingarea_gtk2() [1/3]

```
seq64::gui_drawingarea_gtk2::gui_drawingarea_gtk2 (
    const gui_drawingarea_gtk2 & ) [private]
```

10.20.2.2 gui_drawingarea_gtk2() [2/3]

```
seq64::gui_drawingarea_gtk2::gui_drawingarea_gtk2 (
    perform & p,
    int window_x = 0,
    int window_y = 0 )
```

10.20.2.3 gui_drawingarea_gtk2() [3/3]

```
seq64::gui_drawingarea_gtk2::gui_drawingarea_gtk2 (
    perform & a_perf,
    Gtk::Adjustment & a_hadjust,
    Gtk::Adjustment & a_vadjust,
    int window_x = 0,
    int window_y = 0 )
```

10.20.2.4 ~gui_drawingarea_gtk2()

```
seq64::gui_drawingarea_gtk2::~~gui_drawingarea_gtk2 ( ) [virtual]
```

10.20.3 Member Function Documentation

10.20.3.1 operator=()

```
gui_drawingarea_gtk2& seq64::gui_drawingarea_gtk2::operator= (
    const gui_drawingarea_gtk2 & ) [private]
```

10.20.3.2 window_x()

```
int seq64::gui_drawingarea_gtk2::window_x ( ) const [inline]
```

10.20.3.3 width()

```
int seq64::gui_drawingarea_gtk2::width ( ) const [inline]
```

10.20.3.4 window_y()

```
int seq64::gui_drawingarea_gtk2::window_y ( ) const [inline]
```

10.20.3.5 height()

```
int seq64::gui_drawingarea_gtk2::height ( ) const [inline]
```

10.20.3.6 current_x()

```
int seq64::gui_drawingarea_gtk2::current_x ( ) const [inline]
```

10.20.3.7 current_y()

```
int seq64::gui_drawingarea_gtk2::current_y ( ) const [inline]
```

10.20.3.8 drop_x()

```
int seq64::gui_drawingarea_gtk2::drop_x ( ) const [inline]
```

10.20.3.9 drop_y()

```
int seq64::gui_drawingarea_gtk2::drop_y ( ) const [inline]
```

10.20.3.10 get_color()

```
const Color& seq64::gui_drawingarea_gtk2::get_color (
    PaletteColor c ) const [inline]
```

10.20.3.11 `get_sequence_color()`

```
const gui\_palette\_gtk2::Color & seq64::gui_drawingarea_gtk2::get_sequence_color (
    int seqnum ) const [protected]
```

EXPERIMENTAL.

EXPERIMENTAL

10.20.3.12 `force_draw()`

```
virtual void seq64::gui_drawingarea_gtk2::force_draw ( ) [inline], [protected], [virtual]
```

This function forces a redraw. Some classes extend this function.

Reimplemented in [seq64::seqroll](#), [seq64::seqevent](#), and [seq64::seqkeys](#).

10.20.3.13 `perf()` [1/2]

```
perform& seq64::gui_drawingarea_gtk2::perf ( ) [inline], [protected]
```

10.20.3.14 `perf()` [2/2]

```
const perform& seq64::gui_drawingarea_gtk2::perf ( ) const [inline], [protected]
```

10.20.3.15 `clear_window()`

```
void seq64::gui_drawingarea_gtk2::clear_window ( ) [inline], [protected]
```

One less need to access `m_window` directly.

10.20.3.16 `set_line()`

```
void seq64::gui_drawingarea_gtk2::set_line (
    Gdk::LineStyle ls,
    int width = 1 ) [inline], [protected]
```

Sets the attributes of a line to be drawn.

Parameters

<i>ls</i>	Provides the Gtk-specific line style.
<i>width</i>	Provides the width of the line to be drawn. It defaults to the most common value, 1.

10.20.3.17 draw_line() [1/6]

```
void seq64::gui_drawingarea_gtk2::draw_line (
    int x1,
    int y1,
    int x2,
    int y2 ) [inline], [protected]
```

Parameters

<i>x1</i>	The x coordinate of the starting point.
<i>y1</i>	The y coordinate of the starting point.
<i>x2</i>	The x coordinate of the ending point.
<i>y2</i>	The y coordinate of the ending point.

10.20.3.18 draw_line() [2/6]

```
void seq64::gui_drawingarea_gtk2::draw_line (
    const Color & c,
    int x1,
    int y1,
    int x2,
    int y2 ) [protected]
```

Parameters

<i>c</i>	The foreground color in which to draw the line.
<i>x1</i>	The x coordinate of the starting point.
<i>y1</i>	The y coordinate of the starting point.
<i>x2</i>	The x coordinate of the ending point.
<i>y2</i>	The y coordinate of the ending point.

10.20.3.19 draw_line_on_pixmap() [1/2]

```
void seq64::gui_drawingarea_gtk2::draw_line_on_pixmap (
    int x1,
    int y1,
    int x2,
    int y2 ) [inline], [protected]
```

Parameters

<i>x1</i>	The x coordinate of the starting point.
-----------	---

Parameters

<i>y1</i>	The y coordinate of the starting point.
<i>x2</i>	The x coordinate of the ending point.
<i>y2</i>	The y coordinate of the ending point.

10.20.3.20 draw_line_on_pixmap() [2/2]

```
void seq64::gui_drawingarea_gtk2::draw_line_on_pixmap (
    const Color & c,
    int x1,
    int y1,
    int x2,
    int y2 ) [protected]
```

Parameters

<i>c</i>	The foreground color in which to draw the line.
<i>x1</i>	The x coordinate of the starting point.
<i>y1</i>	The y coordinate of the starting point.
<i>x2</i>	The x coordinate of the ending point.
<i>y2</i>	The y coordinate of the ending point.

10.20.3.21 draw_line() [3/6]

```
void seq64::gui_drawingarea_gtk2::draw_line (
    Glib::RefPtr< Gdk::Pixmap > & pixmap,
    int x1,
    int y1,
    int x2,
    int y2 ) [inline], [protected]
```

Parameters

<i>pixmap</i>	Provides the Gdk::Pixmap pointer needed to draw the line.
<i>x1</i>	The x coordinate of the starting point.
<i>y1</i>	The y coordinate of the starting point.
<i>x2</i>	The x coordinate of the ending point.
<i>y2</i>	The y coordinate of the ending point.

10.20.3.22 draw_line() [4/6]

```
void seq64::gui_drawingarea_gtk2::draw_line (
    Glib::RefPtr< Gdk::Pixmap > & pixmap,
    const Color & c,
    int x1,
    int y1,
    int x2,
    int y2 ) [protected]
```

Parameters

<i>pixmap</i>	Provides the Gdk::Drawable pointer needed to draw the line.
<i>c</i>	The foreground color in which to draw the line.
<i>x1</i>	The x coordinate of the starting point.
<i>y1</i>	The y coordinate of the starting point.
<i>x2</i>	The x coordinate of the ending point.
<i>y2</i>	The y coordinate of the ending point.

10.20.3.23 draw_line() [5/6]

```
void seq64::gui_drawingarea_gtk2::draw_line (
    Glib::RefPtr< Gdk::Drawable > & drawable,
    int x1,
    int y1,
    int x2,
    int y2 ) [inline], [protected]
```

Parameters

<i>drawable</i>	Provides the Gdk::Drawable pointer needed to draw the line.
<i>x1</i>	The x coordinate of the starting point.
<i>y1</i>	The y coordinate of the starting point.
<i>x2</i>	The x coordinate of the ending point.
<i>y2</i>	The y coordinate of the ending point.

10.20.3.24 draw_line() [6/6]

```
void seq64::gui_drawingarea_gtk2::draw_line (
    Glib::RefPtr< Gdk::Drawable > & drawable,
    const Color & c,
    int x1,
    int y1,
    int x2,
    int y2 ) [protected]
```

Parameters

<i>drawable</i>	Provides the Gdk::Drawable pointer needed to draw the line.
<i>c</i>	The foreground color in which to draw the line.
<i>x1</i>	The x coordinate of the starting point.
<i>y1</i>	The y coordinate of the starting point.
<i>x2</i>	The x coordinate of the ending point.
<i>y2</i>	The y coordinate of the ending point.

10.20.3.25 render_string()

```
void seq64::gui_drawingarea_gtk2::render_string (
    int x,
    int y,
    const std::string & s,
    font::Color color,
    bool invert = false ) [inline], [protected]
```

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>s</i>	The string to be drawn.
<i>color</i>	The color with which to draw the string.
<i>invert</i>	If true, apply color inversion, if active. Defaults to false.

10.20.3.26 render_string_on_pixmap()

```
void seq64::gui_drawingarea_gtk2::render_string_on_pixmap (
    int x,
    int y,
    const std::string & s,
    font::Color color,
    bool invert = false ) [inline], [protected]
```

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>s</i>	The string to be drawn.
<i>color</i>	The color with which to draw the string.
<i>invert</i>	If true, apply color inversion, if active. Defaults to false.

10.20.3.27 draw_rectangle() [1/6]

```
void seq64::gui_drawingarea_gtk2::draw_rectangle (
    int x,
    int y,
    int lx,
    int ly,
    bool fill = true ) [inline], [protected]
```

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.28 draw_rectangle() [2/6]

```
void seq64::gui_drawingarea_gtk2::draw_rectangle (
    const Color & c,
    int x,
    int y,
    int lx,
    int ly,
    bool fill = true ) [protected]
```

It adds setting the foreground color to the [draw_rectangle\(\)](#) function.

Parameters

<i>c</i>	Provides the foreground color to set.
<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.29 draw_rectangle() [3/6]

```
void seq64::gui_drawingarea_gtk2::draw_rectangle (
    Glib::RefPtr< Gdk::Drawable > & drawable,
    int x,
    int y,
```

```

    int lx,
    int ly,
    bool fill = true ) [inline], [protected]

```

Parameters

<i>drawable</i>	The object on which to draw the rectangle.
<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.30 draw_rectangle() [4/6]

```

void seq64::gui_drawingarea_gtk2::draw_rectangle (
    Glib::RefPtr< Gdk::Drawable > & drawable,
    const Color & c,
    int x,
    int y,
    int lx,
    int ly,
    bool fill = true ) [protected]

```

It also supports setting the foreground color to the [draw_rectangle\(\)](#) function.

We have a number of such functions: for the main window, for the main pixmap, and for any drawing surface. Is the small bit of conciseness worth it?

Parameters

<i>drawable</i>	The surface on which to draw the box.
<i>c</i>	Provides the foreground color to set.
<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.31 draw_rectangle() [5/6]

```

void seq64::gui_drawingarea_gtk2::draw_rectangle (
    Glib::RefPtr< Gdk::Pixmap > & pixmap,
    int x,

```

```

    int y,
    int lx,
    int ly,
    bool fill = true ) [inline], [protected]

```

Parameters

<i> pixmap </i>	The object on which to draw the rectangle.
<i> x </i>	The x-coordinate of the origin.
<i> y </i>	The y-coordinate of the origin.
<i> lx </i>	The width of the box.
<i> ly </i>	The height of the box.
<i> fill </i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.32 draw_rectangle() [6/6]

```

void seq64::gui_drawingarea_gtk2::draw_rectangle (
    Glib::RefPtr< Gdk::Pixmap > & pixmap,
    const Color & c,
    int x,
    int y,
    int lx,
    int ly,
    bool fill = true ) [protected]

```

It also supports setting the foreground color to the [draw_rectangle\(\)](#) function.

We have a number of such functions: for the main window, for the main pixmap, and for any drawing surface. Is the small bit of conciseness worth it?

Parameters

<i> pixmap </i>	The surface on which to draw the box.
<i> c </i>	Provides the foreground color to set.
<i> x </i>	The x-coordinate of the origin.
<i> y </i>	The y-coordinate of the origin.
<i> lx </i>	The width of the box.
<i> ly </i>	The height of the box.
<i> fill </i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.33 draw_rectangle_on_pixmap() [1/2]

```

void seq64::gui_drawingarea_gtk2::draw_rectangle_on_pixmap (
    int x,

```

```

    int y,
    int lx,
    int ly,
    bool fill = true ) [inline], [protected]

```

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.34 draw_rectangle_on_pixmap() [2/2]

```

void seq64::gui_drawingarea_gtk2::draw_rectangle_on_pixmap (
    const Color & c,
    int x,
    int y,
    int lx,
    int ly,
    bool fill = true ) [protected]

```

It adds setting the foreground color to the [draw_rectangle\(\)](#) function.

Parameters

<i>c</i>	Provides the foreground color to set.
<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.35 draw_normal_rectangle_on_pixmap()

```

void seq64::gui_drawingarea_gtk2::draw_normal_rectangle_on_pixmap (
    int x,
    int y,
    int lx,
    int ly,
    bool fill = true ) [protected]

```

It uses [Gtk](#) to get the proper background styling for the rectangle.

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

10.20.3.36 `draw_drawable()`

```
void seq64::gui_drawingarea_gtk2::draw_drawable (
    int xsrc,
    int ysrc,
    int xdest,
    int ydest,
    int width,
    int height ) [inline], [protected]
```

10.20.3.37 `scroll_hadjust()`

```
void seq64::gui_drawingarea_gtk2::scroll_hadjust (
    Gtk::Adjustment & hadjust,
    double step ) [protected]
```

This function is currently duplicated in the [gui_drawingarea_gtk2](#) and [gui_window_gtk2](#) modules.

Parameters

<i>hadjust</i>	Provides a reference to the adjustment object to be adjusted. Do we really need this to be a parameter? Why not just use the <code>m_hadjust</code> member? (Note that this member is not present in the similar gui_window_gtk2 class.)
<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. If negative, the adjustment is leftward. If positive, the adjustment is rightward. It can be the value of <code>m_hadjust->get_step_increment()</code> , or provided especially to keep up with the progress bar.

10.20.3.38 `scroll_vadjust()`

```
void seq64::gui_drawingarea_gtk2::scroll_vadjust (
    Gtk::Adjustment & vadjust,
    double step ) [protected]
```


Parameters

<i>vadjust</i>	Provides a reference to the adjustment object to be adjusted.
<i>step</i>	Provides the step value to use for adjusting the vertical scrollbar. If negative, the adjustment is upward. If positive, the adjustment is downward. It can be the value of <code>m_vadjust->get_step_increment()</code> .

10.20.3.39 scroll_hset()

```
void seq64::gui_drawingarea_gtk2::scroll_hset (  
    Gtk::Adjustment & hadjust,  
    double value ) [protected]
```

10.20.3.40 scroll_vset()

```
void seq64::gui_drawingarea_gtk2::scroll_vset (  
    Gtk::Adjustment & vadjust,  
    double value ) [protected]
```

10.20.3.41 set_current_drop_x()

```
void seq64::gui_drawingarea_gtk2::set_current_drop_x (  
    int x ) [inline], [protected]
```

Parameters

<i>x</i>	The x value to be set.
----------	------------------------

10.20.3.42 set_current_drop_y()

```
void seq64::gui_drawingarea_gtk2::set_current_drop_y (  
    int y ) [inline], [protected]
```

Parameters

<i>y</i>	The y value to be set.
----------	------------------------

10.20.3.43 gtk_drawarea_init()

```
void seq64::gui_drawingarea_gtk2::gtk_drawarea_init ( ) [private]
```

10.20.3.44 on_realize()

```
void seq64::gui_drawingarea_gtk2::on_realize ( ) [protected]
```

It allocates any additional resources that weren't initialized in the constructor.

10.20.4 Field Documentation

10.20.4.1 m_gc

```
Glib::RefPtr<Gdk::GC> seq64::gui_drawingarea_gtk2::m_gc [protected]
```

10.20.4.2 m_window

```
Glib::RefPtr<Gdk::Window> seq64::gui_drawingarea_gtk2::m_window [protected]
```

Wrapper functions with undecorated wrapper names are used for accessing this item. We hope to be able to hide this items completely some day.

10.20.4.3 m_vadjust

```
Gtk::Adjustment& seq64::gui_drawingarea_gtk2::m_vadjust [protected]
```

10.20.4.4 m_hadjust

```
Gtk::Adjustment& seq64::gui_drawingarea_gtk2::m_hadjust [protected]
```

10.20.4.5 m_pixmap

```
Glib::RefPtr<Gdk::Pixmap> seq64::gui_drawingarea_gtk2::m_pixmap [protected]
```

Wrapper functions with undecorated wrapper names are used for accessing this item. We hope to be able to hide this items completely some day.

10.20.4.6 m_background

Glib::RefPtr<Gdk::Pixmap> seq64::gui_drawingarea_gtk2::m_background [protected]

Our wrappers still leave this member exposed (giggle).

10.20.4.7 m_foreground

Glib::RefPtr<Gdk::Pixmap> seq64::gui_drawingarea_gtk2::m_foreground [protected]

Our wrappers still leave this member exposed.

10.20.4.8 m_mainperf

`perform`& seq64::gui_drawingarea_gtk2::m_mainperf [protected]

We could move this into yet another base class, since a number of classes don't need it. Probably not worth the effort at this time.

10.20.4.9 m_window_x

int seq64::gui_drawingarea_gtk2::m_window_x [protected]

Could make this constant, but some windows are resizable. Window width value.

10.20.4.10 m_window_y

int seq64::gui_drawingarea_gtk2::m_window_y [protected]

10.20.4.11 m_current_x

int seq64::gui_drawingarea_gtk2::m_current_x [protected]

Current mouse x value.

10.20.4.12 m_current_y

int seq64::gui_drawingarea_gtk2::m_current_y [protected]

Public Member Functions

- [gui_palette_gtk2](#) ()
Principal constructor.
- [~gui_palette_gtk2](#) ()
Provides a destructor to delete allocated objects.
- void [initialize](#) ()
Adds all of the main palette colors in the PaletteColor enumeration into the palette contain.
- const [Color](#) & [get_color](#) (PaletteColor index) const
- [Color](#) [get_color_ex](#) (PaletteColor index, double h, double s=0.65, double v=1.0) const
Gets a color, but returns a modified value via the function Gdk::Color::set_hsv(h, s, v).
- const [Color](#) & [line_color](#) () const
'Getter' function for member m_line_color Provides an experimental way to change some line colors from black to something else.
- const [Color](#) & [progress_color](#) () const
'Getter' function for member m_progress_color Provides an experimental way to change the progress line color from black to something else.
- const [Color](#) & [black](#) () const
'Getter' function for member m_black Although these color getters return static values (if so compiled), these colors are used only in the window and drawing-area classes, so no need to make these functions static.
- const [Color](#) & [dark_red](#) () const
'Getter' function for member m_dk_red
- const [Color](#) & [dark_green](#) () const
'Getter' function for member m_dk_green
- const [Color](#) & [dark_orange](#) () const
'Getter' function for member m_dk_orange
- const [Color](#) & [dark_blue](#) () const
'Getter' function for member m_dk_blue
- const [Color](#) & [dark_magenta](#) () const
'Getter' function for member m_dk_magenta
- const [Color](#) & [dark_cyan](#) () const
'Getter' function for member m_dk_cyan
- const [Color](#) & [white](#) () const
'Getter' function for member m_white
- const [Color](#) & [grey_paint](#) () const
'Getter' function for member m_grey_paint
- const [Color](#) & [dark_grey_paint](#) () const
'Getter' function for member m_dk_grey_paint
- const [Color](#) & [light_grey_paint](#) () const
'Getter' function for member m_lt_grey_paint
- const [Color](#) & [red](#) () const
'Getter' function for member m_red
- const [Color](#) & [orange](#) () const
'Getter' function for member m_orange
- const [Color](#) & [yellow](#) () const
'Getter' function for member m_yellow
- const [Color](#) & [green](#) () const
'Getter' function for member m_green
- const [Color](#) & [magenta](#) () const
'Getter' function for member m_magenta
- const [Color](#) & [blue](#) () const
'Getter' function for member m_blue

- const [Color](#) & [black_paint](#) () const
'Getter' function for member m_blk_paint
- const [Color](#) & [white_paint](#) () const
'Getter' function for member m_wht_paint
- const [Color](#) & [black_key_paint](#) () const
'Getter' function for member m_blk_key_paint
- const [Color](#) & [white_key_paint](#) () const
'Getter' function for member m_wht_key_paint
- const [Color](#) & [tempo_paint](#) () const
'Getter' function for member m_tempo_paint
- const [Color](#) & [sel_paint](#) () const
'Getter' function for member m_sel_paint
- const [Color](#) & [bg_color](#) () const
'Getter' function for member m_bg_color
- void [bg_color](#) (const [Color](#) &c)
'Setter' function for member m_bg_color
- const [Color](#) & [fg_color](#) () const
'Getter' function for member m_fg_color
- void [fg_color](#) (const [Color](#) &c)
'Setter' function for member m_fg_color

Static Public Member Functions

- static void [load_inverse_palette](#) (bool inverse=true)
Provides an alternate color palette, somewhat constrained by the colors in the font bitmaps.
- static bool [is_inverse](#) ()
Indicates if the inverse color palette is loaded.

Protected Types

- typedef Gdk::Color [Color](#)
Provides a type for the color object.

Protected Attributes

- palette< [Color](#) > [m_palette](#)

Private Attributes

- [Color m_line_color](#)
Provides the line color.
- [Color m_progress_color](#)
Provides the progress bar color.
- [Color m_bg_color](#)
The current background color.
- [Color m_fg_color](#)
The current foreground color.

Static Private Attributes

- static bool [m_is_inverse](#)
Flags the presense of the inverse color palette.
- static const [Color m_black](#)
Provides the black color.
- static const [Color m_red](#)
Provides the red color.
- static const [Color m_green](#)
Provides the green color.
- static const [Color m_yellow](#)
Provides the yellow color.
- static const [Color m_blue](#)
Provides the blue color.
- static const [Color m_magenta](#)
Provides the magenta color.
- static const [Color m_cyan](#)
Provides the cyan color.
- static const [Color m_white](#)
Provides the white color.
- static const [Color m_dk_black](#)
Provides a blood-red color.
- static const [Color m_dk_red](#)
Provides a blood-red color.
- static const [Color m_dk_green](#)
Provides a dark green color.
- static const [Color m_dk_yellow](#)
Provides a dark green color.
- static const [Color m_dk_blue](#)
Provides the dark blue color.
- static const [Color m_dk_magenta](#)
Provides a dark magenta color.
- static const [Color m_dk_cyan](#)
Provides the dark cyan color.
- static const [Color m_dk_white](#)
Provides a greyish color.
- static const [Color m_orange](#)
Provides the orange color.
- static const [Color m_pink](#)
Provides the pink color.
- static const [Color m_grey](#)
Provides the unvarying grey color.
- static const [Color m_dk_orange](#)
Provides a dark orange color.
- static const [Color m_dk_pink](#)
Provides a dark pink color.
- static const [Color m_dk_grey](#)
The unvarying dark grey color.
- static [Color m_grey_paint](#)
Provides the grey color.
- static [Color m_dk_grey_paint](#)

- Provides the dark grey color.*
 - static [Color m_lt_grey_paint](#)
- Provides the light grey color.*
 - static [Color m_blk_paint](#)
- An invertible black color.*
 - static [Color m_wht_paint](#)
- An invertible white color.*
 - static [Color m_blk_key_paint](#)
- Provides the color of a black key.*
 - static [Color m_wht_key_paint](#)
- Provides the color of a white key.*
 - static [Color m_tempo_paint](#)
- The color of a tempo line.*
 - static [Color m_sel_paint](#)
- The color of a selection box.*

10.21.1 Detailed Description

Note that this class must be derived from `Gtk::DrawingArea` (or `Gtk::Widget`) in order to get access to the `get_↔ default_colormap()` function used in the constructor.

10.21.2 Member Typedef Documentation

10.21.2.1 Color

```
typedef Gdk::Color seq64::gui_palette_gtk2::Color [protected]
```

The following uses are made of each color:

- Black. The background color of armed patterns. The color of most lines in the user interface, including the main grid lines. The default color of progress lines and text.
- White. The default background color of just about everything drawn in the application.
- Grey. The color of minor grid lines and the markers for the currently-selected scale.
- Dark grey. The color of some grid lines, and the background of a queued pattern slot.
- Light grey. The color of some grid lines.
- Red. The optional color of progress bars.
- Orange. The fill-in color for selected notes and events.
- Dark orange. The color of selected event data lines and the color of the selection box for events to be pasted.
- Yellow. The background of the pattern and name slots for empty patterns. The text color for selected empty pattern slots.
- Green. Not yet used.

- Blue. Not yet used.
- Dark cyan. The background color of muted patterns currently in edit, or the pattern that contains the original data for an imported SMF 0 song. The text color of an unmuted pattern currently in edit. These colors apply to the pattern editor and the song editor. The color of the selected background pattern in the song editor.
- Line color. The generic line color, meant for expansion. Currently black.
- Progress color. The progress line color. Black by default, but can be set to red.
- Background color. The currently-in-use background color. Can vary a lot when a pixmap is being redrawn.
- Foreground color. The currently-in-use foreground color. Can vary a lot when a pixmap is being redrawn.

10.21.3 Constructor & Destructor Documentation

10.21.3.1 gui_palette_gtk2()

```
seq64::gui_palette_gtk2::gui_palette_gtk2 ( )
```

In the constructor one can only allocate colors; get_window() returns 0 because this window has not yet been realized. Also note that the possible color names that can be used are found in /usr/share/X11/rgb.txt.

Todo Use an array of colors instead of this switch.

10.21.3.2 ~gui_palette_gtk2()

```
seq64::gui_palette_gtk2::~~gui_palette_gtk2 ( )
```

10.21.4 Member Function Documentation

10.21.4.1 initialize()

```
void seq64::gui_palette_gtk2::initialize ( )
```

The palette is meant to be used to color sequences differently, though this feature is not yet supported in the Gtkmm-2.4 version of Sequencer64.

10.21.4.2 get_color()

```
const Color& seq64::gui_palette_gtk2::get_color (
    PaletteColor index ) const [inline]
```

10.21.4.3 `get_color_ex()`

```
gui_palette_gtk2::Color seq64::gui_palette_gtk2::get_color_ex (
    PaletteColor index,
    double h,
    double s = 0.65,
    double v = 1.0 ) const
```

This function sets the color, by specifying hue, saturation, and value (brightness).

Parameters

<i>h</i>	Hue, in the range 0..360 degrees. Ignored if set to -1.0.
<i>s</i>	Saturation, in the range 0..1, defaults to 0.65.
<i>v</i>	Value (a.k.a. brightness), in the range 0..1, defaults to 1.0.

10.21.4.4 load_inverse_palette()

```
void seq64::gui_palette_gtk2::load_inverse_palette (
    bool inverse = true ) [static]
```

Inverse is not a complete inverse. It is more like a "night" mode. However, there are still some bright colors even in this mode. Some colors, such as the selection color (orange) are the same in either mode.

Parameters

<i>inverse</i>	If true, load the alternate palette. Otherwise, load the default palette.
----------------	---

10.21.4.5 is_inverse()

```
static bool seq64::gui_palette_gtk2::is_inverse ( ) [inline], [static]
```

10.21.4.6 line_color()

```
const Color& seq64::gui_palette_gtk2::line_color ( ) const [inline]
```

Might eventually be selectable from the "user" configuration file

10.21.4.7 progress_color()

```
const Color& seq64::gui_palette_gtk2::progress_color ( ) const [inline]
```

Now selectable from the "user" configuration file.

10.21.4.8 black()

```
const Color& seq64::gui_palette_gtk2::black ( ) const [inline]
```

10.21.4.9 dark_red()

```
const Color& seq64::gui_palette_gtk2::dark_red ( ) const [inline]
```

10.21.4.10 dark_green()

```
const Color& seq64::gui_palette_gtk2::dark_green ( ) const [inline]
```

10.21.4.11 dark_orange()

```
const Color& seq64::gui_palette_gtk2::dark_orange ( ) const [inline]
```

10.21.4.12 dark_blue()

```
const Color& seq64::gui_palette_gtk2::dark_blue ( ) const [inline]
```

10.21.4.13 dark_magenta()

```
const Color& seq64::gui_palette_gtk2::dark_magenta ( ) const [inline]
```

10.21.4.14 dark_cyan()

```
const Color& seq64::gui_palette_gtk2::dark_cyan ( ) const [inline]
```

10.21.4.15 white()

```
const Color& seq64::gui_palette_gtk2::white ( ) const [inline]
```

10.21.4.16 grey_paint()

```
const Color& seq64::gui_palette_gtk2::grey_paint ( ) const [inline]
```

10.21.4.17 dark_grey_paint()

```
const Color& seq64::gui_palette_gtk2::dark_grey_paint ( ) const [inline]
```

10.21.4.18 light_grey_paint()

```
const Color& seq64::gui_palette_gtk2::light_grey_paint ( ) const [inline]
```

10.21.4.19 red()

```
const Color& seq64::gui_palette_gtk2::red ( ) const [inline]
```

10.21.4.20 orange()

```
const Color& seq64::gui_palette_gtk2::orange ( ) const [inline]
```

10.21.4.21 yellow()

```
const Color& seq64::gui_palette_gtk2::yellow ( ) const [inline]
```

10.21.4.22 green()

```
const Color& seq64::gui_palette_gtk2::green ( ) const [inline]
```

10.21.4.23 magenta()

```
const Color& seq64::gui_palette_gtk2::magenta ( ) const [inline]
```

10.21.4.24 blue()

```
const Color& seq64::gui_palette_gtk2::blue ( ) const [inline]
```

10.21.4.25 black_paint()

```
const Color& seq64::gui_palette_gtk2::black_paint ( ) const [inline]
```

10.21.4.26 white_paint()

```
const Color& seq64::gui_palette_gtk2::white_paint ( ) const [inline]
```

10.21.4.27 black_key_paint()

```
const Color& seq64::gui_palette_gtk2::black_key_paint ( ) const [inline]
```

10.21.4.28 white_key_paint()

```
const Color& seq64::gui_palette_gtk2::white_key_paint ( ) const [inline]
```

10.21.4.29 tempo_paint()

```
const Color& seq64::gui_palette_gtk2::tempo_paint ( ) const [inline]
```

10.21.4.30 sel_paint()

```
const Color& seq64::gui_palette_gtk2::sel_paint ( ) const [inline]
```

10.21.4.31 bg_color() [1/2]

```
const Color& seq64::gui_palette_gtk2::bg_color ( ) const [inline]
```

10.21.4.32 bg_color() [2/2]

```
void seq64::gui_palette_gtk2::bg_color (
    const Color & c ) [inline]
```

10.21.4.33 fg_color() [1/2]

```
const Color& seq64::gui_palette_gtk2::fg_color ( ) const [inline]
```

10.21.4.34 fg_color() [2/2]

```
void seq64::gui_palette_gtk2::fg_color (
    const Color & c ) [inline]
```

10.21.5 Field Documentation

10.21.5.1 m_palette

```
palette<Color> seq64::gui_palette_gtk2::m_palette [protected]
```

10.21.5.2 m_is_inverse

```
bool seq64::gui_palette_gtk2::m_is_inverse [static], [private]
```

By default, the inverse color palette is not loaded.

10.21.5.3 m_black

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_black [static], [private]
```

10.21.5.4 m_red

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_red [static], [private]
```

10.21.5.5 m_green

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_green [static], [private]
```

10.21.5.6 m_yellow

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_yellow [static], [private]
```

10.21.5.7 m_blue

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_blue [static], [private]
```

10.21.5.8 m_magenta

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_magenta [static], [private]
```

10.21.5.9 m_cyan

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_cyan [static], [private]
```

10.21.5.10 m_white

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_white [static], [private]
```

10.21.5.11 m_dk_black

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_black [static], [private]
```

10.21.5.12 m_dk_red

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_red [static], [private]
```

10.21.5.13 m_dk_green

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_green [static], [private]
```


10.21.5.14 m_dk_yellow

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_yellow [static], [private]
```

10.21.5.15 m_dk_blue

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_blue [static], [private]
```

10.21.5.16 m_dk_magenta

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_magenta [static], [private]
```

10.21.5.17 m_dk_cyan

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_cyan [static], [private]
```

10.21.5.18 m_dk_white

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_white [static], [private]
```

10.21.5.19 m_orange

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_orange [static], [private]
```

10.21.5.20 m_pink

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_pink [static], [private]
```

10.21.5.21 m_grey

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_grey [static], [private]
```

10.21.5.22 m_dk_orange

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_orange [static], [private]
```

10.21.5.23 m_dk_pink

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_pink [static], [private]
```

10.21.5.24 m_dk_grey

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_grey [static], [private]
```

10.21.5.25 m_grey_paint

```
STATIC_COLOR seq64::gui_palette_gtk2::m_grey_paint [static], [private]
```

10.21.5.26 m_dk_grey_paint

```
STATIC_COLOR seq64::gui_palette_gtk2::m_dk_grey_paint [static], [private]
```

10.21.5.27 m_lt_grey_paint

```
STATIC_COLOR seq64::gui_palette_gtk2::m_lt_grey_paint [static], [private]
```

10.21.5.28 m_blk_paint

```
STATIC_COLOR seq64::gui_palette_gtk2::m_blk_paint [static], [private]
```

10.21.5.29 m_wht_paint

```
STATIC_COLOR seq64::gui_palette_gtk2::m_wht_paint [static], [private]
```

10.21.5.30 m_blk_key_paint

STATIC_COLOR seq64::gui_palette_gtk2::m_blk_key_paint [static], [private]

10.21.5.31 m_wht_key_paint

STATIC_COLOR seq64::gui_palette_gtk2::m_wht_key_paint [static], [private]

10.21.5.32 m_tempo_paint

STATIC_COLOR seq64::gui_palette_gtk2::m_tempo_paint [static], [private]

10.21.5.33 m_sel_paint

STATIC_COLOR seq64::gui_palette_gtk2::m_sel_paint [static], [private]

10.21.5.34 m_line_color

Color seq64::gui_palette_gtk2::m_line_color [private]

10.21.5.35 m_progress_color

Color seq64::gui_palette_gtk2::m_progress_color [private]

10.21.5.36 m_bg_color

Color seq64::gui_palette_gtk2::m_bg_color [private]

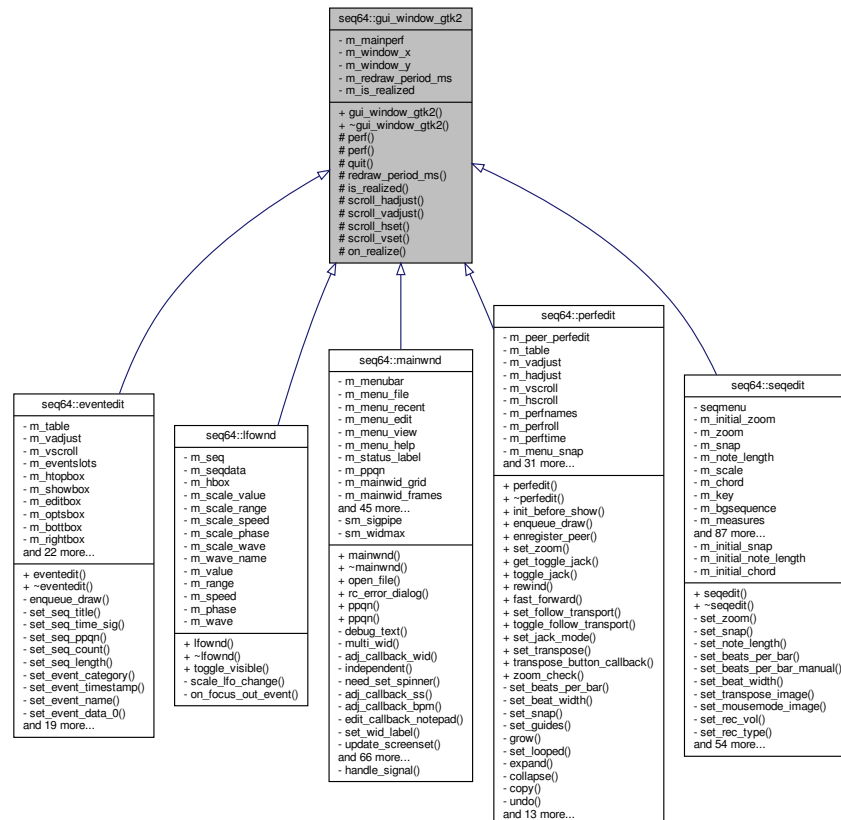
10.21.5.37 m_fg_color

Color seq64::gui_palette_gtk2::m_fg_color [private]

10.22 seq64::gui_window_gtk2 Class Reference

This class supports a basic interface for Gtk::Window-derived objects.

Inheritance diagram for seq64::gui_window_gtk2:



Public Member Functions

- [gui_window_gtk2](#) ([perform](#) &p, int window_x=0, int window_y=0)
Principal constructor, has a reference to the all-important perform object.
- virtual [~gui_window_gtk2](#) ()
This rote constructor does nothing.

Protected Member Functions

- [perform](#) & [perf](#) ()
'Getter' function for member m_mainperf
- const [perform](#) & [perf](#) () const
'Getter' function for member m_mainperf, const version
- virtual void [quit](#) ()
Provides "quit" functionality that WE HAVE OVERLOOKED!!! At some point we need to rectify this situation, probably for the sake of session support.
- int [redraw_period_ms](#) () const

- *'Getter' function for member m_redraw_period_ms*
- bool `is_realized` () const
 - *'Getter' function for member m_is_realized*
- void `scroll_hadjust` (Gtk::Adjustment &hadjust, double step)
 - *This function provides optimization for the on_scroll_event() functions, and should provide support for having the seqedit/seqroll/seqtime/seqdata panes follow the scrollbar, in a future upgrade.*
- void `scroll_vadjust` (Gtk::Adjustment &vadjust, double step)
 - *This function is the vertical version of scroll_hadjust().*
- void `scroll_hset` (Gtk::Adjustment &hadjust, double value)
 - *This function is the horizontal scroll setter.*
- void `scroll_vset` (Gtk::Adjustment &vadjust, double value)
 - *This function is the vertical scroll setter.*
- void `on_realize` ()
 - *This callback function calls the base-class on_realize() function, and sets the m_is_realized flag.*

Private Attributes

- `perform` & `m_mainperf`
 - *The master object, sort of a sequence buss for all of the sequence.*
- int `m_window_x`
 - *Window sizes.*
- int `m_window_y`
 - *The height of the window.*
- int `m_redraw_period_ms`
 - *Provides the timer period for the eventedit timer, used to determine the rate of redrawing.*
- bool `m_is_realized`
 - *Indicates if on_realize() has been called.*

10.22.1 Constructor & Destructor Documentation

10.22.1.1 gui_window_gtk2()

```
seq64::gui_window_gtk2::gui_window_gtk2 (
    perform & p,
    int window_x = 0,
    int window_y = 0 )
```

Note

We've collected the redraw timeouts into a base-class member. Most were valued at c_redraw_ms (40 ms), but mainwnd used 25 ms, so beware. We will eventually make this a user-interface parameter.

Parameters

<i>p</i>	Refers to the main performance object.
<i>window_x</i>	The width of the window.
<i>window_y</i>	The height of the window.
Generated by Doxygen	

10.22.1.2 ~gui_window_gtk2()

```
seq64::gui_window_gtk2::~~gui_window_gtk2 ( ) [virtual]
```

10.22.2 Member Function Documentation

10.22.2.1 perf() [1/2]

```
perform& seq64::gui_window_gtk2::perf ( ) [inline], [protected]
```

10.22.2.2 perf() [2/2]

```
const perform& seq64::gui_window_gtk2::perf ( ) const [inline], [protected]
```

10.22.2.3 quit()

```
virtual void seq64::gui_window_gtk2::quit ( ) [inline], [protected], [virtual]
```

10.22.2.4 redraw_period_ms()

```
int seq64::gui_window_gtk2::redraw_period_ms ( ) const [inline], [protected]
```

10.22.2.5 is_realized()

```
bool seq64::gui_window_gtk2::is_realized ( ) const [inline], [protected]
```

10.22.2.6 scroll_hadjust()

```
void seq64::gui_window_gtk2::scroll_hadjust (
    Gtk::Adjustment & hadjust,
    double step ) [protected]
```

This function is currently duplicated in the [gui_drawingarea_gtk2](#) and [gui_window_gtk2](#) modules.

Parameters

<i>hadjust</i>	Provides a reference to the adjustment object to be adjusted.
<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. If negative, the adjustment is leftward. If positive, the adjustment is rightward. It can be the value of <code>m_hadjust->get_step_increment()</code> , or provided especially to keep up with the progress bar.

10.22.2.7 scroll_vadjust()

```
void seq64::gui_window_gtk2::scroll_vadjust (
    Gtk::Adjustment & vadjust,
    double step ) [protected]
```

Parameters

<i>vadjust</i>	Provides a reference to the adjustment object to be adjusted.
<i>step</i>	Provides the step value to use for adjusting the vertical scrollbar. If greater than 0, the movement is downward. If less than zero, the movement is upward.

10.22.2.8 scroll_hset()

```
void seq64::gui_window_gtk2::scroll_hset (
    Gtk::Adjustment & hadjust,
    double value ) [protected]
```

Parameters

<i>hadjust</i>	Provides a reference to the adjustment object to be set. It is clamped as necessary.
<i>value</i>	Provides the value to use for setting the horizontal scrollbar.

10.22.2.9 scroll_vset()

```
void seq64::gui_window_gtk2::scroll_vset (
    Gtk::Adjustment & vadjust,
    double value ) [protected]
```

Parameters

<i>vadjust</i>	Provides a reference to the vertical adjustment object to be set. It is clamped as necessary.
<i>value</i>	Provides the value to use for setting the vertical scrollbar.

10.22.2.10 on_realize()

```
void seq64::gui_window_gtk2::on_realize ( ) [protected]
```

10.22.3 Field Documentation

10.22.3.1 m_mainperf

```
perform& seq64::gui_window_gtk2::m_mainperf [private]
```

And a whole lot more than that.

10.22.3.2 m_window_x

```
int seq64::gui_window_gtk2::m_window_x [private]
```

Could make this constant, but some windows are resizable. The width of the window.

10.22.3.3 m_window_y

```
int seq64::gui_window_gtk2::m_window_y [private]
```

10.22.3.4 m_redraw_period_ms

```
int seq64::gui_window_gtk2::m_redraw_period_ms [private]
```

This is currently hardwired to 40 ms in Linux, and 20 ms in Windows. Note that mainwnd used 25 ms.

10.22.3.5 m_is_realized

```
bool seq64::gui_window_gtk2::m_is_realized [private]
```

In some cases, we don't want to draw in objects that haven't yet appeared, otherwise crashes occur.

10.23 seq64::jack_assistant Class Reference

This class provides the performance mode JACK support.

Public Member Functions

- [jack_assistant](#) ([perform](#) &[parent](#), [midibpm](#) bpmminute=SEQ64_DEFAULT_BPM, int ppqn=SEQ64_USE_DEFAULT_PPQN, int bpm=SEQ64_DEFAULT_BEATS_PER_MEASURE, int beatwidth=SEQ64_DEFAULT_BEAT_WIDTH)
- This constructor initializes a number of member variables, some of them public!*
- [~jack_assistant](#) ()
- The destructor doesn't need to do anything yet.*
- [perform](#) & [parent](#) ()
- 'Getter' function for member m_jack_parent Needed for external callbacks.*
- const [perform](#) & [parent](#) () const
- 'Getter' function for member m_jack_parent, const version*
- bool [is_running](#) () const
- 'Getter' function for member m_jack_running*
- bool [is_master](#) () const
- 'Getter' function for member m_jack_master*
- int [get_ppqn](#) () const
- 'Getter' function for member m_ppqn*
- int [get_beat_width](#) () const
- 'Getter' function for member m_beat_width*
- void [set_beat_width](#) (int bw)
- 'Setter' function for member m_beat_width*
- int [get_beats_per_measure](#) () const
- 'Getter' function for member m_beats_per_measure*
- void [set_beats_per_measure](#) (int bpm)
- 'Setter' function for member m_beats_per_measure*
- [midibpm](#) [get_beats_per_minute](#) () const
- 'Getter' function for member m_beats_per_minute*
- void [set_beats_per_minute](#) ([midibpm](#) bpmminute)
- 'Setter' function for member m_beats_per_minute For the future, changing the BPM (beats/minute) internally.*
- [jack_transport_state_t](#) [transport_state](#) () const
- 'Getter' function for member m_jack_transport_state*
- bool [transport_not_starting](#) () const
- Returns true if the JACK transport state is not JackTransportStarting.*
- bool [init](#) ()
- Initializes JACK support.*
- bool [deinit](#) ()
- Tears down the JACK infrastructure.*
- bool [session_event](#) ()
- Writes the MIDI file named "<jack session dir>-file.mid" using a midifile object, quits if told to by JACK, and can free the JACK session event.*
- bool [activate](#) ()
- Activate JACK here.*
- void [start](#) ()
- If JACK is supported, starts the JACK transport.*
- void [stop](#) ()
- If JACK is supported, stops the JACK transport.*
- void [position](#) (bool state, [midipulse](#) tick=0)
- If JACK is supported and running, sets the position of the transport to the new frame number, frame 0.*
- bool [output](#) ([jack_scratchpad](#) &pad)
- Performance output function for JACK, called by the perform function of the same name.*

- void [set_ppqn](#) (int ppqn)
'Setter' function for member m_ppqn For the future, changing the PPQN internally.
- double [get_jack_tick](#) () const
'Getter' function for member m_jack_tick
- const jack_position_t & [get_jack_pos](#) () const
'Getter' function for member m_jack_pos
- void [toggle_jack_mode](#) ()
'Setter' function for member m_toggle_jack
- void [set_jack_mode](#) (bool mode)
'Setter' function for member m_toggle_jack
- bool [get_jack_mode](#) () const
'Getter' function for member m_toggle_jack Seems misnamed.
- [midipulse](#) [get_jack_stop_tick](#) () const
'Getter' function for member m_jack_stop_tick
- void [set_jack_stop_tick](#) (long tick)
'Setter' function for member m_jack_stop_tick
- jack_nframes_t [jack_frame_rate](#) () const
'Getter' function for member m_jack_frame_rate
- bool [get_follow_transport](#) () const
'Getter' function for member m_follow_transport
- void [set_follow_transport](#) (bool aset)
'Setter' function for member m_follow_transport
- void [toggle_follow_transport](#) ()
'Setter' function for member m_follow_transport
- bool [toggle_song_start_mode](#) ()
'Setter' function for member [parent\(\)](#).toggle_song_start_mode()
- bool [song_start_mode](#) () const
'Getter' function for member [parent\(\)](#).song_start_mode()
- void [set_start_from_perfedit](#) (bool start)
'Setter' function for member [parent\(\)](#).start_from_perfedit()
- jack_client_t * [client](#) () const
- const std::string & [client_name](#) () const
'Getter' function for member m_jack_client_name
- const std::string & [client_uuid](#) () const
'Getter' function for member m_jack_client_uuid

Static Public Member Functions

- static void [show_position](#) (const jack_position_t &pos)
Shows a one-line summary of a JACK position structure.
- static std::string [get_state_name](#) (const jack_transport_state_t &state)
Converts a JACK transport value to a human-readable string.

Private Member Functions

- void [set_jack_running](#) (bool flag)
'Setter' function for member m_jack_running
- double [tick_multiplier](#) () const
Convenience function for internal use.
- jack_client_t * [client_open](#) (const std::string &clientname)
A member wrapper function for the new free function [create_jack_client\(\)](#).
- void [get_jack_client_info](#) ()
Tries to obtain the best information on the JACK client and the UUID assigned to this client.
- int [sync](#) (jack_transport_state_t state=(jack_transport_state_t)(-1))
A helper function for syncing up with JACK parameters.

Static Private Member Functions

- static bool [info_message](#) (const std::string &msg)
Common-code for console messages.
- static bool [error_message](#) (const std::string &msg)
Common-code for error messages.

Private Attributes

- [perform](#) & [m_jack_parent](#)
Provides the perform object that needs this JACK assistant/scratchpad class.
- jack_client_t * [m_jack_client](#)
Provides a handle into JACK, so that the application, as a JACK client, can issue commands and retrieve status information from JACK.
- std::string [m_jack_client_name](#)
A new member to hold the actual name of the client assigned by JACK.
- std::string [m_jack_client_uuid](#)
A new member to hold the actual UUID of the client assigned by JACK.
- jack_nframes_t [m_jack_frame_current](#)
Holds the current frame number obtained from JACK transport, via a call to [jack_get_current_transport_frame\(\)](#).
- jack_nframes_t [m_jack_frame_last](#)
Holds the last frame number we got from JACK, so that progress can be tracked.
- jack_position_t [m_jack_pos](#)
Provides positioning information on JACK playback.
- jack_transport_state_t [m_jack_transport_state](#)
Holds the JACK transport state.
- jack_transport_state_t [m_jack_transport_state_last](#)
Holds the last JACK transport state.
- double [m_jack_tick](#)
The tick/pulse value derived from the current frame number, the ticks/beat value, the beats/minute value, and the frame rate.
- jack_session_event_t * [m_session_ev](#)
Provides a kind of handle to the JACK session manager.
- bool [m_jack_running](#)
Indicates if JACK Sync has been enabled successfully.
- bool [m_jack_master](#)
Indicates if JACK Sync has been enabled successfully, with the application running as JACK Master.

- `jack_nframes_t m_jack_frame_rate`
Holds the current frame rate.
- `bool m_toggle_jack`
Ostensibly a toggle, the functions that access this member are called "jack_mode" functions.
- `midipulse m_jack_stop_tick`
Used in `jack_process_callback()` to reposition when JACK transport is not rolling or starting.
- `bool m_follow_transport`
TBD.
- `int m_ppqn`
Holds the global PPQN value for the Sequencer64 session.
- `int m_beats_per_measure`
Holds the song's beats/measure value for using in setting JACK position.
- `int m_beat_width`
Holds the song's beat width value (denominator of the time signature) for using in setting JACK position.
- `midibpm m_beats_per_minute`
Holds the song's beats/minute (BPM) value for using in setting JACK position.

Static Private Attributes

- `static jack_status_pair_t sm_status_pairs []`
Pairs the JACK status bits with human-readable descriptions of each one.

Friends

- `int jack_transport_callback (jack_nframes_t nframes, void *arg)`
Implemented second patch for JACK Transport from `freddix/seq24` GitHub project.
- `void jack_shutdown_callback (void *arg)`
Global functions for JACK support and JACK sessions.
- `void jack_timebase_callback (jack_transport_state_t state, jack_nframes_t nframes, jack_position_t *pos, int new_pos, void *arg)`
The JACK timebase function defined here sets the JACK position structure.
- `long get_current_jack_position (void *arg)`
This function gets the current JACK position.
- `void jack_session_callback (jack_session_event_t *ev, void *arg)`
Set the `m_jsession_ev` (event) value of the perform object.

10.23.1 Constructor & Destructor Documentation

10.23.1.1 `jack_assistant()`

```
seq64::jack_assistant::jack_assistant (
    perform & parent,
    midibpm bpminute = SEQ64_DEFAULT_BPM,
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    int bpmeasure = SEQ64_DEFAULT_BEATS_PER_MEASURE,
    int beatwidth = SEQ64_DEFAULT_BEAT_WIDTH )
```

Note that the perform object currently calls `jack_assistant::init()`, but that call could be made here instead.

Parameters

<i>parent</i>	Provides a reference to the main perform object that needs to control JACK event.
<i>bpmminute</i>	The beats/minute to set up JACK to use (applies to Master setup).
<i>ppqn</i>	The parts-per-quarter-note setting in force for the present tune.
<i>bpmeasure</i>	The beats/measure (time signature numerator) in force for the present tune.
<i>beatwidth</i>	The beat-width (time signature denominator) in force for the present tune.

10.23.1.2 ~jack_assistant()

```
seq64::jack_assistant::~~jack_assistant ( )
```

The perform object currently calls [jack_assistant::deinit\(\)](#), but that call could be made here instead.

10.23.2 Member Function Documentation

10.23.2.1 show_position()

```
void seq64::jack_assistant::show_position (
    const jack_position_t & pos ) [static]
```

This function is meant for experimenting and learning.

The fields of this structure are as follows. Only the fields we care about are shown.

```
jack_nframes_t    frame_rate:    current frame rate (per second)
jack_nframes_t    frame:        frame number, always present
jack_position_bits_t valid:      which other fields are valid
```

JackPositionBBT:

```
int32_t           bar:          current bar
int32_t           beat:         current beat-within-bar
int32_t           tick:         current tick-within-beat
double            bar_start_tick
float             beats_per_bar: time signature "numerator"
float             beat_type:     time signature "denominator"
double            ticks_per_beat
double            beats_per_minute
```

JackBBTFrameOffset:

```
jack_nframes_t    bbt_offset;    frame offset for the BBT fields
```

Only the most "important" and time-varying fields are shown. The format output is brief and inscrutable unless you read this format example:

```

nnnnn frame B:B:T N/D TPB BPM BBT
^      ^      ^      ^      ^      ^      ^
|      |      |      |      |      |      |
|      |      |      |      |      |      |----- bbt_offset (frame), even if invalid
|      |      |      |      |      |      |----- beats_per_minute
|      |      |      |      |      |      |----- ticks_per_beat (PPQN * 10?)
|      |      |      |      |      |      |----- beat_type (denominator)
|      |      |      |      |      |      |----- beats_per_bar (numerator)
|      |      |      |      |      |      |----- bar : beat : tick
|      |      |      |      |      |      |----- frame (number)
|----- the "valid" bits

```

The "valid" field is shown as bits in the same bit order as shown here, but represented as a five-character string, "nnnnn", n = 0 or 1:

```

JackVideoFrameOffset = 0x100
JackAudioVideoRatio  = 0x080
JackBBTFrameOffset   = 0x040
JackPositionTimecode = 0x020
JackPositionBBT      = 0x010

```

We care most about nnnnn = "00101" in our experiments (the most common output will be "00001"). And we don't worry about non-integer measurements... we truncate them to integers. Change the output format if you want to play with non-Western timings.

Parameters

<i>pos</i>	The JACK position structure to dump.
------------	--------------------------------------

10.23.2.2 get_state_name()

```

std::string seq64::jack_assistant::get_state_name (
    const jack_transport_state_t & state ) [static]

```

Parameters

<i>state</i>	Provides the transport state value.
--------------	-------------------------------------

Returns

Returns the state name.

10.23.2.3 parent() [1/2]

```

perform& seq64::jack_assistant::parent ( ) [inline]

```

10.23.2.4 parent() [2/2]

```
const perform& seq64::jack_assistant::parent ( ) const [inline]
```

10.23.2.5 is_running()

```
bool seq64::jack_assistant::is_running ( ) const [inline]
```

10.23.2.6 is_master()

```
bool seq64::jack_assistant::is_master ( ) const [inline]
```

10.23.2.7 get_ppqn()

```
int seq64::jack_assistant::get_ppqn ( ) const [inline]
```

10.23.2.8 get_beat_width()

```
int seq64::jack_assistant::get_beat_width ( ) const [inline]
```

10.23.2.9 set_beat_width()

```
void seq64::jack_assistant::set_beat_width (
    int bw ) [inline]
```

Parameters

<i>bw</i>	Provides the beat-width (denominator of the time signature) value to set.
-----------	---

10.23.2.10 get_beats_per_measure()

```
int seq64::jack_assistant::get_beats_per_measure ( ) const [inline]
```

10.23.2.11 set_beats_per_measure()

```
void seq64::jack_assistant::set_beats_per_measure (
    int bpm ) [inline]
```

Parameters

<i>bpm</i>	Provides the beats/measure (numerator of the time signature) value to set.
------------	--

10.23.2.12 get_beats_per_minute()

```
midibpm seq64::jack_assistant::get_beats_per_minute ( ) const [inline]
```

10.23.2.13 set_beats_per_minute()

```
void seq64::jack_assistant::set_beats_per_minute (
    midibpm bpmminute )
```

We should consider adding validation. However, [perform::set_beats_per_minute\(\)](#) does validate already. Also, since [jack_transport_reposition\(\)](#) can be "called at any time by any client", we have removed the check for "is master". We do seem to see more "bad position structure" messages, though.

Parameters

<i>bpmminute</i>	Provides the beats/minute value to set.
------------------	---

10.23.2.14 transport_state()

```
jack_transport_state_t seq64::jack_assistant::transport_state ( ) const [inline]
```

10.23.2.15 transport_not_starting()

```
bool seq64::jack_assistant::transport_not_starting ( ) const [inline]
```


10.23.2.16 init()

```
bool seq64::jack_assistant::init ( )
```

Then we become a new client of the JACK server.

A sync callback is needed for polling of slow-sync clients. But seq24/sequencer64 are not slow-sync clients. We don't really need to be a slow-sync client, as far as we can tell. We can't get JACK working exactly the way it does in seq24 without the callback in place. Plus, it does things important to the setup of JACK. So now this setup is permanent.

Jack transport settings:

There are three settings: On, Master, and Master Conditional. Currently, they can all be selected in the user-interface's File / Options / JACK/LASH page. We really want only the proper combinations to be set, for clarity (the user-interface now takes care of this. We need to initialize if any of them are set, and the rc_settings::with_jack() function tells us that.

jack_set_process_callback() patch:

Implemented first patch from freddix/seq24 GitHub project, to fix JACK transport. One line of code. Well, we added some error-checking. :-)
Found some old notes on the Web the this patch really only works (to prevent seq24 freeze) if seq24 is set as JACK Master, or if another client application, such as Qtractor, is running as JACK Master (and then seq24 will apparently follow it).

STAZED: The call to [jack_timebase_callback\(\)](#) to supply jack with BBT, etc. would occasionally fail when the *pos information had zero or some garbage in the pos.frame_rate variable. This would occur when there was a rapid change of frame position by another client... i.e. qjackctl. From the jack API:

"pos address of the position structure for the next cycle;
pos->frame will be its frame number. If new_pos is FALSE, this structure contains extended position information from the current cycle. If TRUE, it contains whatever was set by the requester. The timebase_callback's task is to update the extended information here."

The "If TRUE" line seems to be the issue. It seems that qjackctl does not always set pos.frame_rate so we get garbage and some strange BBT calculations that display in qjackctl. So we need to set it here and just use m_↔ jack_frame_rate for calculations instead of pos.frame_rate.

Returns

Returns true if JACK is now considered to be running (or if it was already running.)

10.23.2.17 deinit()

```
bool seq64::jack_assistant::deinit ( )
```

Todo Note that we still need a way to call jack_release_timebase() when the user turns off the "JACK Master" status of Sequencer64.

Returns

Returns the value of m_jack_running, which should be false.

10.23.2.18 session_event()

```
bool seq64::jack_assistant::session_event ( )
```

ca 2015-07-24 Just a note: The OMA (OpenMandrivaAssociation) patch was already applied to seq24 v.0.9.2. It put quotes around the `-file` argument. However, the `-file` option doesn't work, so let's change that line.

```
sequencer64 --file \"${SESSION_DIR}file.mid\" --jack_session_uuid
```

Why are we using a `Glib::ustring` here? Convenience. But with C++11, we could use a `lexical_cast<>`. No more `ustring`, baby! It doesn't really matter; this function can call `Gtk::Main::quit()`, via the parent's `gui().quit()` function.

Returns

Always returns false.

10.23.2.19 activate()

```
bool seq64::jack_assistant::activate ( )
```

This function is called by [perform::activate\(\)](#) after the master bus is activated successfully.

Returns

Returns true if the `m_jack_client` pointer is null, which means only that we're not running JACK. Also returns true if the pointer exists and the `jack_active()` call succeeds.

Side-effect(s) The `m_jack_running` and `m_jack_master` flags are falsified in `jack_activate()` fails.

10.23.2.20 start()

```
void seq64::jack_assistant::start ( )
```

This function assumes that `m_jack_client` is not null, if `m_jack_running` is true.

10.23.2.21 stop()

```
void seq64::jack_assistant::stop ( )
```

This function assumes that `m_jack_client` is not null, if `m_jack_running` is true.

10.23.2.22 position()

```
void seq64::jack_assistant::position (
    bool songmode,
    midipulse tick = 0 )
```

This new position takes effect in two process cycles. If there are slow-sync clients and the transport is already rolling, it will enter the JackTransportStarting state and begin invoking their sync_callbacks until ready. This function is realtime-safe.

<http://jackaudio.org/files/docs/html/transport-design.html>

This `position()` function is called via `perform::position_jack()` in the mainwnd, perfedit, perfroll, and seqroll graphical user-interface support objects.

The code that was disabled sets the current tick to 0 or, if state was true, to the leftmost tick (which is probably the position of the L marker). The current tick is then converted to a frame number, and then we locate the transport to that position. We're going to enable this code, but make it dependent on a new boolean parameter that defaults to false, in anticipation of trying it out later.

Stazed:

```
The jack_frame calculation is all that is needed to change JACK
position. The BBT calculation can be sent, but will be overridden by the
first call to jack_timebase_callback() of any Master set. If no Master
is set, then the BBT will display the new position but will not change
it, even if the transport is rolling. There is no need to send BBT on
position change -- the fact that jack_transport_locate() exists and only
uses the frame position is proof that BBT is not needed! Upon further
reflection, why not send BBT? Because other programs do not... let's
follow convention. The calculation for jack_transport_locate(), works,
is simpler, and does not send BBT. The calculation for
jack_transport_reposition() will be commented out again.
jack_BBT_position() is not necessary to change jack position!
```

Note that there are potentially a couple of divide-by-zero opportunities in this function.

Helgrind complains about a possible data race involving `jack_transport_locate()` when starting playing.

Parameters

<i>songmode</i>	True if the caller wants to position while in Song mode.
-----------------	--

Alternate parameter to `_left_tick` (non-seq32 version):

```
If true, the current tick is set to the leftmost tick, instead of the
0th tick. Now used, but only if relocate is true. One question is,
do we want to perform this function if rc().with_jack_transport() is
true? Seems like we should be able to do it only if m_jack_master is
true.
```

Parameters

<i>tick</i>	If using Song mode for this call then this value is set as the "current tick" value. If it's value is bad (SEQ64_NULL_MIDIPULSE), then this parameter is set to 0 before being used.
-------------	--

10.23.2.23 output()

```
bool seq64::jack_assistant::output (
    jack_scratchpad & pad )
```

This code comes from [perform::output_func\(\)](#) from seq24.

Note

Follow up on this note found "out there": "Maybe I'm wrong but if I understood correctly, recent jack1 transport no longer goes into Jack_Transport_Starting state before going to Jack_Transport_Rolling (this was deliberately dropped), but seq24 currently needs this to start off with JACK transport." On the other hand, some people have no issues. This may have been due to the lack of m_jack_pos initialization.

Stazed:

Another note about JACK. If another JACK client supplies tempo/BBT different from seq42 (as Master), the perfroll grid will be incorrect. Perfroll uses internal temp/BBT and cannot update on the fly. Even if seq42 could support tempo/BBT changes, all info would have to be available before the transport start, to work. For this reason, the tempo/BBT info will be plugged from the seq42 internal settings here, always. This is the method used by probably all other JACK clients with some sort of time-line. The JACK API indicates that BBT is optional and AFIK, other sequencers only use frame & frame_rate from JACK for internal calculations. The tempo and BBT info is always internal. Also, if there is no Master set, then we would need to plug it here to follow the JACK frame anyways.

Parameters

<i>pad</i>	Provides a JACK scratchpad for sharing certain items between the perform object and the jack_assistant object.
------------	--

Returns

Returns true if JACK is running.

10.23.2.24 set_ppqn()

```
void seq64::jack_assistant::set_ppqn (
    int ppqn ) [inline]
```

We should consider adding validation. But it is used by perform.

Parameters

<i>ppqn</i>	Provides the PPQN value to set.
-------------	---------------------------------

10.23.2.25 get_jack_tick()

```
double seq64::jack_assistant::get_jack_tick ( ) const [inline]
```

10.23.2.26 get_jack_pos()

```
const jack_position_t& seq64::jack_assistant::get_jack_pos ( ) const [inline]
```

10.23.2.27 toggle_jack_mode()

```
void seq64::jack_assistant::toggle_jack_mode ( ) [inline]
```

10.23.2.28 set_jack_mode()

```
void seq64::jack_assistant::set_jack_mode (
    bool mode ) [inline]
```

10.23.2.29 get_jack_mode()

```
bool seq64::jack_assistant::get_jack_mode ( ) const [inline]
```

10.23.2.30 get_jack_stop_tick()

```
midipulse seq64::jack_assistant::get_jack_stop_tick ( ) const [inline]
```

10.23.2.31 set_jack_stop_tick()

```
void seq64::jack_assistant::set_jack_stop_tick (
    long tick ) [inline]
```

10.23.2.32 jack_frame_rate()

```
jack_nframes_t seq64::jack_assistant::jack_frame_rate ( ) const [inline]
```

10.23.2.33 get_follow_transport()

```
bool seq64::jack_assistant::get_follow_transport ( ) const [inline]
```

10.23.2.34 set_follow_transport()

```
void seq64::jack_assistant::set_follow_transport (
    bool aset ) [inline]
```

10.23.2.35 toggle_follow_transport()

```
void seq64::jack_assistant::toggle_follow_transport ( ) [inline]
```

10.23.2.36 toggle_song_start_mode()

```
bool seq64::jack_assistant::toggle_song_start_mode ( )
```

10.23.2.37 song_start_mode()

```
bool seq64::jack_assistant::song_start_mode ( ) const
```

10.23.2.38 set_start_from_perfeddit()

```
void seq64::jack_assistant::set_start_from_perfeddit (
    bool start )
```

10.23.2.39 client()

```
jack_client_t * seq64::jack_assistant::client ( ) const
```

10.23.2.40 client_name()

```
const std::string& seq64::jack_assistant::client_name ( ) const [inline]
```

10.23.2.41 client_uuid()

```
const std::string& seq64::jack_assistant::client_uuid ( ) const [inline]
```

10.23.2.42 set_jack_running()

```
void seq64::jack_assistant::set_jack_running (
    bool flag ) [inline], [private]
```

Parameters

<i>flag</i>	Provides the is-running value to set.
-------------	---------------------------------------

10.23.2.43 tick_multiplier()

```
double seq64::jack_assistant::tick_multiplier ( ) const [inline], [private]
```

Should we change 4.0 to a member value? What does it mean?

Returns

Returns the multiplier to convert a JACK tick value according to the PPQN, ticks/beat, and beat-type settings.

10.23.2.44 client_open()

```
jack_client_t * seq64::jack_assistant::client_open (
    const std::string & clientname ) [private]
```

Parameters

<i>clientname</i>	Provides the name of the client, used in the call to create_jack_client() . By default, this name is the macro SEQ64_PACKAGE (i.e. "sequencer64").
-------------------	--

Returns

Returns a pointer to the JACK client if JACK has opened the client connection successfully. Otherwise, a null pointer is returned.

10.23.2.45 `get_jack_client_info()`

```
void seq64::jack_assistant::get_jack_client_info ( ) [private]
```

Sets `m_jack_client_name` and `m_jack_client_info` as side-effects.

10.23.2.46 `sync()`

```
int seq64::jack_assistant::sync (
    jack_transport_state_t state = (jack_transport_state_t)(-1) ) [private]
```

Sequencer64 is not a slow-sync client (and Stazed support doesn't use it), so that callback is not really needed, but we probably need this sub-function here to start out with the right values for interacting with JACK.

Note the call to `jack_transport_query()`. This call is *not* in seq24, but seems to be needed in sequencer64 because we put `m_jack_pos` in the initializer list, which sets all its fields to 0. Seq24 accesses `m_jack_pos` before it ever gets set, but its fields have values. These values are bogus, but are consistent from run to run on my computer, and allow seq24 to follow another JACK Master, on some computers. It explains why people had different experiences with JACK sync.

If we explicitly call `jack_transport_query()` here, without changing the *state* parameter, then sequencer64 also can follow another JACK Master. (CURRENTLY BUGGY!)

Note that we should consider massaging the following `jack_position_t` members to set them to 0 (or 0.0) if less than 1.0 or 0.5:

```
- bar_start_tick
- ticks_per_beat
- beats_per_minute
- frame_time
- next_time
- audio_frames_per_video_frame
```

Also, why does `bbt_offset` start at 2128362496?

Parameters

<i>state</i>	The JACK transport state to be set.
--------------	-------------------------------------

10.23.2.47 info_message()

```
bool seq64::jack_assistant::info_message (
    const std::string & msg ) [static], [private]
```

Adds markers and a newline.

Parameters

<i>msg</i>	The message to print, sans the newline.
------------	---

Returns

Returns true.

10.23.2.48 error_message()

```
bool seq64::jack_assistant::error_message (
    const std::string & msg ) [static], [private]
```

Adds markers, and sets m_jack_running to false.

Parameters

<i>msg</i>	The message to print, sans the newline.
------------	---

Returns

Returns false for convenience/brevity in setting function return values.

10.23.3 Friends And Related Function Documentation

10.23.3.1 jack_transport_callback

```
int jack_transport_callback (
    jack_nframes_t nframes,
    void * arg ) [friend]
```

Added the following function. This function is supposed to allow seq24/sequencer64 to follow JACK transport.

For more advanced ideas, see the `MetronomeJack::process_callback()` function in the `klick` project. It plays a metronome tick after calculating if it needs to or not. (Maybe we could use it to provide our own tick for recording patterns.)

If debugging is enabled in the build, then this function outputs the current JACK position information every couple of seconds, which can be useful to examine the interactions with other JACK clients.

The code enabled via `USE_JACK_BBT_OFFSET` sets the JACK position field `bbt_offset` to 0. It doesn't seem to have any effect, though it can be seen when calling `show_position()` in the `jack_transport_callback()` function.

Stazed:

```
This process callback is called by JACK whether stopped or rolling.
Assuming every JACK cycle... "...client supplied function that is
called by the engine anytime there is work to be done". There seems to
be no definition of '...work to be done'. nframes = buffer_size -- is
not used.
```

Parameters

<i>nframes</i>	Unused.
<i>arg</i>	Used for debug output now. Note that this function will be called very often, and this pointer will currently not be used unless debugging is turned on.

Returns

Returns 0 on success, non-zero on error.

10.23.3.2 jack_shutdown_callback

```
void jack_shutdown_callback (
    void * arg ) [friend]
```

This callback is to shut down JACK by clearing the `jack_assistant :: m_jack_running` flag.

Parameters

<i>arg</i>	Points to the <code>jack_assistant</code> in charge of JACK support for the perform object.
------------	---

10.23.3.3 jack_timebase_callback

```
void jack_timebase_callback (
    jack_transport_state_t state,
    jack_nframes_t nframes,
    jack_position_t * pos,
    int new_pos,
    void * arg ) [friend]
```

The original version of the function worked properly with Hydrogen, but not with Klick. The new code seems to work with both. More testing and clarification is needed. This new code was "discovered" in the source-code for the "SooperLooper" project:

<http://essey.net/sooperlooper/>

The first difference with the new code is that it handles the case where the JACK position is moved (`new_pos == true`). If this is true, and the `JackPositionBBT` bit is off in `pos->valid`, then the new BBT value is set.

The second set of differences are in the "else" clause. In the new code, it is very simple: calculate the new tick value, back it off by the number of ticks in a beat, and perhaps go to the first beat of the next bar.

In the old code (complex!), the simple BBT adjustment is always made. This changes (perhaps) the `beats_per_bar`, `beat_type`, etc. We need to make these settings use the actual global values for beats set for `Sequencer64`. Then, if transitioning from `JackTransportStarting` to `JackTransportRolling` (instead of checking `new_pos!`), the BBT values (bar, beat, and tick) are finally adjusted. Here are the steps, with old and new steps noted:

```
-# Calculate the "delta" ticks based on the current frame, the
ticks_per_beat, the beats_per_minute, and the frame_rate. The old
code saves this in a local, the new code assigns it to pos->tick.
-# Old code: save this delta as a positive value.
-# Figure out the settings and modify bar, beat, tick, and
bar_start_tick. The old and new code seem to have the same intent,
but it seems like the new code is faster and also correct.
- Old code: Calculations are made by division and mod
operations.
- New code: Calculations are made by increments and decrements
in a while loop.
```

Stazed:

The call to `jack_timebase_callback()` to supply JACK with BBT, etc. would occasionally fail when the pos information had zero or some garbage in the `pos.frame_rate` variable. This would occur when there was a rapid change of frame position by another client... i.e. `qjackctl`. From the JACK API:

```
pos address of the position structure for the next cycle; pos->frame
will be its frame number. If new_pos is FALSE, this structure contains
extended position information from the current cycle. If TRUE, it
contains whatever was set by the requester. The timebase_callback's
task is to update the extended information here."
```

The "If TRUE" line seems to be the issue. It seems that `qjackctl` does not always set `pos.frame_rate` so we get garbage and some strange BBT calculations that display in `qjackctl`. So we need to set it here and just use `m_↵ jack_frame_rate` for calculations instead of `pos.frame_rate`.

Parameters

<i>state</i>	Indicates the current state of JACK transport.
<i>nframes</i>	The number of JACK frames in the current time period.
<i>pos</i>	Provides the position structure to be filled in, the address of the position structure for the next cycle; <code>pos->frame</code> will be its frame number. If <code>new_pos</code> is FALSE, this structure contains extended position information from the current cycle. If TRUE, it contains whatever was set by the requester. The <code>timebase_callback</code> 's task is to update the extended information here.
<i>new_pos</i>	TRUE (non-zero) for a newly requested pos, or for the first cycle after the <code>timebase_callback</code> is defined. This is usually 0 in <code>Sequencer64</code> at present, and 1 if one, say, presses "rewind" in <code>qjackctl</code> .
<i>arg</i>	Provides the <code>jack_assistant</code> pointer, currently unchecked for nullity.

Todo Shouldn't we process the first clause ONLY if new_pos is true?

10.23.3.4 get_current_jack_position

```
long get_current_jack_position (
    void * arg ) [friend]
```

The Seq32 version also uses its tempo map to adjust this, but Sequencer64 currently does not.

Warning

Currently valgrind flags j->[client\(\)](#) as uninitialized.

Parameters

<i>arg</i>	Provides the putative jack_assistant pointer, assumed to be not null.
------------	---

Returns

Returns the calculated tick position if no errors occur. Otherwise, returns 0.

10.23.3.5 jack_session_callback

```
void jack_session_callback (
    jack_session_event_t * ev,
    void * arg ) [friend]
```

Glib is then used to connect in `perform::jack_session_event()`. However, the `perform` object's GUI-support interface is used instead of the following, so that the `libseq64` library can be independent of a specific GUI framework:

```
Glib::signal_idle().
    connect(sigc::mem_fun(*jack, &jack_assistant::session_event));
```

Parameters

<i>ev</i>	The JACK event to be set.
<i>arg</i>	The pointer to the jack_assistant object. Currently not checked for nullity.

10.23.4 Field Documentation

10.23.4.1 sm_status_pairs

```
jack_status_pair_t seq64::jack_assistant::sm_status_pairs[] [static], [private]
```

10.23.4.2 m_jack_parent

```
perform& seq64::jack_assistant::m_jack_parent [private]
```

10.23.4.3 m_jack_client

```
jack_client_t* seq64::jack_assistant::m_jack_client [mutable], [private]
```

10.23.4.4 m_jack_client_name

```
std::string seq64::jack_assistant::m_jack_client_name [private]
```

We might show this in the user-interface at some point.

10.23.4.5 m_jack_client_uuid

```
std::string seq64::jack_assistant::m_jack_client_uuid [private]
```

We might show this in the user-interface at some point.

10.23.4.6 m_jack_frame_current

```
jack_nframes_t seq64::jack_assistant::m_jack_frame_current [private]
```

10.23.4.7 m_jack_frame_last

```
jack_nframes_t seq64::jack_assistant::m_jack_frame_last [private]
```

Also used in incrementing m_jack_tick.

10.23.4.8 m_jack_pos

```
jack_position_t seq64::jack_assistant::m_jack_pos [private]
```

This structure is filled via a call to jack_transport_query(). It holds, among other items, the frame rate (often 48000), the ticks/beat, and the beats/minute.

10.23.4.9 m_jack_transport_state

```
jack_transport_state_t seq64::jack_assistant::m_jack_transport_state [private]
```

Common values are JackTransportStopped, JackTransportRolling, and JackTransportLooping.

10.23.4.10 m_jack_transport_state_last

```
jack_transport_state_t seq64::jack_assistant::m_jack_transport_state_last [private]
```

10.23.4.11 m_jack_tick

```
double seq64::jack_assistant::m_jack_tick [private]
```

10.23.4.12 m_jsession_ev

```
jack_session_event_t* seq64::jack_assistant::m_jsession_ev [private]
```

Used in the [session_event\(\)](#) function.

10.23.4.13 m_jack_running

```
bool seq64::jack_assistant::m_jack_running [private]
```

10.23.4.14 m_jack_master

```
bool seq64::jack_assistant::m_jack_master [private]
```

10.23.4.15 m_jack_frame_rate

```
jack_nframes_t seq64::jack_assistant::m_jack_frame_rate [private]
```

Just in case. QJackCtl does not always set pos.frame_rate, so we get garbage and some strange BBT calculations displayed in qjackctl.

10.23.4.16 m_toggle_jack

```
bool seq64::jack_assistant::m_toggle_jack [private]
```

10.23.4.17 m_jack_stop_tick

```
midipulse seq64::jack_assistant::m_jack_stop_tick [private]
```

Repositions the transport marker.

10.23.4.18 m_follow_transport

```
bool seq64::jack_assistant::m_follow_transport [private]
```

10.23.4.19 m_ppqn

```
int seq64::jack_assistant::m_ppqn [private]
```

It is used for calculating ticks/beat (pulses/beat) and for setting the tick position.

10.23.4.20 m_beats_per_measure

```
int seq64::jack_assistant::m_beats_per_measure [private]
```

10.23.4.21 m_beat_width

```
int seq64::jack_assistant::m_beat_width [private]
```

10.23.4.22 m_beats_per_minute

```
midibpm seq64::jack_assistant::m_beats_per_minute [private]
```

10.24 seq64::jack_scratchpad Class Reference

Provide a temporary structure for passing data and results between a perform and [jack_assistant](#) object.

Data Fields

- double [js_current_tick](#)
Holds current location.
- double [js_total_tick](#)
Current location ignoring L/R.
- double [js_clock_tick](#)
Identical to js_total_tick.
- bool [js_jack_stopped](#)
Flags `perform::inner_stop()`.
- bool [js_dumping](#)
Non-JACK playback in progress?
- bool [js_init_clock](#)
We now have a good JACK lock.
- bool [js_looping](#)
seqedit loop button is active.
- bool [js_playback_mode](#)
Song mode (versus live mode).
- double [js_ticks_converted](#)
Keeps track of ...?
- double [js_ticks_delta](#)
Minor difference in tick.
- double [js_ticks_converted_last](#)
Keeps track of position?
- long [js_delta_tick_frac](#)
More precision for seq24 0.9.3.

10.24.1 Detailed Description

The [jack_assistant](#) class already has access to the members of `perform`, but it needs access to and modification of "local" variables in `perform::output_func()`. This scratchpad is useful even if JACK support is not enabled.

10.24.2 Field Documentation

10.24.2.1 [js_current_tick](#)

```
double seq64::jack_scratchpad::js_current_tick
```

10.24.2.2 [js_total_tick](#)

```
double seq64::jack_scratchpad::js_total_tick
```


10.24.2.3 js_clock_tick

double seq64::jack_scratchpad::js_clock_tick

10.24.2.4 js_jack_stopped

bool seq64::jack_scratchpad::js_jack_stopped

10.24.2.5 js_dumping

bool seq64::jack_scratchpad::js_dumping

10.24.2.6 js_init_clock

bool seq64::jack_scratchpad::js_init_clock

10.24.2.7 js_looping

bool seq64::jack_scratchpad::js_looping

10.24.2.8 js_playback_mode

bool seq64::jack_scratchpad::js_playback_mode

10.24.2.9 js_ticks_converted

double seq64::jack_scratchpad::js_ticks_converted

10.24.2.10 js_ticks_delta

double seq64::jack_scratchpad::js_ticks_delta

10.24.2.11 js_ticks_converted_last

```
double seq64::jack_scratchpad::js_ticks_converted_last
```

10.24.2.12 js_delta_tick_frac

```
long seq64::jack_scratchpad::js_delta_tick_frac
```

10.25 seq64::jack_status_pair_t Struct Reference

Provides an internal type to make it easier to display a specific and accurate human-readable message when a JACK operation fails.

Data Fields

- unsigned [jf_bit](#)
Holds one of the bit-values from `jack_status_t`, which is defined as an "enum JackStatus" type.
- std::string [jf_meaning](#)
Holds a textual description of the corresponding status bit.

10.25.1 Field Documentation

10.25.1.1 jf_bit

```
unsigned seq64::jack_status_pair_t::jf_bit
```

10.25.1.2 jf_meaning

```
std::string seq64::jack_status_pair_t::jf_meaning
```

10.26 seq64::keybindentry Class Reference

Class for management of application key-bindings.

Inherits `Entry`.

Public Member Functions

- [keybindentry](#) ([type](#) t, unsigned *location_to_write, [perform](#) *p=nullptr, int s=0)
This constructor initializes the member with values dependent on the value type provided in the first parameter.
- void [set](#) (unsigned val)
Gets the key name from the integer value; if there is one, then it is printed into a temporary buffer, otherwise the value is printed into that buffer as is.
- virtual bool [on_key_press_event](#) (GdkEventKey *event)
Handles a key press by calling [set\(\)](#) with the event's key value.

Private Types

- enum [type](#) {
 [location](#),
 [events](#),
 [groups](#) }
Provides the type of keybindings that can be made.

Private Attributes

- unsigned * [m_key](#)
Points to the value of the key that is part of this key-binding.
- [type](#) [m_type](#)
Stores the type of key-binding.
- [perform](#) * [m_perf](#)
Stores an optional pointer to a perform object.
- int [m_slot](#)
Provides an index into a set of group-keys or event-keys.

Friends

- class [options](#)

10.26.1 Member Enumeration Documentation

10.26.1.1 type

```
enum seq64::keybindentry::type [private]
```

Enumerator

location	Used for handling a keystroke made while a keyboard-options field is active, for selecting a key via the keyboard, and binding to pattern/sequence boxes, we think. It is used in the options class to associate a key with the binding.
events	Used for binding to events.
groups	Used for binding to groups.

10.26.2 Constructor & Destructor Documentation

10.26.2.1 keybindentry()

```
seq64::keybindentry::keybindentry (
    type t,
    unsigned * location_to_write,
    perform * p = nullptr,
    int s = 0 )
```

Usage In options, a pointer to a new key-binding entry is managed by calling `keybindentry(keybindentry←::location, &perf->keyname)`.

Parameters

<i>t</i>	Provides the type of key-binding: location, events, or groups.
<i>location_to_write</i>	The location that holds the value of the key associated with the key-binding.
<i>p</i>	Points to the performance object used with this key-binding. The default value of this parameter is the null pointer, but it is needed for the pattern hot-keys frame and the mute-group frame of the Options dialog.
<i>s</i>	Provides the slot value for this key-binding. The default value of this parameter is zero, but it is needed to provide numeric labels for the hot-keys and mute-group frames of the Options dialog.

10.26.3 Member Function Documentation

10.26.3.1 set()

```
void seq64::keybindentry::set (
    unsigned val )
```

Then we call `set_text(buf)`. The `set_width_char()` function is then called.

10.26.3.2 on_key_press_event()

```
bool seq64::keybindentry::on_key_press_event (
    GdkEventKey * event ) [virtual]
```

This value is used to set the event or key depending on the value of `m_type`.

Parameters

<i>event</i>	Provides the key-press event.
--------------	-------------------------------

Returns

Returns the result of the call to Entry::on_key_press_event().

10.26.4 Friends And Related Function Documentation

10.26.4.1 options

```
friend class options [friend]
```

10.26.5 Field Documentation

10.26.5.1 m_key

```
unsigned* seq64::keybindentry::m_key [private]
```

Not yet sure by the address of this key value is needed. It can be a null pointer, as well.

10.26.5.2 m_type

```
type seq64::keybindentry::m_type [private]
```

10.26.5.3 m_perf

```
perform* seq64::keybindentry::m_perf [private]
```

10.26.5.4 m_slot

```
int seq64::keybindentry::m_slot [private]
```

(This item should be changed to unsigned, though.)

10.27 seq64::keys_perform Class Reference

This class supports the performance mode.

Inheritance diagram for seq64::keys_perform:



Public Member Functions

- [keys_perform](#) ()

This construction initializes a vast number of member variables, some of them public!

- virtual `~keys_perform ()`
The destructor is a rote empty virtual destructor.
- void `set_keys (const keys_perform_transfer &kpt)`
Copies fields from the transfer structure in this object.
- void `get_keys (keys_perform_transfer &kpt)`
Copies fields from this object into the transfer structure.
- unsigned `bpm_up () const`
'Getter' function for member m_key_bpm_up
- void `bpm_up (unsigned x)`
'Setter' function for member m_key_bpm_up
- unsigned `bpm_dn () const`
'Getter' function for member m_key_bpm_dn
- void `bpm_dn (unsigned x)`
'Setter' function for member m_key_bpm_dn
- unsigned `replace () const`
'Getter' function for member m_key_replace
- void `replace (unsigned x)`
'Setter' function for member m_key_replace
- unsigned `queue () const`
'Getter' function for member m_key_queue
- void `queue (unsigned x)`
'Setter' function for member m_key_queue
- unsigned `keep_queue () const`
'Getter' function for member m_key_keep_queue
- void `keep_queue (unsigned x)`
'Setter' function for member m_key_keep_queue
- unsigned `snapshot_1 () const`
'Getter' function for member m_key_snapshot_1
- void `snapshot_1 (unsigned x)`
'Setter' function for member m_key_snapshot_1
- unsigned `snapshot_2 () const`
'Getter' function for member m_key_snapshot_2
- void `snapshot_2 (unsigned x)`
'Setter' function for member m_key_snapshot_2
- unsigned `screensset_up () const`
'Getter' function for member m_key_screensset_up
- void `screensset_up (unsigned x)`
'Setter' function for member m_key_screensset_up
- unsigned `screensset_dn () const`
'Getter' function for member m_key_screensset_dn
- void `screensset_dn (unsigned x)`
'Setter' function for member m_key_screensset_dn
- unsigned `set_playing_screensset () const`
'Getter' function for member m_key_playing_screensset
- void `set_playing_screensset (unsigned x)`
'Setter' function for member m_key_playing_screensset
- unsigned `group_on () const`
'Getter' function for member m_key_group_on
- void `group_on (unsigned x)`
'Setter' function for member m_key_group_on

- unsigned [group_off](#) () const
'Getter' function for member m_key_group_off
- void [group_off](#) (unsigned x)
'Setter' function for member m_key_group_off
- unsigned [group_learn](#) () const
'Getter' function for member m_key_group_learn
- void [group_learn](#) (unsigned x)
'Setter' function for member m_key_group_learn
- unsigned [start](#) () const
'Getter' function for member m_key_start
- void [start](#) (unsigned x)
'Setter' function for member m_key_start
- unsigned [pause](#) () const
'Getter' function for member m_key_pause
- void [pause](#) (unsigned x)
'Setter' function for member m_key_pause
- unsigned [pattern_edit](#) () const
'Getter' function for member m_key_pattern_edit
- void [pattern_edit](#) (unsigned x)
'Setter' function for member m_key_pattern_edit
- unsigned [pattern_shift](#) () const
'Getter' function for member m_key_pattern_shift
- void [pattern_shift](#) (unsigned x)
'Setter' function for member m_key_pattern_shift
- unsigned [event_edit](#) () const
'Getter' function for member m_key_event_edit
- void [event_edit](#) (unsigned x)
'Setter' function for member m_key_event_edit
- unsigned [stop](#) () const
'Getter' function for member m_key_stop
- void [stop](#) (unsigned x)
'Setter' function for member m_key_stop
- unsigned [song_mode](#) () const
- void [song_mode](#) (unsigned key)
- unsigned [menu_mode](#) () const
- void [menu_mode](#) (unsigned key)
- unsigned [follow_transport](#) () const
- void [follow_transport](#) (unsigned key)
- unsigned [fast_forward](#) () const
- void [fast_forward](#) (unsigned key)
- unsigned [rewind](#) () const
- void [rewind](#) (unsigned key)
- unsigned [pointer_position](#) () const
- void [pointer_position](#) (unsigned key)
- unsigned [toggle_mutes](#) () const
- void [toggle_mutes](#) (unsigned key)
- unsigned [toggle_jack](#) () const
- void [toggle_jack](#) (unsigned key)
- unsigned [tap_bpm](#) () const
- void [tap_bpm](#) (unsigned key)
- unsigned [song_record](#) () const
- void [song_record](#) (unsigned key)

- unsigned [oneshot_queue](#) () const
- void [oneshot_queue](#) (unsigned key)
- bool [show_ui_sequence_key](#) () const
'Getter' function for member m_key_show_ui_sequency_key
- void [show_ui_sequence_key](#) (bool flag)
'Setter' function for member m_key_show_ui_sequency_key
- bool [show_ui_sequence_number](#) () const
'Getter' function for member m_key_show_ui_sequence_number
- void [show_ui_sequence_number](#) (bool flag)
'Setter' function for member m_key_show_ui_sequence_key
- [SlotMap](#) & [get_key_events](#) ()
'Getter' function for member m_key_events
- int [get_key_count](#) (unsigned k) const
Returns the number of times the given key appears in the SlotMap, either 0 or 1.
- [SlotMap](#) & [get_key_groups](#) ()
'Getter' function for member m_key_groups
- [RevSlotMap](#) & [get_key_events_rev](#) ()
'Getter' function for member m_key_events_rev
- [RevSlotMap](#) & [get_key_groups_rev](#) ()
'Getter' function for member m_key_groups_rev
- unsigned [lookup_keyevent_key](#) (int seqnum)
'Getter' function for member m_key_events_rev[seqnum]
- unsigned [lookup_keygroup_key](#) (int groupnum)
'Getter' function for member m_key_events_rev[groupnum]
- int [lookup_keyevent_seq](#) (unsigned keycode)
'Getter' function for member m_key_events_rev[keycode]
- int [lookup_keygroup_group](#) (unsigned keycode)
'Getter' function for member m_key_events_rev[keycode]
- virtual std::string [key_name](#) (unsigned key) const
Obtains the name of the key.
- virtual void [set_all_key_events](#) ()
Provides base class functionality.
- virtual void [set_all_key_groups](#) ()
Provides base class functionality.
- void [set_key_event](#) (unsigned keycode, int sequence_slot)
At construction time, this function sets up one keycode and one event slot.
- void [set_key_group](#) (unsigned keycode, int group_slot)
At construction time, this function sets up one keycode and one group slot.
- int [group_max](#) () const
'Getter' function for member m_group_max

Protected Types

- typedef std::map< unsigned, int > [SlotMap](#)
This typedef defines a map in which the key is the keycode, that is, the integer value of a keystroke, and the value is the pattern/sequence number or slot.
- typedef std::map< int, unsigned > [RevSlotMap](#)
This typedef is like SlotMap, but used for lookup in the other direction.

Protected Member Functions

- void [group_max](#) (int groupcount)
'Setter' function for member m_group_max
- unsigned * [at_bpm_up](#) ()
The following are tricky ways to get at address of the key and group operation values so that we don't directly expose the members to manipulation.
- unsigned * [at_bpm_dn](#) ()
'Getter' function for member m_key_bpm_dn Address getter for the bpm_dn operation.
- unsigned * [at_replace](#) ()
'Getter' function for member m_key_replace Address getter for the replace operation.
- unsigned * [at_queue](#) ()
'Getter' function for member m_key_queue Address getter for the queue operation.
- unsigned * [at_keep_queue](#) ()
'Getter' function for member m_key_keep_queue Address getter for the keep_queue operation.
- unsigned * [at_snapshot_1](#) ()
'Getter' function for member m_key_snapshot_1 Address getter for the snapshot_1 operation.
- unsigned * [at_snapshot_2](#) ()
'Getter' function for member m_key_snapshot_2 Address getter for the snapshot_2 operation.
- unsigned * [at_screenset_up](#) ()
'Getter' function for member m_key_screenset_up Address getter for the screenset_up operation.
- unsigned * [at_screenset_dn](#) ()
'Getter' function for member m_key_screenset_dn Address getter for the screenset_dn operation.
- unsigned * [at_set_playing_screenset](#) ()
'Getter' function for member m_key_playing_screenset Address getter for the set_playing_screenset operation.
- unsigned * [at_group_on](#) ()
'Getter' function for member m_key_group_on Address getter for the group_on operation.
- unsigned * [at_group_off](#) ()
'Getter' function for member m_key_group_off Address getter for the group_off operation.
- unsigned * [at_group_learn](#) ()
'Getter' function for member m_key_group_learn Address getter for the group_learn operation.
- unsigned * [at_start](#) ()
'Getter' function for member m_key_start Address getter for the start operation.
- unsigned * [at_pause](#) ()
'Getter' function for member m_key_pause Address getter for the pause operation.
- unsigned * [at_song_mode](#) ()
'Getter' function for member m_key_song_mode Address getter for the song-mode operation.
- unsigned * [at_toggle_jack](#) ()
'Getter' function for member m_key_toggle_jack Address getter for the toggle-jack operation.
- unsigned * [at_menu_mode](#) ()
'Getter' function for member m_key_menu_mode Address getter for the menu-mode operation.
- unsigned * [at_follow_transport](#) ()
'Getter' function for member m_key_follow_transport Address getter for the follow-transport operation.
- unsigned * [at_fast_forward](#) ()
'Getter' function for member m_key_fast_forward Address getter for the fast-forward operation.
- unsigned * [at_rewind](#) ()
'Getter' function for member m_key_rewind Address getter for the rewind operation.
- unsigned * [at_pointer_position](#) ()
'Getter' function for member m_key_pointer_position Address getter for the pointer operation.
- unsigned * [at_toggle_mutes](#) ()
'Getter' function for member m_key_toggle_mutes Address getter for the toggle-mutes operation.

- unsigned * [at_tap_bpm](#) ()
'Getter' function for member m_key_tap_bpm Address getter for the tap_bpm operation.
- unsigned * [at_song_record](#) ()
'Getter' function for member m_key_song_record Address getter for the song-record operation.
- unsigned * [at_oneshot_queue](#) ()
'Getter' function for member m_key_oneshot_queue Address getter for the oneshot-record operation.
- unsigned * [at_pattern_edit](#) ()
'Getter' function for member m_key_pattern_edit Address getter for the pattern-edit operation.
- unsigned * [at_pattern_shift](#) ()
'Getter' function for member m_key_pattern_shift Address getter for the pattern-shift operation.
- unsigned * [at_event_edit](#) ()
'Getter' function for member m_key_event_edit Address getter for the event-edit operation.
- unsigned * [at_stop](#) ()
'Getter' function for member m_key_stop Address getter for the stop operation.
- bool * [at_show_ui_sequence_key](#) ()
'Getter' function for member m_key_show_ui_sequence_key Address getter for the show_ui_sequence_key value.
- bool * [at_show_ui_sequence_number](#) ()
'Getter' function for member m_key_show_ui_sequence_number Address getter for the show_ui_sequence_number value.
- virtual void [set_basic_key_events](#) ()
- virtual void [set_basic_key_groups](#) ()

Private Attributes

- bool [m_key_show_ui_sequence_key](#)
If set, shows the shortcut-keys on each filled pattern slot in the main window.
- bool [m_key_show_ui_sequence_number](#)
If set, shows the sequence number on each filled pattern and empty pattern slot in the main window.
- [SlotMap](#) [m_key_events](#)
Holds the mapping of keys to the pattern slots.
- [SlotMap](#) [m_key_groups](#)
Holds the mapping of keys to the mute groups.
- [RevSlotMap](#) [m_key_events_rev](#)
Holds the reverse mapping of the pattern slots to the keys.
- [RevSlotMap](#) [m_key_groups_rev](#)
Holds the reverse mapping of the mute groups to the keys.
- int [m_group_max](#)
With larger set sizes, we have to support fewer mute-groups (unless we decide to allow 32 sets even for larger sets, upping the pattern count from 1024 to 3072 or 4096).
- unsigned [m_key_bpm_up](#)
Provides key assignments for some key sequencer features.
- unsigned [m_key_bpm_dn](#)
BPM down, semicolon.
- unsigned [m_key_replace](#)
Replace, Ctrl-L.
- unsigned [m_key_queue](#)
Queue, Ctrl-R.
- unsigned [m_key_keep_queue](#)
Keep queue, backslash.
- unsigned [m_key_snapshot_1](#)

- Snapshot 1, Alt-L.*
- unsigned [m_key_snapshot_2](#)
 - Snapshot 1, Alt-R.*
- unsigned [m_key_screenset_up](#)
 - Set up, Right-].*
- unsigned [m_key_screenset_dn](#)
 - Set down, Left-[.*
- unsigned [m_key_set_playing_screenset](#)
 - Set set, Home key.*
- unsigned [m_key_group_on](#)
 - Group on, igrave key.*
- unsigned [m_key_group_off](#)
 - Group off, apostrophe!*
- unsigned [m_key_group_learn](#)
 - Group learn, Insert.*
- unsigned [m_key_start](#)
 - Start play, Space key.*
- unsigned [m_key_pause](#)
 - Pause play, Period.*
- unsigned [m_key_song_mode](#)
 - Song versus Live mode.*
- unsigned [m_key_toggle_jack](#)
 - Toggle JACK connect.*
- unsigned [m_key_menu_mode](#)
 - Menu enabled/disabled.*
- unsigned [m_key_follow_transport](#)
 - Toggle following JACK.*
- unsigned [m_key_rewind](#)
 - Start rewind.*
- unsigned [m_key_fast_forward](#)
 - Start fast-forward.*
- unsigned [m_key_pointer_position](#)
 - Set progress to mouse.*
- unsigned [m_key_toggle_mutes](#)
 - Toggle all patterns.*
- unsigned [m_key_tap_bpm](#)
 - To tap out the BPM.*
- unsigned [m_key_pattern_edit](#)
 - Show pattern editor.*
- unsigned [m_key_pattern_shift](#)
 - Shift pattern hotkey.*
- unsigned [m_key_event_edit](#)
 - Show event editor.*
- unsigned [m_key_stop](#)
 - Stop play, Escape.*
- unsigned [m_key_song_record](#)
 - Turn on song-record.*
- unsigned [m_key_oneshot_queue](#)
 - Turn on 1-shot record.*

Friends

- class [options](#)
- class [perform](#)
- class [optionsfile](#)

10.27.1 Detailed Description

It provides a way a mapping keystrokes to sequencer actions and song settings.

10.27.2 Member Typedef Documentation

10.27.2.1 SlotMap

```
typedef std::map<unsigned, int> seq64::keys\_perform::SlotMap [protected]
```

10.27.2.2 RevSlotMap

```
typedef std::map<int, unsigned> seq64::keys\_perform::RevSlotMap [protected]
```

10.27.3 Constructor & Destructor Documentation

10.27.3.1 keys_perform()

```
seq64::keys\_perform::keys\_perform ( )
```

10.27.3.2 ~keys_perform()

```
seq64::keys\_perform::~~keys\_perform ( ) [virtual]
```

10.27.4 Member Function Documentation

10.27.4.1 set_keys()

```
void seq64::keys\_perform::set\_keys (  
    const keys\_perform\_transfer & kpt )
```

This structure holds all of the key settings from the File / Options / Keyboard tab dialog.

Parameters

<i>kpt</i>	The structure that holds the values of the keys to be used for various purposes in controlling a performance live.
------------	--

10.27.4.2 `get_keys()`

```
void seq64::keys_perform::get_keys (
    keys_perform_transfer & kpt )
```

Parameters

<i>kpt</i>	The structure that holds the values of the keys to be used for various purposes in controlling a performance live.
------------	--

10.27.4.3 `bpm_up()` [1/2]

```
unsigned seq64::keys_perform::bpm_up ( ) const [inline]
```

10.27.4.4 `bpm_up()` [2/2]

```
void seq64::keys_perform::bpm_up (
    unsigned x ) [inline]
```

Parameters

<i>x</i>	The key value to assign to the operation.
----------	---

10.27.4.5 `bpm_dn()` [1/2]

```
unsigned seq64::keys_perform::bpm_dn ( ) const [inline]
```

10.27.4.6 `bpm_dn()` [2/2]

```
void seq64::keys_perform::bpm_dn (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.7 replace() [1/2]

```
unsigned seq64::keys_perform::replace ( ) const [inline]
```

10.27.4.8 replace() [2/2]

```
void seq64::keys_perform::replace (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.9 queue() [1/2]

```
unsigned seq64::keys_perform::queue ( ) const [inline]
```

10.27.4.10 queue() [2/2]

```
void seq64::keys_perform::queue (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.11 keep_queue() [1/2]

```
unsigned seq64::keys_perform::keep_queue ( ) const [inline]
```

10.27.4.12 keep_queue() [2/2]

```
void seq64::keys_perform::keep_queue (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.13 snapshot_1() [1/2]

```
unsigned seq64::keys_perform::snapshot_1 ( ) const [inline]
```

10.27.4.14 snapshot_1() [2/2]

```
void seq64::keys_perform::snapshot_1 (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.15 snapshot_2() [1/2]

```
unsigned seq64::keys_perform::snapshot_2 ( ) const [inline]
```

10.27.4.16 snapshot_2() [2/2]

```
void seq64::keys_perform::snapshot_2 (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.17 screenset_up() [1/2]

```
unsigned seq64::keys_perform::screenset_up ( ) const [inline]
```

10.27.4.18 screenset_up() [2/2]

```
void seq64::keys_perform::screenset_up (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.19 screenset_dn() [1/2]

```
unsigned seq64::keys_perform::screenset_dn ( ) const [inline]
```

10.27.4.20 screenset_dn() [2/2]

```
void seq64::keys_perform::screenset_dn (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.21 set_playing_screenset() [1/2]

```
unsigned seq64::keys_perform::set_playing_screenset ( ) const [inline]
```

10.27.4.22 set_playing_screenset() [2/2]

```
void seq64::keys_perform::set_playing_screenset (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.23 group_on() [1/2]

```
unsigned seq64::keys_perform::group_on ( ) const [inline]
```

10.27.4.24 group_on() [2/2]

```
void seq64::keys_perform::group_on (  
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.25 group_off() [1/2]

```
unsigned seq64::keys_perform::group_off ( ) const [inline]
```

10.27.4.26 group_off() [2/2]

```
void seq64::keys_perform::group_off (  
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.27 group_learn() [1/2]

```
unsigned seq64::keys_perform::group_learn ( ) const [inline]
```

10.27.4.28 group_learn() [2/2]

```
void seq64::keys_perform::group_learn (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.29 start() [1/2]

```
unsigned seq64::keys_perform::start ( ) const [inline]
```

10.27.4.30 start() [2/2]

```
void seq64::keys_perform::start (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.31 pause() [1/2]

```
unsigned seq64::keys_perform::pause ( ) const [inline]
```

10.27.4.32 pause() [2/2]

```
void seq64::keys_perform::pause (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.33 `pattern_edit()` [1/2]

```
unsigned seq64::keys_perform::pattern_edit ( ) const [inline]
```

10.27.4.34 `pattern_edit()` [2/2]

```
void seq64::keys_perform::pattern_edit (
    unsigned x ) [inline]
```

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

10.27.4.35 `pattern_shift()` [1/2]

```
unsigned seq64::keys_perform::pattern_shift ( ) const [inline]
```

10.27.4.36 `pattern_shift()` [2/2]

```
void seq64::keys_perform::pattern_shift (
    unsigned x ) [inline]
```

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

10.27.4.37 `event_edit()` [1/2]

```
unsigned seq64::keys_perform::event_edit ( ) const [inline]
```

10.27.4.38 `event_edit()` [2/2]

```
void seq64::keys_perform::event_edit (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.39 stop() [1/2]

```
unsigned seq64::keys_perform::stop ( ) const [inline]
```

10.27.4.40 stop() [2/2]

```
void seq64::keys_perform::stop (
    unsigned x ) [inline]
```

Parameters

x	The key value to assign to the operation.
---	---

10.27.4.41 song_mode() [1/2]

```
unsigned seq64::keys_perform::song_mode ( ) const [inline]
```

10.27.4.42 song_mode() [2/2]

```
void seq64::keys_perform::song_mode (
    unsigned key ) [inline]
```

10.27.4.43 menu_mode() [1/2]

```
unsigned seq64::keys_perform::menu_mode ( ) const [inline]
```

10.27.4.44 menu_mode() [2/2]

```
void seq64::keys_perform::menu_mode (
    unsigned key ) [inline]
```

10.27.4.45 follow_transport() [1/2]

```
unsigned seq64::keys_perform::follow_transport ( ) const [inline]
```

10.27.4.46 follow_transport() [2/2]

```
void seq64::keys_perform::follow_transport (
    unsigned key ) [inline]
```

10.27.4.47 fast_forward() [1/2]

```
unsigned seq64::keys_perform::fast_forward ( ) const [inline]
```

10.27.4.48 fast_forward() [2/2]

```
void seq64::keys_perform::fast_forward (
    unsigned key ) [inline]
```

10.27.4.49 rewind() [1/2]

```
unsigned seq64::keys_perform::rewind ( ) const [inline]
```

10.27.4.50 rewind() [2/2]

```
void seq64::keys_perform::rewind (
    unsigned key ) [inline]
```

10.27.4.51 pointer_position() [1/2]

```
unsigned seq64::keys_perform::pointer_position ( ) const [inline]
```

10.27.4.52 pointer_position() [2/2]

```
void seq64::keys_perform::pointer_position (
    unsigned key ) [inline]
```

10.27.4.53 toggle_mutes() [1/2]

```
unsigned seq64::keys_perform::toggle_mutes ( ) const [inline]
```

10.27.4.54 toggle_mutes() [2/2]

```
void seq64::keys_perform::toggle_mutes (
    unsigned key ) [inline]
```

10.27.4.55 toggle_jack() [1/2]

```
unsigned seq64::keys_perform::toggle_jack ( ) const [inline]
```

10.27.4.56 toggle_jack() [2/2]

```
void seq64::keys_perform::toggle_jack (
    unsigned key ) [inline]
```

10.27.4.57 tap_bpm() [1/2]

```
unsigned seq64::keys_perform::tap_bpm ( ) const [inline]
```

10.27.4.58 tap_bpm() [2/2]

```
void seq64::keys_perform::tap_bpm (
    unsigned key ) [inline]
```

10.27.4.59 song_record() [1/2]

```
unsigned seq64::keys_perform::song_record ( ) const [inline]
```

10.27.4.60 song_record() [2/2]

```
void seq64::keys_perform::song_record (
    unsigned key ) [inline]
```

10.27.4.61 oneshot_queue() [1/2]

```
unsigned seq64::keys_perform::oneshot_queue ( ) const [inline]
```

10.27.4.62 oneshot_queue() [2/2]

```
void seq64::keys_perform::oneshot_queue (
    unsigned key ) [inline]
```

10.27.4.63 show_ui_sequence_key() [1/2]

```
bool seq64::keys_perform::show_ui_sequence_key ( ) const [inline]
```

Used in mainwid, options, optionsfile, userfile, and perform.

10.27.4.64 show_ui_sequence_key() [2/2]

```
void seq64::keys_perform::show_ui_sequence_key (
    bool flag ) [inline]
```

Parameters

<i>flag</i>	The flag for showing the sequence key characters in each pattern slot.
-------------	--

10.27.4.65 show_ui_sequence_number() [1/2]

```
bool seq64::keys_perform::show_ui_sequence_number ( ) const [inline]
```


Used in mainwid, options, optionsfile, userfile, and perform.

10.27.4.66 show_ui_sequence_number() [2/2]

```
void seq64::keys_perform::show_ui_sequence_number (
    bool flag ) [inline]
```

Parameters

<i>flag</i>	The flag for showing the sequence number in each pattern slot.
-------------	--

10.27.4.67 get_key_events()

```
SlotMap& seq64::keys_perform::get_key_events ( ) [inline]
```

10.27.4.68 get_key_count()

```
int seq64::keys_perform::get_key_count (
    unsigned k ) const [inline]
```

Parameters

<i>k</i>	The key value to be checked.
----------	------------------------------

10.27.4.69 get_key_groups()

```
SlotMap& seq64::keys_perform::get_key_groups ( ) [inline]
```

10.27.4.70 get_key_events_rev()

```
RevSlotMap& seq64::keys_perform::get_key_events_rev ( ) [inline]
```

10.27.4.71 get_key_groups_rev()

```
RevSlotMap& seq64::keys_perform::get_key_groups_rev ( ) [inline]
```

10.27.4.72 lookup_keyevent_key()

```
unsigned seq64::keys_perform::lookup_keyevent_key (
    int seqnum )
```

Parameters

<i>seqnum</i>	Provides the sequence number to look up in the reverse key map for patterns/sequences. If the count for this value is 0, then a SEQ64_Clear character is returned.
---------------	--

10.27.4.73 lookup_keygroup_key()

```
unsigned seq64::keys_perform::lookup_keygroup_key (
    int groupnum )
```

Parameters

<i>groupnum</i>	Provides the group number to look up in the reverse key map for groups.
-----------------	---

Returns

Returns the key for the desired group. If the count for the desired group is 0, then a SEQ64_Clear character is returned.

10.27.4.74 lookup_keyevent_seq()

```
int seq64::keys_perform::lookup_keyevent_seq (
    unsigned keycode )
```

Parameters

<i>keycode</i>	Provides the keycode to look up in the (forward) key map for patterns/sequences. If the count for this value is 0, then a 0 is returned.
----------------	--

10.27.4.75 lookup_keygroup_group()

```
int seq64::keys_perform::lookup_keygroup_group (
    unsigned keycode )
```

Parameters

<i>keycode</i>	Provides the sequence number to look up in the reverse key map for groups. If the count for this value is 0, then a 0 is returned. We might consider returning (-1) as an error code at some point.
----------------	---

10.27.4.76 key_name()

```
std::string seq64::keys_perform::key_name (
    unsigned key ) const [virtual]
```

In gtkmm, this is done via the `gdk_keyval_name()` function. Here, in the base class, we just provide an easy-to-create string.

Parameters

<i>key</i>	Provides the numeric value of the keystroke.
------------	--

Returns

Returns the name of the key, in the format "Key 0xkkkk".

Reimplemented in [seq64::keys_perform_gtk2](#).

10.27.4.77 set_all_key_events()

```
virtual void seq64::keys_perform::set_all_key_events ( ) [inline], [virtual]
```

Must be called by the derived-class's override of this function.

Reimplemented in [seq64::keys_perform_gtk2](#).

10.27.4.78 set_all_key_groups()

```
virtual void seq64::keys_perform::set_all_key_groups ( ) [inline], [virtual]
```

Must be called by the derived-class's override of this function.

Reimplemented in [seq64::keys_perform_gtk2](#).

10.27.4.79 set_key_event()

```
void seq64::keys_perform::set_key_event (
    unsigned keycode,
    int sequence_slot )
```

It is called 32 times, corresponding the pattern/sequence slots in the Patterns window.

Parameters

<i>keycode</i>	The key to be assigned.
<i>sequence_slot</i>	The perform event slot into which the keycode will be assigned.

10.27.4.80 `set_key_group()`

```
void seq64::keys_perform::set_key_group (
    unsigned keycode,
    int group_slot )
```

It is called 32 times, corresponding the pattern/sequence slots in the Patterns window.

Parameters

<i>keycode</i>	The key to be assigned.
<i>group_slot</i>	The perform group slot into which the keycode will be assigned.

10.27.4.81 `group_max()` [1/2]

```
int seq64::keys_perform::group_max ( ) const [inline]
```

10.27.4.82 `group_max()` [2/2]

```
void seq64::keys_perform::group_max (
    int groupcount ) [inline], [protected]
```

10.27.4.83 `at_bpm_up()`

```
unsigned* seq64::keys_perform::at_bpm_up ( ) [inline], [protected]
```

They are used in the options module, and, for brevity, are accessed using the `PREFKEY_ADDR()` macro. 'Getter' function for member `m_key_bpm_up` Address getter for the bpm_up operation.

10.27.4.84 `at_bpm_dn()`

```
unsigned* seq64::keys_perform::at_bpm_dn ( ) [inline], [protected]
```

10.27.4.85 at_replace()

```
unsigned* seq64::keys_perform::at_replace ( ) [inline], [protected]
```

10.27.4.86 at_queue()

```
unsigned* seq64::keys_perform::at_queue ( ) [inline], [protected]
```

10.27.4.87 at_keep_queue()

```
unsigned* seq64::keys_perform::at_keep_queue ( ) [inline], [protected]
```

10.27.4.88 at_snapshot_1()

```
unsigned* seq64::keys_perform::at_snapshot_1 ( ) [inline], [protected]
```

10.27.4.89 at_snapshot_2()

```
unsigned* seq64::keys_perform::at_snapshot_2 ( ) [inline], [protected]
```

10.27.4.90 at_screenset_up()

```
unsigned* seq64::keys_perform::at_screenset_up ( ) [inline], [protected]
```

10.27.4.91 at_screenset_dn()

```
unsigned* seq64::keys_perform::at_screenset_dn ( ) [inline], [protected]
```

10.27.4.92 at_set_playing_screenset()

```
unsigned* seq64::keys_perform::at_set_playing_screenset ( ) [inline], [protected]
```

10.27.4.93 at_group_on()

```
unsigned* seq64::keys_perform::at_group_on ( ) [inline], [protected]
```

10.27.4.94 at_group_off()

```
unsigned* seq64::keys_perform::at_group_off ( ) [inline], [protected]
```

10.27.4.95 at_group_learn()

```
unsigned* seq64::keys_perform::at_group_learn ( ) [inline], [protected]
```

10.27.4.96 at_start()

```
unsigned* seq64::keys_perform::at_start ( ) [inline], [protected]
```

10.27.4.97 at_pause()

```
unsigned* seq64::keys_perform::at_pause ( ) [inline], [protected]
```

10.27.4.98 at_song_mode()

```
unsigned* seq64::keys_perform::at_song_mode ( ) [inline], [protected]
```

10.27.4.99 at_toggle_jack()

```
unsigned* seq64::keys_perform::at_toggle_jack ( ) [inline], [protected]
```

10.27.4.100 at_menu_mode()

```
unsigned* seq64::keys_perform::at_menu_mode ( ) [inline], [protected]
```

10.27.4.101 at_follow_transport()

```
unsigned* seq64::keys_perform::at_follow_transport ( ) [inline], [protected]
```

10.27.4.102 at_fast_forward()

```
unsigned* seq64::keys_perform::at_fast_forward ( ) [inline], [protected]
```

10.27.4.103 at_rewind()

```
unsigned* seq64::keys_perform::at_rewind ( ) [inline], [protected]
```

10.27.4.104 at_pointer_position()

```
unsigned* seq64::keys_perform::at_pointer_position ( ) [inline], [protected]
```

10.27.4.105 at_toggle_mutes()

```
unsigned* seq64::keys_perform::at_toggle_mutes ( ) [inline], [protected]
```

10.27.4.106 at_tap_bpm()

```
unsigned* seq64::keys_perform::at_tap_bpm ( ) [inline], [protected]
```

10.27.4.107 at_song_record()

```
unsigned* seq64::keys_perform::at_song_record ( ) [inline], [protected]
```

10.27.4.108 at_oneshot_queue()

```
unsigned* seq64::keys_perform::at_oneshot_queue ( ) [inline], [protected]
```

10.27.4.109 at_pattern_edit()

```
unsigned* seq64::keys_perform::at_pattern_edit ( ) [inline], [protected]
```

10.27.4.110 at_pattern_shift()

```
unsigned* seq64::keys_perform::at_pattern_shift ( ) [inline], [protected]
```

10.27.4.111 at_event_edit()

```
unsigned* seq64::keys_perform::at_event_edit ( ) [inline], [protected]
```

10.27.4.112 at_stop()

```
unsigned* seq64::keys_perform::at_stop ( ) [inline], [protected]
```

10.27.4.113 at_show_ui_sequence_key()

```
bool* seq64::keys_perform::at_show_ui_sequence_key ( ) [inline], [protected]
```

10.27.4.114 at_show_ui_sequence_number()

```
bool* seq64::keys_perform::at_show_ui_sequence_number ( ) [inline], [protected]
```

10.27.4.115 set_basic_key_events()

```
void seq64::keys_perform::set_basic_key_events ( ) [protected], [virtual]
```

10.27.4.116 set_basic_key_groups()

```
void seq64::keys_perform::set_basic_key_groups ( ) [protected], [virtual]
```


10.27.5 Friends And Related Function Documentation

10.27.5.1 options

```
friend class options [friend]
```

10.27.5.2 perform

```
friend class perform [friend]
```

10.27.5.3 optionsfile

```
friend class optionsfile [friend]
```

10.27.6 Field Documentation

10.27.6.1 m_key_show_ui_sequence_key

```
bool seq64::keys_perform::m_key_show_ui_sequence_key [private]
```

10.27.6.2 m_key_show_ui_sequence_number

```
bool seq64::keys_perform::m_key_show_ui_sequence_number [private]
```

Also shows the sequence number as part of the sequence name in the performance window (song editor). Always disabled in legacy mode.

10.27.6.3 m_key_events

```
SlotMap seq64::keys_perform::m_key_events [private]
```

Do not access directly, use the set/lookup functions declared below.

10.27.6.4 m_key_groups

```
SlotMap seq64::keys_perform::m_key_groups [private]
```

Do not access directly, use the set/lookup functions declared below.

10.27.6.5 m_key_events_rev

```
RevSlotMap seq64::keys_perform::m_key_events_rev [private]
```

Do not access directly, use the set/lookup functions declared below.

10.27.6.6 m_key_groups_rev

```
RevSlotMap seq64::keys_perform::m_key_groups_rev [private]
```

Do not access directly, use the set/lookup functions declared below.

10.27.6.7 m_group_max

```
int seq64::keys_perform::m_group_max [private]
```

This number is logged by the perform object to allow us to know that we can actually use the desired mute group.

10.27.6.8 m_key_bpm_up

```
unsigned seq64::keys_perform::m_key_bpm_up [private]
```

Used in mainwnd, options, optionsfile, perfedit, seqroll, userfile, and perform.

We could instead use the [keys_perform_transfer](#) structure instead of all these individual members. BPM up, apostrophe!!!

10.27.6.9 m_key_bpm_dn

```
unsigned seq64::keys_perform::m_key_bpm_dn [private]
```

10.27.6.10 m_key_replace

```
unsigned seq64::keys_perform::m_key_replace [private]
```

10.27.6.11 m_key_queue

unsigned seq64::keys_perform::m_key_queue [private]

10.27.6.12 m_key_keep_queue

unsigned seq64::keys_perform::m_key_keep_queue [private]

10.27.6.13 m_key_snapshot_1

unsigned seq64::keys_perform::m_key_snapshot_1 [private]

10.27.6.14 m_key_snapshot_2

unsigned seq64::keys_perform::m_key_snapshot_2 [private]

10.27.6.15 m_key_screenset_up

unsigned seq64::keys_perform::m_key_screenset_up [private]

10.27.6.16 m_key_screenset_dn

unsigned seq64::keys_perform::m_key_screenset_dn [private]

10.27.6.17 m_key_set_playing_screenset

unsigned seq64::keys_perform::m_key_set_playing_screenset [private]

10.27.6.18 m_key_group_on

unsigned seq64::keys_perform::m_key_group_on [private]

10.27.6.19 m_key_group_off

unsigned seq64::keys_perform::m_key_group_off [private]

10.27.6.20 m_key_group_learn

unsigned seq64::keys_perform::m_key_group_learn [private]

10.27.6.21 m_key_start

unsigned seq64::keys_perform::m_key_start [private]

10.27.6.22 m_key_pause

unsigned seq64::keys_perform::m_key_pause [private]

10.27.6.23 m_key_song_mode

unsigned seq64::keys_perform::m_key_song_mode [private]

10.27.6.24 m_key_toggle_jack

unsigned seq64::keys_perform::m_key_toggle_jack [private]

10.27.6.25 m_key_menu_mode

unsigned seq64::keys_perform::m_key_menu_mode [private]

10.27.6.26 m_key_follow_transport

unsigned seq64::keys_perform::m_key_follow_transport [private]

10.27.6.27 m_key_rewind

```
unsigned seq64::keys_perform::m_key_rewind [private]
```

10.27.6.28 m_key_fast_forward

```
unsigned seq64::keys_perform::m_key_fast_forward [private]
```

10.27.6.29 m_key_pointer_position

```
unsigned seq64::keys_perform::m_key_pointer_position [private]
```

10.27.6.30 m_key_toggle_mutes

```
unsigned seq64::keys_perform::m_key_toggle_mutes [private]
```

10.27.6.31 m_key_tap_bpm

```
unsigned seq64::keys_perform::m_key_tap_bpm [private]
```

10.27.6.32 m_key_pattern_edit

```
unsigned seq64::keys_perform::m_key_pattern_edit [private]
```

10.27.6.33 m_key_pattern_shift

```
unsigned seq64::keys_perform::m_key_pattern_shift [private]
```

10.27.6.34 m_key_event_edit

```
unsigned seq64::keys_perform::m_key_event_edit [private]
```

10.27.6.35 m_key_stop

`unsigned seq64::keys_perform::m_key_stop [private]`

10.27.6.36 m_key_song_record

`unsigned seq64::keys_perform::m_key_song_record [private]`

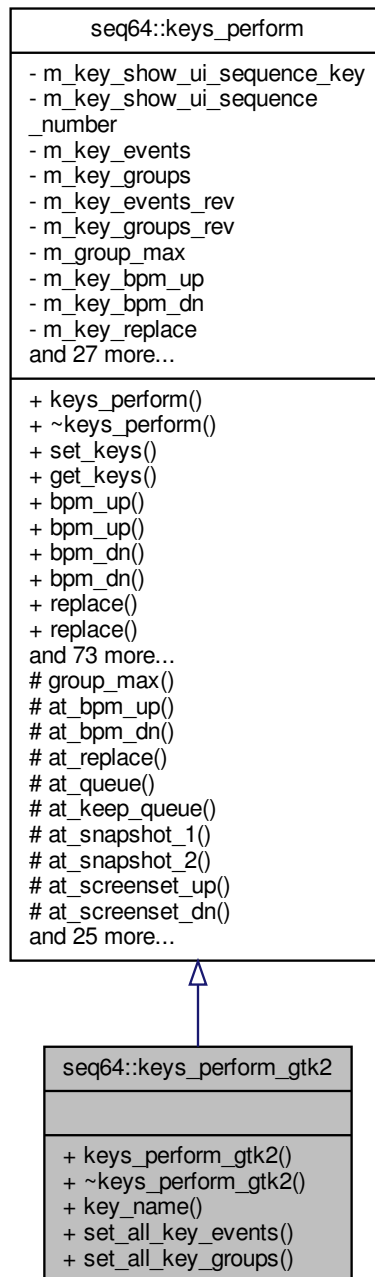
10.27.6.37 m_key_oneshot_queue

`unsigned seq64::keys_perform::m_key_oneshot_queue [private]`

10.28 seq64::keys_perform_gtk2 Class Reference

This class supports the performance mode.

Inheritance diagram for seq64::keys_perform_gtk2:



Public Member Functions

- [keys_perform_gtk2 \(\)](#)
This construction initializes a vast number of member variables, some of them public!
- virtual [~keys_perform_gtk2 \(\)](#)
A rote virtual destructor.
- virtual `std::string` [key_name](#) (unsigned key) const

Converts a key's value to a human-readable string.

- virtual void [set_all_key_events](#) ()

Sets up the keys for arming/unmuting events in the Gtk-2 environment.

- virtual void [set_all_key_groups](#) ()

Sets up the keys for group events in the Gtk-2 environment.

Additional Inherited Members

10.28.1 Detailed Description

It has way too many data members, many of the public. Might be ripe for refactoring.

10.28.2 Constructor & Destructor Documentation

10.28.2.1 `keys_perform_gtk2()`

```
seq64::keys_perform_gtk2::keys_perform_gtk2 ( )
```

10.28.2.2 `~keys_perform_gtk2()`

```
seq64::keys_perform_gtk2::~keys_perform_gtk2 ( ) [virtual]
```

No action.

10.28.3 Member Function Documentation

10.28.3.1 `key_name()`

```
virtual std::string seq64::keys_perform_gtk2::key_name (
    unsigned key ) const [inline], [virtual]
```

Parameters

<i>key</i>	The numerical value of the key.
------------	---------------------------------

Returns

Returns the string version of the key.

Reimplemented from [seq64::keys_perform](#).

10.28.3.2 set_all_key_events()

```
void seq64::keys_perform_gtk2::set_all_key_events ( ) [virtual]
```

The base-class function call makes sure the the related lists are cleared before rebuilding them here.

The main keys are offloaded to the base class now. Qt and Gdk keys overlap a lot.

Reimplemented from [seq64::keys_perform](#).

10.28.3.3 set_all_key_groups()

```
void seq64::keys_perform_gtk2::set_all_key_groups ( ) [virtual]
```

The base-class function call makes sure the the related lists are cleared before rebuilding them here.

The main keys are offloaded to the base class now. Qt and Gdk keys overlap a lot.

Reimplemented from [seq64::keys_perform](#).

10.29 seq64::keys_perform_transfer Struct Reference

Provides a data-transfer structure to make it easier to fill in a [keys_perform](#) object's members using `sscanf()`.

Data Fields

- unsigned [kpt_bpm_up](#)
- unsigned [kpt_bpm_dn](#)
- unsigned [kpt_screenset_up](#)
- unsigned [kpt_screenset_dn](#)
- unsigned [kpt_set_playing_screenset](#)
- unsigned [kpt_group_on](#)
- unsigned [kpt_group_off](#)
- unsigned [kpt_group_learn](#)
- unsigned [kpt_replace](#)
- unsigned [kpt_queue](#)
- unsigned [kpt_keep_queue](#)
- unsigned [kpt_snapshot_1](#)
- unsigned [kpt_snapshot_2](#)
- unsigned [kpt_start](#)

- unsigned [kpt_stop](#)
- bool [kpt_show_ui_sequence_key](#)
- bool [kpt_show_ui_sequence_number](#)
- unsigned [kpt_pattern_edit](#)
- unsigned [kpt_pattern_shift](#)
- unsigned [kpt_event_edit](#)
- unsigned [kpt_tap_bpm](#)
- unsigned [kpt_pause](#)
- unsigned [kpt_song_mode](#)
- unsigned [kpt_toggle_jack](#)
- unsigned [kpt_menu_mode](#)
- unsigned [kpt_follow_transport](#)
- unsigned [kpt_fast_forward](#)
- unsigned [kpt_rewind](#)
- unsigned [kpt_pointer_position](#)
- unsigned [kpt_toggle_mutes](#)
- unsigned [kpt_song_record](#)
- unsigned [kpt_oneshot_queue](#)

10.29.1 Field Documentation

10.29.1.1 kpt_bpm_up

unsigned seq64::keys_perform_transfer::kpt_bpm_up

10.29.1.2 kpt_bpm_dn

unsigned seq64::keys_perform_transfer::kpt_bpm_dn

10.29.1.3 kpt_screenset_up

unsigned seq64::keys_perform_transfer::kpt_screenset_up

10.29.1.4 kpt_screenset_dn

unsigned seq64::keys_perform_transfer::kpt_screenset_dn

10.29.1.5 kpt_set_playing_screenset

unsigned seq64::keys_perform_transfer::kpt_set_playing_screenset

10.29.1.6 kpt_group_on

unsigned seq64::keys_perform_transfer::kpt_group_on

10.29.1.7 kpt_group_off

unsigned seq64::keys_perform_transfer::kpt_group_off

10.29.1.8 kpt_group_learn

unsigned seq64::keys_perform_transfer::kpt_group_learn

10.29.1.9 kpt_replace

unsigned seq64::keys_perform_transfer::kpt_replace

10.29.1.10 kpt_queue

unsigned seq64::keys_perform_transfer::kpt_queue

10.29.1.11 kpt_keep_queue

unsigned seq64::keys_perform_transfer::kpt_keep_queue

10.29.1.12 kpt_snapshot_1

unsigned seq64::keys_perform_transfer::kpt_snapshot_1

10.29.1.13 kpt_snapshot_2

unsigned seq64::keys_perform_transfer::kpt_snapshot_2

10.29.1.14 kpt_start

unsigned seq64::keys_perform_transfer::kpt_start

10.29.1.15 kpt_stop

unsigned seq64::keys_perform_transfer::kpt_stop

10.29.1.16 kpt_show_ui_sequence_key

bool seq64::keys_perform_transfer::kpt_show_ui_sequence_key

10.29.1.17 kpt_show_ui_sequence_number

bool seq64::keys_perform_transfer::kpt_show_ui_sequence_number

10.29.1.18 kpt_pattern_edit

unsigned seq64::keys_perform_transfer::kpt_pattern_edit

10.29.1.19 kpt_pattern_shift

unsigned seq64::keys_perform_transfer::kpt_pattern_shift

10.29.1.20 kpt_event_edit

unsigned seq64::keys_perform_transfer::kpt_event_edit

10.29.1.21 kpt_tap_bpm

unsigned seq64::keys_perform_transfer::kpt_tap_bpm

10.29.1.22 kpt_pause

unsigned seq64::keys_perform_transfer::kpt_pause

10.29.1.23 kpt_song_mode

unsigned seq64::keys_perform_transfer::kpt_song_mode

10.29.1.24 kpt_toggle_jack

unsigned seq64::keys_perform_transfer::kpt_toggle_jack

10.29.1.25 kpt_menu_mode

unsigned seq64::keys_perform_transfer::kpt_menu_mode

10.29.1.26 kpt_follow_transport

unsigned seq64::keys_perform_transfer::kpt_follow_transport

10.29.1.27 kpt_fast_forward

unsigned seq64::keys_perform_transfer::kpt_fast_forward

10.29.1.28 kpt_rewind

unsigned seq64::keys_perform_transfer::kpt_rewind

10.29.1.29 kpt_pointer_position

```
unsigned seq64::keys_perform_transfer::kpt_pointer_position
```

10.29.1.30 kpt_toggle_mutes

```
unsigned seq64::keys_perform_transfer::kpt_toggle_mutes
```

10.29.1.31 kpt_song_record

```
unsigned seq64::keys_perform_transfer::kpt_song_record
```

10.29.1.32 kpt_oneshot_queue

```
unsigned seq64::keys_perform_transfer::kpt_oneshot_queue
```

10.30 seq64::keystroke Class Reference

Encapsulates any practical keystroke.

Public Member Functions

- [keystroke](#) ()
The default constructor for class keystroke.
- [keystroke](#) (unsigned [key](#), bool [press](#)=SEQ64_KEYSTROKE_PRESS, int [modkey](#)=int(SEQ64_NO_MASK))
The principal constructor.
- [keystroke](#) (const [keystroke](#) &[rhs](#))
Provides the rote copy constructor.
- [keystroke](#) & [operator=](#) (const [keystroke](#) &[rhs](#))
Provides the rote principal assignment operator.
- bool [is_press](#) () const
'Getter' function for member m_is_press
- bool [is_letter](#) (unsigned [ch](#)=SEQ64_KEYSTROKE_BAD_VALUE) const
'Getter' function for member m_key to test letters, handles ASCII only.
- bool [is](#) (unsigned [ch](#)) const
Tests the key value to see if it matches the given character exactly (no case-insensitivity).
- bool [is](#) (unsigned [ch1](#), unsigned [ch2](#)) const
Tests the key value to see if it matches the given character exactly (no case-insensitivity).
- bool [is_delete](#) () const
'Getter' function for member m_key to test for a delete-causing key.

- unsigned [key](#) () const
'Getter' function for member m_key
- void [shift_lock](#) ()
If a lower-case letter, a number, or another character on the "main" part of the keyboard, shift the m_key value to upper-case or the character shifted on a standard American keyboard.
- [seq_modifier_t modifier](#) () const
'Getter' function for member m_modifier
- bool [mod_control](#) () const
'Getter' function for member m_modifier tested for Ctrl key.
- bool [mod_control_shift](#) () const
'Getter' function for member m_modifier tested for Ctrl and Shift key.
- bool [mod_super](#) () const
'Getter' function for member m_modifier tested for Mod4/Super/Windows key.

Private Attributes

- bool [m_is_press](#)
Determines if the key was a press or a release.
- unsigned [m_key](#)
The key that was pressed or released.
- [seq_modifier_t m_modifier](#)
The optional modifier value.

10.30.1 Detailed Description

Useful in passing more generic events to non-GUI classes.

10.30.2 Constructor & Destructor Documentation

10.30.2.1 [keystroke\(\)](#) [1/3]

```
seq64::keystroke::keystroke ( )
```

10.30.2.2 [keystroke\(\)](#) [2/3]

```
seq64::keystroke::keystroke (
    unsigned key,
    bool press = SEQ64_KEYSTROKE_PRESS,
    int modkey = int(SEQ64_NO_MASK) )
```

Parameters

<i>key</i>	The keystroke number of the key that was pressed or released.
<i>press</i>	If true, the keystroke action was a press, otherwise it was a release.
<i>modkey</i>	The modifier key combination that was pressed, if any, in the form of a bit-mask, as defined in the gdk_basic_keys module. Common mask values are SEQ64_SHIFT_MASK, SEQ64_CONTROL_MASK, SEQ64_MOD1_MASK, and SEQ64_MOD4_MASK. If no modifier, this value is SEQ64_NO_MASK.

10.30.2.3 **keystroke()** [3/3]

```
seq64::keystroke::keystroke (
    const keystroke & rhs )
```

Parameters

<i>rhs</i>	The object to be copied.
------------	--------------------------

10.30.3 Member Function Documentation

10.30.3.1 **operator=()**

```
keystroke & seq64::keystroke::operator= (
    const keystroke & rhs )
```

Parameters

<i>rhs</i>	The object to be assigned.
------------	----------------------------

Returns

Returns the reference to the current object, for use in assignment chains.

10.30.3.2 **is_press()**

```
bool seq64::keystroke::is_press ( ) const [inline]
```


10.30.3.3 is_letter()

```
bool seq64::keystroke::is_letter (
    unsigned ch = SEQ64_KEYSTROKE_BAD_VALUE ) const
```

Parameters

<i>ch</i>	An optional character to test as an ASCII letter.
-----------	---

Returns

If a character is not provided, true is returned if it is an upper or lower-case letter. Otherwise, true is returned if the `m_key` value matches the character case-insensitively.

Tricky Code**10.30.3.4 is()** [1/2]

```
bool seq64::keystroke::is (
    unsigned ch ) const    [inline]
```

Parameters

<i>ch</i>	The character to be tested.
-----------	-----------------------------

Returns

Returns true if `m_key == ch`.

10.30.3.5 is() [2/2]

```
bool seq64::keystroke::is (
    unsigned ch1,
    unsigned ch2 ) const    [inline]
```

Parameters

<i>ch1</i>	The first character to be tested.
<i>ch2</i>	The second character to be tested.

Returns

Returns true if `m_key == ch1` or `ch2`.

10.30.3.6 is_delete()

```
bool seq64::keystroke::is_delete ( ) const    [inline]
```

10.30.3.7 key()

```
unsigned seq64::keystroke::key ( ) const [inline]
```

10.30.3.8 shift_lock()

```
void seq64::keystroke::shift_lock ( )
```

Currently also assumes the ASCII character set.

There's an oddity here: the shift of '2' is the '@' character, but seq24 seems to have treated it like the "" character. Some others were treated the same:

Key:	1	2	3	4	5	6	7	8	9	0
Shift:	!	@	#	\$	%	^	&	*	()
Seq24:	!	"	#	\$	%	&	'	()	space

This function is meant to avoid using the Caps-Lock when picking a group-learn character in the group-learn mode.

10.30.3.9 modifier()

```
seq_modifier_t seq64::keystroke::modifier ( ) const [inline]
```

10.30.3.10 mod_control()

```
bool seq64::keystroke::mod_control ( ) const [inline]
```

10.30.3.11 mod_control_shift()

```
bool seq64::keystroke::mod_control_shift ( ) const [inline]
```

10.30.3.12 mod_super()

```
bool seq64::keystroke::mod_super ( ) const [inline]
```

10.30.4 Field Documentation

10.30.4.1 m_is_press

```
bool seq64::keystroke::m_is_press [private]
```

See the SEQ64_KEYSTROKE_PRESS and SEQ64_KEYSTROKE_RELEASE readability macros.

10.30.4.2 m_key

```
unsigned seq64::keystroke::m_key [private]
```

Generally, the extended ASCII range (0 to 255) is supported. However, Gtk-2.x/3.x will generally support the full gamut of characters defined in the gdk_basic_keys.h module. We define minimum and maximum range macros for keystrokes that are a bit generous.

10.30.4.3 m_modifier

```
seq_modifier_t seq64::keystroke::m_modifier [private]
```

Note that SEQ64_NO_MASK is our word for 0, meaning "no modifier".

10.31 seq64::lash Class Reference

This class supports LASH operations, if compiled with LASH support (i.e.

Public Member Functions

- [lash](#) ([perform](#) &p, int argc, char **argv)
This constructor calls [lash_extract\(\)](#), using the command-line arguments, if SEQ64_LASH_SUPPORT is enabled.
- void [set_alsa_client_id](#) (int id)
Make ourselves a LASH ALSA client.
- void [start](#) ()
Process any LASH events every 250 msec, which is an arbitrarily chosen interval.
- bool [process_events](#) ()
Process LASH events.

Private Member Functions

- bool [init](#) ()
Initializes LASH support, if enabled.
- void [handle_event](#) (lash_event_t *conf)
Handle a LASH event.
- void [handle_config](#) (lash_config_t *conf)
Handle a LASH configuration item.

Private Attributes

- [perform](#) & [m_perform](#)
A hook into the single perform object in the application.
- [lash_client_t](#) * [m_client](#)
Holds the client "handle" returned by the `lash_init()` function.
- [lash_args_t](#) * [m_lash_args](#)
Holds the command-line arguments used by the `lash_init()` function.
- [bool](#) [m_is_lash_supported](#)
Indicates if LASH support has been compiled into the library.

10.31.1 Detailed Description

SEQ64_LASH_SUPPORT is defined). All of the ifdef skeleton work is done in this class in such a way that any other part of the code can use this class whether or not lash support is actually built in; the functions will just do nothing.

10.31.2 Constructor & Destructor Documentation

10.31.2.1 `lash()`

```
seq64::lash::lash (
    perform & p,
    int argc,
    char ** argv )
```

We fixed the crazy usage of `argc` and `argv` here and in the client code in the `seq24` module.

Parameters

<i>p</i>	The perform object that needs to implement LASH support.
<i>argc</i>	The number of command-line arguments.
<i>argv</i>	The command-line arguments.

10.31.3 Member Function Documentation

10.31.3.1 `set_alsa_client_id()`

```
void seq64::lash::set_alsa_client_id (
    int id )
```

/param *id* The ALSA client ID to be set.

10.31.3.2 start()

```
void seq64::lash::start ( )
```

10.31.3.3 process_events()

```
bool seq64::lash::process_events ( )
```

Returns

Always returns true.

10.31.3.4 init()

```
bool seq64::lash::init ( ) [private]
```

Returns

Returns true if the LASH subsystem was able to be initialized, and a LASH client representative (m_client) was allocated.

10.31.3.5 handle_event()

```
void seq64::lash::handle_event (
    lash_event_t * ev ) [private]
```

Parameters

ev	Provides the event to be handled.
----	-----------------------------------

10.31.3.6 handle_config()

```
void seq64::lash::handle_config (
    lash_config_t * conf ) [private]
```

Currently incomplete.

Parameters

<i>conf</i>	Provides the configuration item to handle.
-------------	--

10.31.4 Field Documentation

10.31.4.1 m_perform

```
perform& seq64::lash::m_perform [private]
```

10.31.4.2 m_client

```
lash_client_t* seq64::lash::m_client [private]
```

10.31.4.3 m_lash_args

```
lash_args_t* seq64::lash::m_lash_args [private]
```

10.31.4.4 m_is_lash_supported

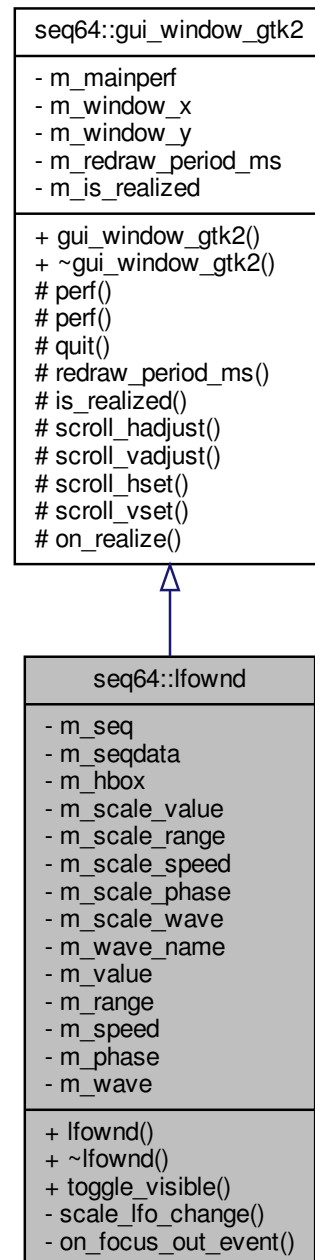
```
bool seq64::lash::m_is_lash_supported [private]
```

Is set to true if SEQ64_LASH_SUPPORT is defined. This variable is not used, but we will keep it around for the possibility of testing LASH support at run time.

10.32 seq64::lfownd Class Reference

One LFO window class.

Inheritance diagram for seq64::lfownd:



Public Member Functions

- `lfownd` (`perform` &p, `sequence` &seq, `seqdata` &sdata)

- *Constructs the LFO window.*
virtual [~lfownd](#) ()
- *Provides a rote destructor.*
void [toggle_visible](#) ()
- *Toggles the visibility of the LFO window.*

Private Member Functions

- void [scale_lfo_change](#) ()
Changes the scaling provided by this window.
- bool [on_focus_out_event](#) (GdkEventFocus *p0)
Undoes the LFO changes if there is undo available.

Private Attributes

- [sequence](#) & [m_seq](#)
The sequence associated with this window.
- [seqdata](#) & [m_seqdata](#)
The seqdata associated with this window.
- Gtk::HBox * [m_hbox](#)
The main horizontal packing box.
- Gtk::VScale * [m_scale_value](#)
Vertical slider for value.
- Gtk::VScale * [m_scale_range](#)
Vertical slider for range.
- Gtk::VScale * [m_scale_speed](#)
Vertical slider for speed.
- Gtk::VScale * [m_scale_phase](#)
Vertical slider for phase.
- Gtk::VScale * [m_scale_wave](#)
Vertical slider for wave type.
- Gtk::Label * [m_wave_name](#)
Human readable name for wave type.
- double [m_value](#)
Value.
- double [m_range](#)
Range.
- double [m_speed](#)
Speed.
- double [m_phase](#)
Phase.
- [wave_type_t](#) [m_wave](#)
Wave type.

Additional Inherited Members

10.32.1 Detailed Description

Personally, it seems a bit of a odd duck to be included in Sequencer64, so we're thinking of a better way to manage the data managed by this window.

10.32.2 Constructor & Destructor Documentation

10.32.2.1 lfownd()

```
seq64::lfownd::lfownd (
    perform & p,
    sequence & seq,
    seqdata & sdata )
```

Parameters

<i>p</i>	The performance object, which holds parameters necessary for manipulating events.
<i>seq</i>	The sequence/pattern that is to be affected by the LFO window. It holds the actual MIDI events being modified.
<i>sdata</i>	The data pane/panel of the pattern editor window representing the sequence. We need to tell it to redraw.

10.32.2.2 ~lfownd()

```
seq64::lfownd::~~lfownd ( ) [virtual]
```

10.32.3 Member Function Documentation

10.32.3.1 toggle_visible()

```
void seq64::lfownd::toggle_visible ( )
```

10.32.3.2 scale_lfo_change()

```
void seq64::lfownd::scale_lfo_change ( ) [private]
```

Changes take place right away in this callback.

10.32.3.3 on_focus_out_event()

```
bool seq64::lfownd::on_focus_out_event (
    GdkEventFocus * p0 ) [private]
```

Returns

Always returns true.

10.32.4 Field Documentation

10.32.4.1 m_seq

`sequence& seq64::lfownd::m_seq [private]`

10.32.4.2 m_seqdata

`seqdata& seq64::lfownd::m_seqdata [private]`

10.32.4.3 m_hbox

`Gtk::HBox* seq64::lfownd::m_hbox [private]`

10.32.4.4 m_scale_value

`Gtk::VScale* seq64::lfownd::m_scale_value [private]`

10.32.4.5 m_scale_range

`Gtk::VScale* seq64::lfownd::m_scale_range [private]`

10.32.4.6 m_scale_speed

`Gtk::VScale* seq64::lfownd::m_scale_speed [private]`

10.32.4.7 m_scale_phase

`Gtk::VScale* seq64::lfownd::m_scale_phase [private]`

10.32.4.8 m_scale_wave

Gtk::VScale* seq64::lfownd::m_scale_wave [private]

10.32.4.9 m_wave_name

Gtk::Label* seq64::lfownd::m_wave_name [private]

10.32.4.10 m_value

double seq64::lfownd::m_value [private]

10.32.4.11 m_range

double seq64::lfownd::m_range [private]

10.32.4.12 m_speed

double seq64::lfownd::m_speed [private]

10.32.4.13 m_phase

double seq64::lfownd::m_phase [private]

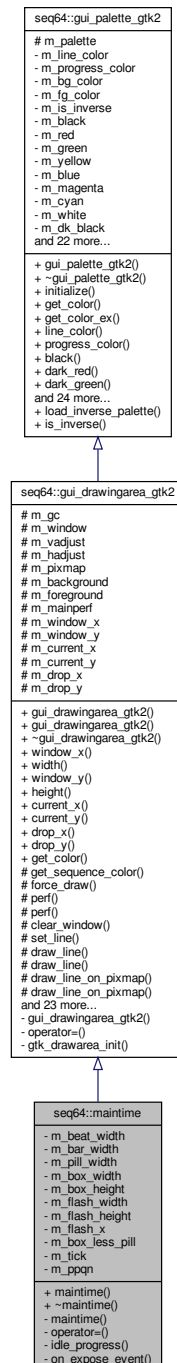
10.32.4.14 m_wave

[wave_type_t](#) seq64::lfownd::m_wave [private]

10.33 seq64::maintime Class Reference

This class provides the drawing of the progress bar at the top of the main window, along with two "pills" that move in time with the beat and measure.

Inheritance diagram for seq64::maintime:



Public Member Functions

- [maintime](#) ([perform](#) &p, int ppqn=SEQ64_USE_DEFAULT_PPQN)

This constructor sets up the colors black, white, and grey, and then allocates them.

- virtual `~maintime()`

Let's provide a do-nothing virtual destructor.

Private Member Functions

- `maintime` (const `maintime` &)
- `maintime` & `operator=` (const `maintime` &)
- int `idle_progress` (`midipulse` ticks)

This function clears the window, sets the foreground to black, draws the "time" window's rectangle, and then draws a rectangle for noting the progress of the beat, and the progress for a bar.

- bool `on_expose_event` (`GdkEventExpose` *ev)

This function merely idles.

Private Attributes

- const int `m_beat_width`

Provides the divisor for ticks to produce a beat value.

- const int `m_bar_width`

Provides the divisor for ticks to produce a bar value.

- const int `m_pill_width`

Provides the width of the pills, little black squares that show the progress of a beat and a bar (measure).

- const int `m_box_width`

The width/length of the rectangle to be drawn inside the maintime window.

- const int `m_box_height`

The height of the rectangle to be drawn inside the maintime window.

- const int `m_flash_width`

The width/length of the flashing rectangle to be drawn inside the maintime window.

- const int `m_flash_height`

The height of the flashing rectangle to be drawn inside the maintime window.

- const int `m_flash_x`

The x value at which a flash should occur.

- const int `m_box_less_pill`

The width/length of the maintime window minus the width of the pill.

- `midipulse` `m_tick`

Saves the tick value for `on_expose_event()`.

- int `m_ppqn`

Provides the active PPQN value.

Friends

- class `mainwnd`

Additional Inherited Members

10.33.1 Detailed Description

We added a lot of members to hold the results of calculations that involve what are essentially constant. This saves CPU time, and maybe a little memory for the code to make those calculations more than once.

10.33.2 Constructor & Destructor Documentation

10.33.2.1 maintime() [1/2]

```
seq64::maintime::maintime (
    const maintime & ) [private]
```

10.33.2.2 maintime() [2/2]

```
seq64::maintime::maintime (
    perform & p,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

In the constructor you can only allocate colors; get_window() would return 0 because the windows has not yet been realized.

10.33.2.3 ~maintime()

```
virtual seq64::maintime::~~maintime ( ) [inline], [virtual]
```

10.33.3 Member Function Documentation

10.33.3.1 operator=()

```
maintime& seq64::maintime::operator= (
    const maintime & ) [private]
```

10.33.3.2 idle_progress()

```
int seq64::maintime::idle_progress (
    midipulse ticks ) [private]
```

Idle hands do the devil's work. We should eventually support some generic coloring for "dark themes". The default coloring is better for "light themes".

Parameters

<i>ticks</i>	Provides the main tick setting. This setting is provided by <code>mainwnd()</code> , in its timer callback.
--------------	---

Returns

Always returns 1 (it used to return "true!").

10.33.3.3 on_expose_event()

```
bool seq64::maintime::on_expose_event (
    GdkEventExpose * ev ) [private]
```

We don't need the `m_tick` member, the function works as well if 0 is passed in. We've removed `m_tick` permanently.

Actually, it might be useful after all, to avoid flickering under JACK transport. Let's put it back for now. (It doesn't help, but we will leave it in, the overhead is small.)

10.33.4 Friends And Related Function Documentation

10.33.4.1 mainwnd

```
friend class mainwnd [friend]
```

10.33.5 Field Documentation

10.33.5.1 m_beat_width

```
const int seq64::maintime::m_beat_width [private]
```

Currently, this value is hardwired to 4, but will eventually be wired up as `usr().midi_beat_width()`.

10.33.5.2 m_bar_width

```
const int seq64::maintime::m_bar_width [private]
```

Currently, this value is hardwired to 16, but will eventually be wired up as `usr().midi_beat_width() * usr().midi_beats_per_bar()`.

10.33.5.3 m_pill_width

```
const int seq64::maintime::m_pill_width [private]
```


10.33.5.4 m_box_width

```
const int seq64::maintime::m_box_width [private]
```

This item absolutely depends on the main window being non-resizable.

10.33.5.5 m_box_height

```
const int seq64::maintime::m_box_height [private]
```

This item absolutely depends on the main window being non-resizable.

10.33.5.6 m_flash_width

```
const int seq64::maintime::m_flash_width [private]
```

Just a bit smaller than m_box_width.

10.33.5.7 m_flash_height

```
const int seq64::maintime::m_flash_height [private]
```

Just a bit smaller than m_box_width.

10.33.5.8 m_flash_x

```
const int seq64::maintime::m_flash_x [private]
```

10.33.5.9 m_box_less_pill

```
const int seq64::maintime::m_box_less_pill [private]
```

10.33.5.10 m_tick

```
midipulse seq64::maintime::m_tick [private]
```

It might actually be useful after all. And the overhead is tiny.

10.33.5.11 m_ppqn

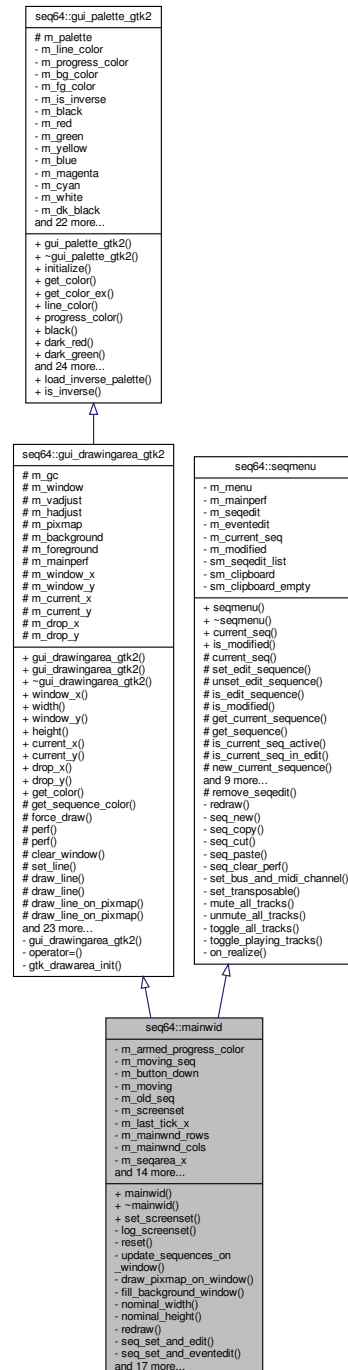
```
int seq64::maintime::m_ppqn [private]
```

While this is effectively a constant for the duration of a tune, it might change as different tunes are loaded.

10.34 seq64::mainwid Class Reference

This class implements the piano roll area of the application.

Inheritance diagram for seq64::mainwid:



Public Member Functions

- [mainwid](#) (perform &p, int ss=0)

This constructor sets all of the members.

- virtual [~mainwid](#) ()

A rote destructor.

- int [set_screenset](#) (int ss)

Set the current screen-set.

Private Member Functions

- void [log_screenset](#) (int ss)

'Setter' function for member m_screenset This function is used for altering the current screen-set displayed by a single mainwid in multi-mainwid mode.

- void [reset](#) ()

This function redraws everything and queues up a redraw operation.

- void [update_sequences_on_window](#) ()

Updates the image of multiple sequencer/pattern slots.

- void [draw_pixmap_on_window](#) ()

This function queues the blit of pixmap to window.

- void [fill_background_window](#) ()

This function updates the background window, clearing it.

- int [nominal_width](#) () const

'Getter' function for member m_mainwid_x

- int [nominal_height](#) () const

'Getter' function for member m_mainwid_y

- virtual void [redraw](#) (int seq)

This virtual function, overridden from the seqmenu base class, draws the the given pattern/sequence again.

- virtual void [seq_set_and_edit](#) (int seqnum)

Calculates the sequence number based on the screenset and then calls the base-class function to bring up the pattern/sequence editor.

- virtual void [seq_set_and_eventedit](#) (int seqnum)

Calculates the sequence number based on the screenset and then calls the base-class function to bring up the event editor.

- void [draw_marker_on_sequence](#) (int seq, int tick)

Does the actual drawing of one pattern/sequence position marker, a vertical progress bar.

- void [update_markers](#) (int ticks)

Draw the cursors (long vertical bars) on each sequence, so that they follow the playing progress of each sequence in the mainwid (Patterns Panel).

- bool [valid_sequence](#) (int seq)

Common-code helper function.

- void [draw_sequence_on_pixmap](#) (int seq)

This function draws a specific pattern/sequence on the pixmap located in the main window of the application, the Patterns Panel.

- void [draw_sequences_on_pixmap](#) ()

This function fills the pixmap with sequences.

- void [draw_sequence_pixmap_on_window](#) (int seq)

This function draws a sequence pixmap in the Patterns Panel.

- int [seq_from_xy](#) (int x, int y)

Translates XY coordinates in the Patterns Panel to a sequence number.

- int [timeout](#) ()

Provides a stock callback, because some kind of callback is needed.

- void [calculate_base_sizes](#) (int seq, int &basex, int &basey)

Provides a way to calculate the base x and y size values for the pattern map.

- void `select_fg_bg_colors` (int seqnum)
Picks the foreground and background colors based on the sequence in edit and the SEQ64_EDIT_SEQUENCE_↔ HIGHLIGHT macro.
- void `on_realize` ()
For this GTK callback, on realization of window, initialize the shiz.
- bool `on_expose_event` (GdkEventExpose *ev)
Implements the GTK expose event callback.
- bool `on_button_press_event` (GdkEventButton *ev)
Handles a press of a mouse button in one of the sequence/pattern slots.
- bool `on_button_release_event` (GdkEventButton *ev)
Handles a release of a mouse button.
- bool `on_motion_notify_event` (GdkEventMotion *p0)
Handle the motion of the mouse if a mouse button is down and in another sequence and if the current sequence is not in edit mode.
- bool `on_focus_in_event` (GdkEventFocus *)
Handles an on-focus event.
- bool `on_focus_out_event` (GdkEventFocus *)
Handles an out-of-focus event.

Private Attributes

- Color `m_armed_progress_color`
Holds the progress color for armed sequences, which have a black background.
- sequence `m_moving_seq`
Holds a partial copy of the sequence we are moving on the patterns panel.
- bool `m_button_down`
Indicates that the mouse button is still down.
- bool `m_moving`
Indicates that we are still in the middle of a drag-and-drop operation.
- int `m_old_seq`
Holds the sequence number of a sequence being drag-and-dropped.
- int `m_screenset`
Indicates the current screenset that is visible.
- long `m_last_tick_x` [c_max_sequence]
Holds the last active tick for each sequence, used in erasing the progress bar.
- const int `m_mainwnd_rows`
These values are assigned to the values given by the constants of similar names in globals.h, and we will make them parameters or user-interface configuration items later.
- const int `m_mainwnd_cols`
Number of columns, unused in settings.
- const int `m_seqarea_x`
Roughly with width of the main window.
- const int `m_seqarea_y`
Roughly with height of the main window.
- const int `m_seqarea_seq_x`
To be determined.
- const int `m_seqarea_seq_y`
To be determined.
- const int `m_mainwid_x`
Horizontal size of the main window grid.
- const int `m_mainwid_y`

- Vertical size of the main window grid.*

 - const int [m_mainwid_border_x](#)
Border offset fudge factor, x.
 - const int [m_mainwid_border_y](#)
Border offset fudge factor, y.
 - const int [m_mainwid_spacing](#)
Spacing, untested setting.
 - const int [m_text_size_x](#)
Text width, varies with font in use.
 - const int [m_text_size_y](#)
Text height, varies with font in use.
 - const int [m_max_sets](#)
Maximum number of sets, used all over.
 - const int [m_screenset_slots](#)
Provides a convenience variable for avoiding multiplications.
 - int [m_screenset_offset](#)
Provides a convenience variable for avoiding multiplications.
 - const int [m_progress_height](#)
Provides the height of the progress bar, to save calculations and for consistency between drawing and erasing the progress bar.

Friends

- class [mainwnd](#)
- void [update_mainwid_sequences](#) ()
This global function in the [seq64](#) namespace calls `mainwnd :: update_sequences_on_window()`, if the global `mainwid` object exists.

Additional Inherited Members

10.34.1 Detailed Description

It inherits from [gui_drawingarea_gtk2](#) to support the font, color, and other GUI functionality, and from [seqmenu](#) to support the right-click Edit/New/Cut right-click menu. The friend class and function are for updating the current sequence and for control via the `mainwnd` object.

10.34.2 Constructor & Destructor Documentation

10.34.2.1 `mainwid()`

```
seq64::mainwid::mainwid (
    perform & p,
    int ss = 0 )
```

And it asks for its size from [usr\(\)](#).`mainwid_width()` and [usr\(\)](#).`mainwid_height` functions. It adds GDK masks for button presses, releases, motion, key presses, and focus changes. Also logs a self-referential singleton pointer to use for the current-edit highlighting support.

Parameters

<i>p</i>	Provides the reference to the all-important perform object.
<i>ss</i>	Indicates the screen we start out at. This is really only useful if SEQ64_MULTI_MAINWND is defined, but doesn't hurt much to have it as a parameter here, and it could be a good feature independent of multi-mainwid support. The default value is 0.

10.34.2.2 ~mainwid()

```
seq64::mainwid::~~mainwid ( ) [virtual]
```

10.34.3 Member Function Documentation

10.34.3.1 set_screenset()

```
int seq64::mainwid::set_screenset (
    int ss )
```

Note that `m_screenset_slots = m_mainwnd_rows * m_mainwnd_cols` and this will also match [perform::m_seqs_↵](#) [in_set](#).

This function calls [perform::set_screenset\(\)](#), which recapitulates the old code above completely, whereas `perform::set_offset()` recapitulates only the line of code immediately above it. However, note that there is a back-and-forth between setting the screenset via `perform` (using MIDI control) versus the GUI in the `mainwnd` class. Probably useful to add a default boolean to prevent circular manipulation.

Parameters

<i>ss</i>	Provides the screen-set number to set.
<i>setperf</i>	If true, then also call perform::set_screenset() , even if multi-wid mode is not in force. Defaults to false. It might be better if it defaults to true.

Returns

Returns the (new) value of `m_screenset` so that the main window can set the set-number in the Set spinner.

10.34.3.2 log_screenset()

```
void seq64::mainwid::log_screenset (
    int ss ) [private]
```

10.34.3.3 reset()

```
void seq64::mainwid::reset ( ) [inline], [private]
```

10.34.3.4 update_sequences_on_window()

```
void seq64::mainwid::update_sequences_on_window ( ) [inline], [private]
```

Used by the friend class mainwnd, but also useful for our new feature to fully highlight the current sequence. Calls [reset\(\)](#) if SEQ64_EDIT_SEQUENCE_HIGHLIGHT is defined.

10.34.3.5 draw_pixmap_on_window()

```
void seq64::mainwid::draw_pixmap_on_window ( ) [inline], [private]
```

10.34.3.6 fill_background_window()

```
void seq64::mainwid::fill_background_window ( ) [inline], [private]
```

10.34.3.7 nominal_width()

```
int seq64::mainwid::nominal_width ( ) const [inline], [private]
```

10.34.3.8 nominal_height()

```
int seq64::mainwid::nominal_height ( ) const [inline], [private]
```

10.34.3.9 redraw()

```
void seq64::mainwid::redraw (
    int seqnum ) [private], [virtual]
```

Parameters

<i>seqnum</i>	Provides the number of the sequence to draw.
---------------	--

Implements [seq64::seqmenu](#).

10.34.3.10 seq_set_and_edit()

```
void seq64::mainwid::seq_set_and_edit (
    int seqnum ) [private], [virtual]
```

Used with the '=' key selection, by default.

Reimplemented from [seq64::seqmenu](#).

10.34.3.11 seq_set_and_eventedit()

```
void seq64::mainwid::seq_set_and_eventedit (
    int seqnum ) [private], [virtual]
```

Used with the '-' key selection, by default.

Reimplemented from [seq64::seqmenu](#).

10.34.3.12 draw_marker_on_sequence()

```
void seq64::mainwid::draw_marker_on_sequence (
    int seqnum,
    int tick ) [private]
```

If the sequence has no events, this function doesn't bother drawing a position marker.

Note that, when Sequencer64 first comes up, and [perform::is_dirty_main\(\)](#) is called, no sequences exist yet. Also, currently the [redraw\(\)](#) is hit when [seq_edit\(\)](#) is called, but not when [seq_event_edit\(\)](#) is called, which makes the latter not paint the in-edit highlight colors (if enabled). Why?

Parameters

<i>seqnum</i>	Provides the number of the sequence to draw.
<i>tick</i>	Provides the location to draw the marker. If pause support is compiled in (i.e. no <code>--disable-pause</code> in the configuration), then this parameter is ignored, and is replaced by the sequences' <code>get_lask_tick()</code> value. This causes correct stop/pause/play progress-bar behavior in each pattern slot. Note: This is now independent of the <code>--disable-pause</code> option!

10.34.3.13 update_markers()

```
void seq64::mainwid::update_markers (
    int tick ) [private]
```

Parameters

<i>tick</i>	Starting point for drawing the markers.
-------------	---

10.34.3.14 valid_sequence()

```
bool seq64::mainwid::valid_sequence (
    int seqnum ) [private]
```

Parameters

<i>seqnum</i>	Provides the number of the sequence to validate.
---------------	--

Returns

Returns true if the sequence number is valid for the current `m_screenset` value.

10.34.3.15 draw_sequence_on_pixmap()

```
void seq64::mainwid::draw_sequence_on_pixmap (
    int seqnum ) [private]
```

The sequence is drawn only if it is in the current screen set (indicated by `m_screenset`). Also, we ignore the sequence if it does not exist.

Note

If only the main window is up, then the sequences just play (muted by default) – the progress bars move in each pattern. Gaps in the sequence in the Song (performance) Editor don't change the appearance of the patterns if only the main window is up. But, if the Song Editor window is up, and the song is started using the controls in the Song Editor, then the active patterns are black while playing, and white when gaps in the sequence are encountered. The muting status in the main window is ignored. The muting in the Song (performance) windows is in force. This setup holds for ALSA, but not for JACK transport.

Parameters

<i>seqnum</i>	Provides the number of the sequence slot that needs to be drawn. It is checked for validity before usage.
---------------	---

10.34.3.16 draw_sequences_on_pixmap()

```
void seq64::mainwid::draw_sequences_on_pixmap ( ) [private]
```

Please note that [draw_sequence_on_pixmap\(\)](#) also draws the empty slots of inactive sequences, so we cannot take shortcuts here.

10.34.3.17 draw_sequence_pixmap_on_window()

```
void seq64::mainwid::draw_sequence_pixmap_on_window (
    int seqnum ) [private]
```

The sequence is drawn only if it is in the current screen set (indicated by `m_screenset`). This function is used when dragging a pattern from one pattern-slot to another pattern-slot.

We have to add 1 pixel to the y height in order to avoid leaving behind a line at the bottom of an empty pattern-slot.

Parameters

<i>seqnum</i>	Provides the number of the sequence to draw.
---------------	--

10.34.3.18 seq_from_xy()

```
int seq64::mainwid::seq_from_xy (
    int x,
    int y ) [private]
```

Parameters

<i>x</i>	Provides the x coordinate.
<i>y</i>	Provides the y coordinate.

Returns

Returns -1 if the sequence number cannot be calculated.

10.34.3.19 timeout()

```
int seq64::mainwid::timeout ( ) [private]
```

Returns

Always returns true.

10.34.3.20 calculate_base_sizes()

```
void seq64::mainwid::calculate_base_sizes (
    int seqnum,
    int & basex,
    int & basey ) [private]
```

The values are returned as side-effects.

Parameters

	<i>seqnum</i>	Provides the number of the sequence to calculate.
out	<i>basex</i>	A return parameter for the x coordinate of the base size.
out	<i>basey</i>	A return parameter for the y coordinate of the base size.

10.34.3.21 select_fg_bg_colors()

```
void seq64::mainwid::select_fg_bg_colors (
    int seqnum ) [private]
```

10.34.3.22 on_realize()

```
void seq64::mainwid::on_realize ( ) [private]
```

It allocates any additional resources that weren't initialized in the constructor.

This function used to call [font::init\(\)](#), and was the only place where the [font::init\(\)](#) function was called. The [init\(\)](#) function gets a color-map from the window. We need a more fool-proof way to do this!

10.34.3.23 on_expose_event()

```
bool seq64::mainwid::on_expose_event (
    GdkEventExpose * ev ) [private]
```

Parameters

<i>ev</i>	The expose event.
-----------	-------------------

Returns

Always returns true.

10.34.3.24 on_button_press_event()

```
bool seq64::mainwid::on_button_press_event (
    GdkEventButton * ev ) [private]
```

If the press is a single left-click, and no Ctrl key is pressed, then this function grabs the focus, calculates the pattern/sequence over which the button press occurred, and sets the m_button_down flag if it is over a pattern. In the release event callback, this then causes the sequence arming/muting to be toggled.

If the press is a single Ctrl-left-click, this function brings up the New or Edit menu. The New menu is brought up if the grid slot is empty, and the Edit menu otherwise. Another way to bring up the same functionality is described in the next paragraph.

If the press is a double-click, it first acts just like two single-clicks (which might confuse the user at first, because it toggles the mute state twice). Then it brings up the Edit menu for the sequence. This new behavior is closer to what users have come to expect from a double-click. I miss the double-click when running seq24.

We also try to handle a Ctrl-double-click as a signal to do an event edit, instead of a sequence edit. The event editor provides a way to look at all events in detail, without having to select the type of event to see. However, this doesn't work, the event is treated like a ctrl-single-click. And we use the Alt key to enable window movement or resizing in our window manager, so that's out.

Parameters

ev	Provides the parameters of the button event.
----	--

Returns

Always returns true.

10.34.3.25 on_button_release_event()

```
bool seq64::mainwid::on_button_release_event (
    GdkEventButton * ev ) [private]
```

This event is a lot more complex than a press. The left button toggles playback status. The right button brings up a popup menu. If the slot is empty, then a "New" popup is presented, otherwise an "Edit" and selection popup is presented.

Also now implements the new "toggle all other patterns" action, initiated via Shift-Left-Click.

Parameters

ev	Provides the parameters of the button event.
----	--

Returns

Always returns true.

Tried disabling the setting of the current sequence; it completely disables drag-n-drop. But leaving it in removes the current-sequence highlighting, which otherwise is fine. So we do it only if moving a pattern (drag-and-drop).

10.34.3.26 on_motion_notify_event()

```
bool seq64::mainwid::on_motion_notify_event (
    GdkEventMotion * ev ) [private]
```

This function moves the selected pattern to another pattern slot. The [perform::delete_sequence\(\)](#) function sets the perform modification flag.

Parameters

<i>ev</i>	Provides the parameters of the button event.
-----------	--

Returns

Always returns true.

10.34.3.27 on_focus_in_event()

```
bool seq64::mainwid::on_focus_in_event (
    GdkEventFocus * ) [private]
```

Just sets the Gtk::HAS_FOCUS flag.

Returns

Always returns false.

10.34.3.28 on_focus_out_event()

```
bool seq64::mainwid::on_focus_out_event (
    GdkEventFocus * ) [private]
```

Just unsets the Gtk::HAS_FOCUS flag.

Returns

Always returns false.

10.34.4 Friends And Related Function Documentation

10.34.4.1 mainwnd

```
friend class mainwnd [friend]
```

10.34.4.2 update_mainwid_sequences

```
void update_mainwid_sequences ( ) [friend]
```

It is used by other objects that can modify the currently-edited sequence shown in the mainwid (main window).

10.34.5 Field Documentation

10.34.5.1 m_armed_progress_color

```
Color seq64::mainwid::m_armed_progress_color [private]
```

If the progress color is `black()`, we want to change it to white for unmuted patterns.

10.34.5.2 m_moving_seq

```
sequence seq64::mainwid::m_moving_seq [private]
```

The assignment is made by `sequence::partial_copy()`, which behaves like the legacy `seq24` code.

10.34.5.3 m_button_down

```
bool seq64::mainwid::m_button_down [private]
```

Used in the drag-and-drop functionality.

10.34.5.4 m_moving

```
bool seq64::mainwid::m_moving [private]
```

10.34.5.5 m_old_seq

```
int seq64::mainwid::m_old_seq [private]
```

10.34.5.6 m_screenset

```
int seq64::mainwid::m_screenset [private]
```

10.34.5.7 m_last_tick_x

```
long seq64::mainwid::m_last_tick_x[c_max_sequence] [private]
```

10.34.5.8 m_mainwnd_rows

```
const int seq64::mainwid::m_mainwnd_rows [private]
```

Some of them already have counterparts in the [user_settings](#) class. Number of rows, unused part of settings.

10.34.5.9 m_mainwnd_cols

```
const int seq64::mainwid::m_mainwnd_cols [private]
```

10.34.5.10 m_seqarea_x

```
const int seq64::mainwid::m_seqarea_x [private]
```

10.34.5.11 m_seqarea_y

```
const int seq64::mainwid::m_seqarea_y [private]
```

10.34.5.12 m_seqarea_seq_x

```
const int seq64::mainwid::m_seqarea_seq_x [private]
```

10.34.5.13 m_seqarea_seq_y

```
const int seq64::mainwid::m_seqarea_seq_y [private]
```

10.34.5.14 m_mainwid_x

```
const int seq64::mainwid::m_mainwid_x [private]
```

10.34.5.15 m_mainwid_y

```
const int seq64::mainwid::m_mainwid_y [private]
```

10.34.5.16 m_mainwid_border_x

```
const int seq64::mainwid::m_mainwid_border_x [private]
```

10.34.5.17 m_mainwid_border_y

```
const int seq64::mainwid::m_mainwid_border_y [private]
```

10.34.5.18 m_mainwid_spacing

```
const int seq64::mainwid::m_mainwid_spacing [private]
```

10.34.5.19 m_text_size_x

```
const int seq64::mainwid::m_text_size_x [private]
```

10.34.5.20 m_text_size_y

```
const int seq64::mainwid::m_text_size_y [private]
```

10.34.5.21 m_max_sets

```
const int seq64::mainwid::m_max_sets [private]
```


10.34.5.22 m_screenset_slots

```
const int seq64::mainwnd::m_screenset_slots [private]
```

It is equal to `m_mainwnd_rows * m_mainwnd_cols`.

10.34.5.23 m_screenset_offset

```
int seq64::mainwnd::m_screenset_offset [private]
```

It is equal to `m_screenset_slots * m_screenset`.

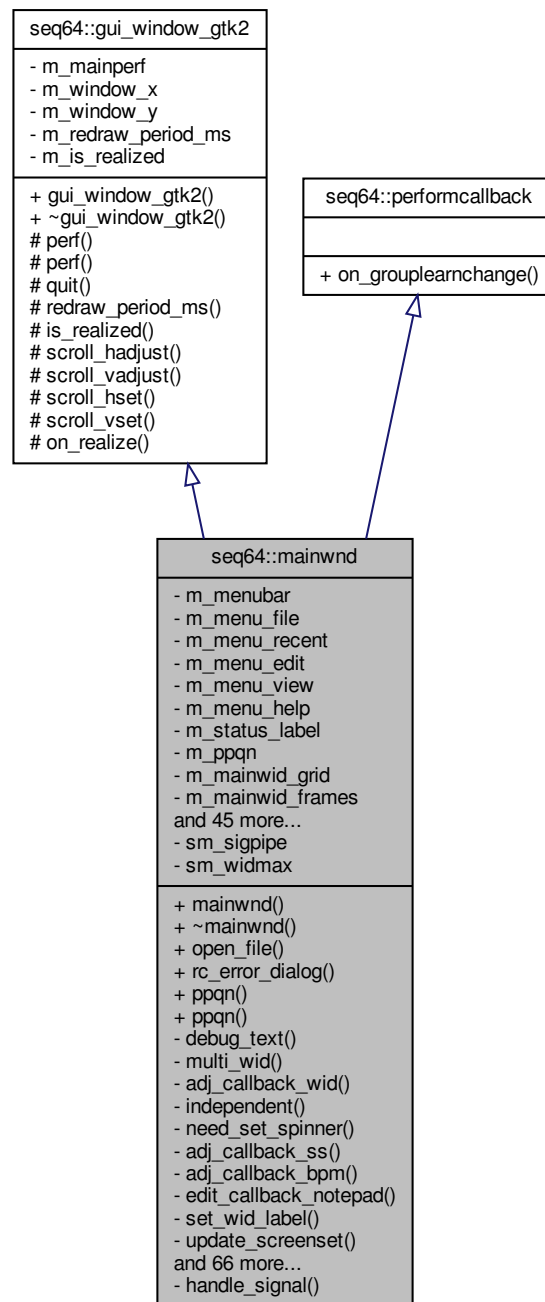
10.34.5.24 m_progress_height

```
const int seq64::mainwnd::m_progress_height [private]
```

10.35 seq64::mainwnd Class Reference

This class implements the functionality of the main window of the application, except for the Patterns Panel functionality, which is implemented in the mainwid class.

Inheritance diagram for seq64::mainwnd:



Public Member Functions

- [mainwnd](#) ([perform](#) &p, bool allowperf2=true, int [ppqn](#)=SEQ64_USE_DEFAULT_PPQN, int mainwid_rows=1, int mainwid_cols=1, bool mainwid_indep=false)

The constructor the main window of the application.

- virtual [~mainwnd](#) ()

This destructor must explicitly delete some allocated resources.

- void [open_file](#) (const std::string &filename)
Opens and parses (reads) a MIDI file or a WRK file.
- void [rc_error_dialog](#) (const std::string &message)
Tells the user that the "rc" file is erroneous.
- int [ppqn](#) () const
'Getter' function for member m_ppqn
- void [ppqn](#) (int ppqn)
'Setter' function for member m_ppqn We can't set the PPQN value when the mainwnd is created, we have to do it later, using this function.

Private Types

- enum [SaveOption](#) {
 [FILE_SAVE_AS_NORMAL](#),
 [FILE_SAVE_AS_EXPORT_SONG](#),
 [FILE_SAVE_AS_EXPORT_MIDI](#) }
Instead of having two save options, we now have three.

Private Member Functions

- void [debug_text](#) (const std::string &tag, int value)
Overwrites the text in the set-notes field, for debugging purposes only.
- bool [multi_wid](#) () const
'Getter' function for member m_mainwid_count > 1
- void [adj_callback_wid](#) (int mainwid_block)
This function is the callback for adjusting the screen-set value of a particular mainwid.
- bool [independent](#) () const
'Getter' function for member m_mainwid_independent
- bool [need_set_spinner](#) (int block) const
'Getter' function for member m_mainwid_independent
- void [adj_callback_ss](#) ()
This function is the callback for adjusting the active screen-set value.
- void [adj_callback_bpm](#) ()
This function is the callback for adjusting the BPM value.
- void [edit_callback_notepad](#) ()
A callback function for handling an edit to the screen-set notepad.
- void [set_wid_label](#) (int ss, int block=0)
Sets the frame label for the given mainwid, based on the set number.
- void [update_screenset](#) ()
New function to consolidate screen-set handling and avoid contention between various sources of screen-set changing by letting the timer callback detect screen-set changes that would affect the user-interface.
- void [update_markers](#) (midipulse tick)
Updates the markers on one (or more, if multi-mainwid is enabled) mainwid objects.
- void [reset](#) ()
Resets one (or more, if multi-mainwid is enabled) mainwid objects.
- void [reset_window](#) ()
Resets the following items:
- void [set_play_image](#) (bool isrunning)
Changes the image used for the pause/play button.
- void [set_songlive_image](#) (bool issong)

- Changes the image used for the song/live mode button.*

 - void [start_playing](#) ()
Starts playing of the song.
 - void [pause_playing](#) ()
Pauses the playing of the song, leaving the progress bar where it stopped.
 - void [stop_playing](#) ()
Stops the playing of the song.
 - void [toggle_playing](#) ()
Reverses the state of playback.
 - bool [timer_callback](#) ()
This function is the GTK timer callback, used to draw our current time and BPM on _events (the main window).
 - int [set_screenset](#) (int screenset)
New function to consolidate screen-set handling.
 - void [tempo_log](#) ()
Logs the current tempo/tick value as a Set Tempo event.
 - void [toggle_tempo_record](#) ()
Toggles the recording of the tempo.
 - void [toggle_time_format](#) ()
Toggles the recording of the live song control done by the musician.
 - void [queue_it](#) ()
Implements the keep-queue button.
 - void [set_song_record](#) ()
Sets the song-recording status.
 - void [toggle_song_record](#) ()
Toggles the recording of the live song control done by the musician.
 - void [toggle_song_snap](#) ()
Toggles the recording of the live song control done by the musician.
 - void [set_song_playback](#) (bool playsong)
Sets the playback mode (Live vs Song).
 - void [song_record_snap](#) (bool snap)
 - void [panic](#) ()
Pushes the panic button.
 - void [learn_toggle](#) ()
Toggle the group-learn status.
 - void [open_performance_edit](#) ()
Opens the Performance Editor (Song Editor).
 - void [open_performance_edit_2](#) ()
Opens the second Performance Editor (Song Editor).
 - void [enregister_perfedits](#) ()
This function brings together the two perfedit objects, so that they can tell each other when to queue up a draw operation.
 - void [sequence_key](#) (int seq)
Use the sequence key to toggle the playing of an active pattern in the current screen-set.
 - int [spinner_max](#) () const
Returns the maximum value we can allow for a spinner.
 - void [apply_song_transpose](#) ()
Apply full song transposition, if enabled.
 - void [clear_mute_groups](#) ()
Clear all mute-group settings.
 - void [reload_mute_groups](#) ()
Reload all mute-group settings from the "rc" file.

- void [update_window_title](#) ()
Updates the title shown in the title bar of the window.
- void [update_recent_files_menu](#) ()
- void [load_recent_file](#) (int index)
Looks up the desired recent MIDI file and opens it.
- void [toLower](#) (std::string &)
Converts a string to lower-case letters.
- void [file_new](#) ()
A callback function for the File / New menu entry.
- void [file_open](#) ()
A callback function for the File / Open menu entry.
- void [file_save](#) ()
A callback function for the File / Save menu entry.
- void [set_song_mute](#) (perform::mute_op_t op)
Sets the song-mute mode.
- int [wid_box_to_slot](#) (int col, int row) const
- bool [wid_slot_to_box](#) (int slot, int &col, int &row) const
- void [file_import_dialog](#) ()
Presents a file dialog to import a MIDI file.
- void [options_dialog](#) ()
Opens the File / Options dialog.
- void [jack_dialog](#) ()
Opens the File / Options dialog to show only the JACK page.
- void [about_dialog](#) ()
Presents a Help / About dialog.
- void [build_info_dialog](#) ()
Presents a Help / Build Info dialog.
- int [query_save_changes](#) ()
Queries the user to save the changes made while the application was running.
- void [new_open_error_dialog](#) ()
Tells the user to close all the edit windows first.
- void [file_save_as](#) (SaveOption option=FILE_SAVE_AS_NORMAL)
A callback function for the File / Save As menu entry.
- void [file_exit](#) ()
A callback function for the File / Exit menu entry.
- void [new_file](#) ()
Actually does the work of setting up for a new file.
- bool [save_file](#) ()
Saves the current state in a MIDI file.
- void [choose_file](#) ()
Creates a file-chooser dialog.
- bool [is_save](#) ()
If the data is modified, then the user is queried, and the file is save if okayed.
- bool [install_signal_handlers](#) ()
Installs the signal handlers and pipe code.
- bool [signal_action](#) (Glib::IOCondition condition)
Handles saving or exiting actions when signalled.
- bool [edit_field_has_focus](#) () const
Check if one of the edit fields (BPM spinbutton, screenset spinbutton, or the Name field) has focus.
- void [populate_menu_file](#) ()
Populates the File menu: File menu items; their accelerator keys; and their hot keys.

- void [populate_menu_edit](#) ()
Populates the Edit menu: Edit menu items; their accelerator keys; and their hot keys.
- void [populate_menu_help](#) ()
Populates the Help menu.
- void [populate_menu_view](#) ()
Populates the View menu: View menu items and their hot keys.
- void [set_status_text](#) (const std::string &text)
Sets the text on the new status label.
- bool [on_delete_event](#) (GdkEventAny *ev)
This callback function handles the user requesting that the main window be closed.
- bool [on_key_press_event](#) (GdkEventKey *ev)
Handles a key press event.
- bool [on_key_release_event](#) (GdkEventKey *ev)
Handles a key release event.
- void [on_realize](#) ()
We are trying to work around an apparent Gtk+ bug (which occurs on my 64-bit Debian Sid laptop, but not on my 32-bit Debian Jessie laptop) that causes Sequencer64 to freeze, emitting [Gtk](#) errors, if one tries to access the main menu via Alt-F, Alt-E, etc.
- virtual void [on_grouplearnchange](#) (bool state)
Notification handler for learn mode toggle.

Static Private Member Functions

- static void [handle_signal](#) (int sig)
This function is the handler for system signals (SIGUSR1, SIGINT...) It writes a message to the pipe and leaves as soon as possible.

Private Attributes

- Gtk::MenuBar * [m_menubar](#)
Theses objects support the menu and its sub-menus.
- Gtk::Menu * [m_menu_file](#)
The File menu entry.
- Gtk::Menu * [m_menu_recent](#)
File/Recent menu popup.
- Gtk::Menu * [m_menu_edit](#)
The (new) Edit menu entry.
- Gtk::Menu * [m_menu_view](#)
The View menu entry.
- Gtk::Menu * [m_menu_help](#)
The Help menu entry.
- Gtk::Label * [m_status_label](#)
Seamless status label next to the "ALSA/JACK/Native" button.
- int [m_ppqn](#)
Saves the PPQN value obtained from the MIDI file (or the default value, the global ppqn, if SEQ64_USE_DEFAULT_PPQN was specified in reading the MIDI file.
- Gtk::Table * [m_mainwid_grid](#)
Provides a place in which to array multiple mainwid objects.
- Gtk::Frame * [m_mainwid_frames](#) [SEQ64_MAINWIDS_MAX]
Holds from 1 x 1 to up to 2 x 3 (1 to 6) pointers for Frame objects.

- `Gtk::Adjustment * m_mainwid_adjustors` [SEQ64_MAINWIDS_MAX]
Holds from 1 x 1 to up to 2 x 3 (1 to 6) pointers to spinner adjustment objects.
- `Gtk::SpinButton * m_mainwid_spinners` [SEQ64_MAINWIDS_MAX]
Holds from 1 x 1 to up to 2 x 3 (1 to 6) pointers to spinner objects.
- `mainwid * m_mainwid_blocks` [SEQ64_MAINWIDS_MAX]
Holds from 1 x 1 to up to 2 x 3 (1 to 6) pointers for mainwid objects.
- `int m_mainwid_rows`
The number of mainwids vertically.
- `int m_mainwid_columns`
The number of mainwids horizontally.
- `int m_mainwid_count`
The number of mainwids.
- `bool m_mainwid_independent`
Indicates if we want to control the set-number of each mainwid separately or not.
- `mainwid * m_main_wid`
The biggest sub-component of mainwnd is the Patterns Panel, which the mainwid implements.
- `Gtk::Adjustment * m_adjust_ss`
The spin/adjustment controls for the screenset value.
- `Gtk::SpinButton * m_spinbutton_ss`
Screenset adjustment.
- `int m_current_screenset`
Saves the active screenset number so that we can better detect changes from both the perform object and the screenset spinbutton, which updates the perform object.
- `maintime * m_main_time`
Is this the bar at the top that shows moving squares, also known as "pills"? Why yes, it is.
- `perfeddit * m_perf_edit`
A pointer to the first song/performance editor.
- `perfeddit * m_perf_edit_2`
A pointer to an optional second song/performance editor.
- `options * m_options`
A pointer to the program options.
- `Gdk::Cursor m_main_cursor`
Mouse cursor?
- `Gtk::Image * m_image_play`
Provides a pointer to hold the images for the pause/play button.
- `Gtk::Button * m_button_panic`
This button is the panic button, which is adapted from Oli Kester's kepler34 project.
- `Gtk::Button * m_button_learn`
This button is the learn button, otherwise known as the "L" button.
- `Gtk::Button * m_button_stop`
Implements the red square stop button.
- `Gtk::Button * m_button_play`
Implements the green triangle play button.
- `Gtk::Button * m_button_tempo_log`
Implements the new magenta tempo-log button.
- `Gtk::ToggleButton * m_button_tempo_record`
Implements the new tempo-record button.
- `bool m_is_tempo_recording`
Indicates if tempo recording is active.
- `Gtk::Button * m_button_perfeddit`
The button for bringing up the Song Editor (Performance Editor).

- `Gtk::Button * m_button_jack`
Sets and indicates the current mode of Sequencer64: JACK, Master, and ALSA.
- `Gtk::ToggleButton * m_button_song_record`
Implements Oli Kester's Kepler34 Song-recording feature.
- `Gtk::ToggleButton * m_button_song_snap`
Implements Oli Kester's Kepler34 Song-recording snap feature.
- `bool m_is_song_recording`
Indicates if song recording is active.
- `bool m_is_snap_recording`
Indicates if song-recording snap is active.
- `Gtk::Label * m_tick_time`
This new item shows the current time into the song performance.
- `Gtk::Button * m_button_time_type`
This button will toggle the m_tick_time_as_bbt member.
- `bool m_tick_time_as_bbt`
Indicates whether to show the time as bar:beats:ticks or as hours:minutes:seconds.
- `Gtk::Adjustment * m_adjust_bpm`
The spin/adjustment controls for the BPM (beats-per-minute) value.
- `Gtk::SpinButton * m_spinbutton_bpm`
BPM spin-button object.
- `Gtk::ToggleButton * m_button_queue`
It seems convenient to have a button that can show that status of keep queue.
- `Gtk::Adjustment * m_adjust_load_offset`
The spin/adjustment controls for the load offset value.
- `Gtk::SpinButton * m_spinbutton_load_offset`
Spin button for import.
- `Gtk::Entry * m_entry_notes`
This item provides user-interface access to the screenset notepad editor.
- `bool m_is_running`
Holds the current status of running, for use in display the play versus pause icon.
- `sigc::connection m_timeout_connect`
Provides a timeout handler.
- `bool m_menu_mode`
Indicates if the menu bar is to be greyed out or not.
- `bool m_call_seq_edit`
Indicates that this object is in a mode where the usual mute/unmute keystroke will instead bring up the pattern slot for editing.
- `int m_call_seq_shift`
A new flag to indicate if the next pattern hot-key will reach into the extended part of the set.
- `bool m_call_seq_eventedit`
Indicates that this object is in a mode where the usual mute/unmute keystroke will instead bring up the pattern slot for event-editing.

Static Private Attributes

- `static int sm_sigpipe [2]`
This small array holds the "handles" for the pipes need to intercept the system signals SIGINT and SIGUSR1, so that the application shuts down gracefully when aborted.
- `static const int sm_widmax`
We iterate through multi-mainwids using a linear array and checking for null pointers.

Additional Inherited Members

10.35.1 Member Enumeration Documentation

10.35.1.1 SaveOption

```
enum seq64::mainwnd::SaveOption [private]
```

Enumerator

FILE_SAVE_AS_NORMAL	
FILE_SAVE_AS_EXPORT_SONG	
FILE_SAVE_AS_EXPORT_MIDI	

10.35.2 Constructor & Destructor Documentation

10.35.2.1 mainwnd()

```
seq64::mainwnd::mainwnd (
    perform & p,
    bool allowperf2 = true,
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    int mainwid_rows = 1,
    int mainwid_cols = 1,
    bool mainwid_indep = false )
```

This constructor is way too large; it would be nicer to provide a number of well-named initialization functions.

Parameters

<i>p</i>	Refers to the main performance object.
<i>allowperf2</i>	Indicates if a second perfedit window should be created. This is currently a run-time option, selectable in the "user" configuration file.
<i>ppqn</i>	An optional PPQN value to use in the song.
<i>mainwid_rows</i>	The number of rows of mainwids to create vertically. The default value is one. Used only if SEQ64_MULTI_MAINWID is defined.
<i>mainwid_cols</i>	The number of columns of mainwids to create horizontally. The default value is one. Used only if SEQ64_MULTI_MAINWID is defined.
<i>mainwid_indep</i>	If true, indicates that the mainwids in multi-wid mode can have their set-numbers controller independently.

Todo Offload most of the work into an initialization function like options does; make the perform parameter a reference.

Top panel items, including the logo (updated for the new version of this application) and the "timeline" progress bar.

10.35.2.2 `~mainwnd()`

```
seq64::mainwnd::~~mainwnd ( ) [virtual]
```

10.35.3 Member Function Documentation

10.35.3.1 `open_file()`

```
void seq64::mainwnd::open_file (
    const std::string & fn )
```

We leave the `ppqn` parameter set to the `SEQ64_USE_DEFAULT` for now, to preserve the legacy behavior of using the global `ppqn`, and scaling the running time against the PPQN read from the MIDI file. Later, we can provide a value like 0, that will certainly be changed by reading the MIDI file.

We don't need to specify the "oldformat" or "global sequence" parameters of the `midifile` constructor when reading the MIDI file, since reading handles both the old and new formats, dealing with new constructs only if they are present in the file.

Parameters

<i>fn</i>	Provides the file-name for the MIDI file to be opened.
-----------	--

10.35.3.2 `rc_error_dialog()`

```
void seq64::mainwnd::rc_error_dialog (
    const std::string & message )
```

We can't yet display the specific error, except in a terminal window.

Parameters

<i>message</i>	Provides the error message returned by the configuration file.
----------------	--

10.35.3.3 `ppqn()` [1/2]

```
int seq64::mainwnd::ppqn ( ) const [inline]
```

10.35.3.4 ppqn() [2/2]

```
void seq64::mainwnd::ppqn (
    int ppqn ) [inline]
```

```
m_ppqn = choose_ppqn(ppqn);
```

10.35.3.5 handle_signal()

```
void seq64::mainwnd::handle_signal (
    int sig ) [static], [private]
```

10.35.3.6 debug_text()

```
void seq64::mainwnd::debug_text (
    const std::string & tag,
    int value ) [private]
```

Parameters

<i>tag</i>	Human-readable text to show the context of the message. Keep it well under 80 characters.
<i>value</i>	An integer value to be shown.

10.35.3.7 multi_wid()

```
bool seq64::mainwnd::multi_wid ( ) const [inline], [private]
```

10.35.3.8 adj_callback_wid()

```
void seq64::mainwnd::adj_callback_wid (
    int widblock ) [private]
```

Note that we have to actually set the perform object's screen-set, to get its screen-set offsets in place, before logging the changes to the mainwid.

Parameters

<i>widblock</i>	This parameter ranges from 0 to 5, depending on how many mainwids are created. This function operates only when multiple mainwids are active. Just to be clear, index 0 is slot [0, 0], 1 is slot [0, 1], etc. The row index varies fastest, just like the seq-number does in a set.
-----------------	--

10.35.3.9 independent()

```
bool seq64::mainwnd::independent ( ) const [inline], [private]
```

10.35.3.10 need_set_spinner()

```
bool seq64::mainwnd::need_set_spinner (
    int block ) const [inline], [private]
```

10.35.3.11 adj_callback_ss()

```
void seq64::mainwnd::adj_callback_ss ( ) [private]
```

Its sets the active screen-set value in the Performance/Song window, the Patterns, and something about setting the text based on a screen-set notepad from the Performance/Song window. We let the perform object keep track of modifications.

Multi-mainwid mode:

This status is flagged by the [multi_wid\(\)](#) function. If true, then we want the Set spin-button to affect the active sequence and all sequences. Let's start Seq64 in 2 columns x 3 rows mode. The mainwid slots are set up so that, like the screen-sets, the row index varies fastest. Here is the layout:

Column		0	1
Row	0	[0,0]	[1,0]
Row	1	[0,1]	[1,1]
Row	2	[0,2]	[1,2]

This two-dimensional array can be access by stepping along a one-dimensional array, with the one-dimensional array having an index we will call "slot number" for this description:

1-D index	2-D indices
[0]	[0,0]
[1]	[0,1]
[2]	[0,2]
[3]	[1,0]
[4]	[1,1]
[5]	[1,2]

When the application starts, the "slot numbers" in the left column correspond exactly to set numbers. The 0th set is the "active" set. When the Set spinner is incremented, the 1st set should become the active one. As it is incremented, the active slot will move from top-to-bottom and then left-to-right, until the number of visible mainwids is reached. In that case, we want to then increment the set number for all of the mainwids in the matrix, so that they move down one row, and slot 0 now holds the active set. Or, for a direct edit, move that set directly to slot 0.

For now, let's just start from slot 0 and set the values of all slots, and worry about preserving relative set-numbers later.

10.35.3.12 adj_callback_bpm()

```
void seq64::mainwnd::adj_callback_bpm ( ) [private]
```

Let the perform object keep track of modifications.

10.35.3.13 edit_callback_notepad()

```
void seq64::mainwnd::edit_callback_notepad ( ) [private]
```

Let the perform object keep track of modifications.

10.35.3.14 set_wid_label()

```
void seq64::mainwnd::set_wid_label (
    int ss,
    int block = 0 ) [private]
```

Parameters

<i>ss</i>	Provides the number of the screen-set to use to retrieve and show the screen-set label. This value is not sanity-checked.
<i>block</i>	Provides the number of the mainwid block for which the screen-set value applies. Defaults to 0. This value is not checked, but the function is private, so we guarantee its safety.

10.35.3.15 update_screenset()

```
void seq64::mainwnd::update_screenset ( ) [private]
```

Updates the screen-set by comparing the current screen-set to that active in the perform object.

Change Note ca 2018-02-03 Fixed issue #135, was using newset!

10.35.3.16 update_markers()

```
void seq64::mainwnd::update_markers (
    midipulse tick ) [private]
```

Parameters

<i>tick</i>	The current tick number for playback, etc.
-------------	--

10.35.3.17 reset()

```
void seq64::mainwnd::reset ( ) [private]
```

The reset function draws the patterns on the pixmap, and then draws the pixmap on the window.

10.35.3.18 reset_window()

```
void seq64::mainwnd::reset_window ( ) [private]
```

- Active screenset
- Screenset numbers for all of the mainwids
- Screenset labels for all of the mainwids
- The values in all of the spinbuttons/screenset fields TODO TODO

10.35.3.19 set_play_image()

```
void seq64::mainwnd::set_play_image (
    bool isrunning ) [private]
```

Is this a memory leak? Some users report segfaults (all of a sudden) with this setting!

Parameters

<i>isrunning</i>	If true, set the image to the "Pause" icon, since playback is running. Otherwise, set it to the "Play" button, since playback is not running.
------------------	---

10.35.3.20 set_sonlive_image()

```
void seq64::mainwnd::set_sonlive_image (
    bool issong ) [private]
```

Parameters

<i>issong</i>	If true, set the image to the "Song" icon. Otherwise, set it to the "Live" button.
---------------	--

10.35.3.21 start_playing()

```
void seq64::mainwnd::start_playing ( ) [private]
```

An accessor to [perform::start_playing\(\)](#). This function is actually a callback for the pause/play button. Now very similar to [perfedit::start_playing\(\)](#), except that the implicit songmode == false parameter is used here.

10.35.3.22 pause_playing()

```
void seq64::mainwnd::pause_playing ( ) [private]
```

Currently, it is just the same as [stop_playing\(\)](#), but we will get it to work.

10.35.3.23 stop_playing()

```
void seq64::mainwnd::stop_playing ( ) [private]
```

An accessor to perform's [stop_playing\(\)](#) function. Also calls the [mainwid::update_sequences_on_window\(\)](#) function. Not sure that we need this call, since the slots seem to update anyway. But we've noticed that, with this call in place, hitting the Stop button causes a subtle change in the appearance of the first non-empty pattern of the "allofarow.mid" file.

After the Stop button is pushed (in ALSA mode), then the Space key ("start") doesn't work properly. The song starts, then quickly stops. It doesn't matter if [update_sequences_on_window\(\)](#) is called or not. This happens even in seq24! This bug has proven incredibly difficult to track down, still working on it.

10.35.3.24 toggle_playing()

```
void seq64::mainwnd::toggle_playing ( ) [private]
```

Meant only to be called when the "Play" button is pressed, if the pause feature has been compiled into the application.

10.35.3.25 timer_callback()

```
bool seq64::mainwnd::timer_callback ( ) [private]
```

It also supports the ALSA pause functionality.

Note

When Sequencer64 first starts up, and no MIDI tune is loaded, the call to [mainwid::update_markers\(\)](#) leads to trying to do some work on sequences that don't yet exist. Also, if a sequence is changed by the event editor, we get a crash; need to find out how seqedit gets away with the changes.

Returns

Always returns true. This allows the callback to be called repeatedly. If one wants to stop the timeout callback, then return false.

10.35.3.26 set_screenset()

```
int seq64::mainwnd::set_screenset (
    int screenset ) [private]
```

Sets the active screenset to the given value. This is used by the main Set spin-button and by the [timer_callback\(\)](#) function if the screen set is changed via the perform object (i.e. via MIDI control).

The perform object's screen-set is modified as well.

Parameters

<i>screenset</i>	The new prospective screen-set value. This will become the active screen-set.
<i>set_adjust_ss</i>	If true (the default), this parameter causes the <code>m_adjust_ss</code> control to be updated. If false, it is not update. A value of false is necessary if the spinbutton was clicked by the user.

10.35.3.27 tempo_log()

```
void seq64::mainwnd::tempo_log ( ) [private]
```

Todo Upgrade this so that a Ctrl-click calls `toggle_tempo_record`, so that we can eliminate a tempo button; there are too many buttons.

10.35.3.28 toggle_tempo_record()

```
void seq64::mainwnd::toggle_tempo_record ( ) [private]
```

10.35.3.29 toggle_time_format()

```
void seq64::mainwnd::toggle_time_format ( ) [private]
```

This functionality currently does not have a key devoted to it, nor is it a saved setting.

10.35.3.30 queue_it()

```
void seq64::mainwnd::queue_it ( ) [private]
```

10.35.3.31 set_song_record()

```
void seq64::mainwnd::set_song_record ( ) [private]
```

Note that calling this function will trigger the button signal callback.

10.35.3.32 toggle_song_record()

```
void seq64::mainwnd::toggle_song_record ( ) [private]
```

This is not a saved setting at this time.

There is no need to change the button color or the image, as the control itself indicates when song-recording is on.

```
Gtk::Image * image_song = manage(new PIXBUF_IMAGE(song_rec_off_xpm));  
m_button_song_record->set_image(*image_song);
```

10.35.3.33 toggle_song_snap()

```
void seq64::mainwnd::toggle_song_snap ( ) [private]
```

This functionality currently does not have a key devoted to it, nor is it a saved setting.

10.35.3.34 set_song_playback()

```
void seq64::mainwnd::set_song_playback (   
    bool playsong ) [private]
```

Parameters

<i>playsong</i>	Set to true if playback (Song) mode is to be in force.
-----------------	--

10.35.3.35 song_record_snap()

```
void seq64::mainwnd::song_record_snap (   
    bool snap ) [inline], [private]
```

10.35.3.36 panic()

```
void seq64::mainwnd::panic ( ) [inline], [private]
```

10.35.3.37 learn_toggle()

```
void seq64::mainwnd::learn_toggle ( ) [inline], [private]
```

Simply forwards the call to [perform::learn_toggle\(\)](#).

10.35.3.38 open_performance_edit()

```
void seq64::mainwnd::open_performance_edit ( ) [private]
```

We will let perform keep track of modifications, and not just set an is-modified flag just because we opened the song editor. We're going to centralize the modification flag in the perform object, and see if it can work.

10.35.3.39 open_performance_edit_2()

```
void seq64::mainwnd::open_performance_edit_2 ( ) [private]
```

Experiment: open a second one and see what happens. It works, but one needs to tell the other to redraw if a change is made.

10.35.3.40 enregister_perfedits()

```
void seq64::mainwnd::enregister_perfedits ( ) [private]
```

10.35.3.41 sequence_key()

```
void seq64::mainwnd::sequence_key (
    int seq ) [private]
```

Parameters

<i>seq</i>	This is actually the key-number.
------------	----------------------------------

10.35.3.42 spinner_max()

```
int seq64::mainwnd::spinner_max ( ) const [inline], [private]
```

Remember that set numbers go from 0 to 31, both internally and visually, for a total of 32 sets.

10.35.3.43 apply_song_transpose()

```
void seq64::mainwnd::apply_song_transpose ( ) [private]
```

Then reset the perfedit transpose setting to 0.

10.35.3.44 clear_mute_groups()

```
void seq64::mainwnd::clear_mute_groups ( ) [private]
```

Sets all values to false/zero. Also, since the intent might be to clean up the MIDI file, the user is prompted to save.

10.35.3.45 reload_mute_groups()

```
void seq64::mainwnd::reload_mute_groups ( ) [private]
```

10.35.3.46 update_window_title()

```
void seq64::mainwnd::update_window_title ( ) [private]
```

Note that the name of the application is obtained by the "(SEQ64_PACKAGE)" construction.

The format of the caption bar is the name of the package/application, followed by the file-specification (shortened if necessary so that the name of the file itself can be seen), ending with the PPQN value in parentheses.

10.35.3.47 update_recent_files_menu()

```
void seq64::mainwnd::update_recent_files_menu ( ) [private]
```

10.35.3.48 load_recent_file()

```
void seq64::mainwnd::load_recent_file (
    int index ) [private]
```

This function passes false as the shorten parameter of [rc_settings::recent_file\(\)](#).

Parameters

<i>index</i>	Indicates which file in the list to open, ranging from 0 to the number of recent files minus 1. If set to -1, then nothing is done.
--------------	---

10.35.3.49 toLower()

```
void seq64::mainwnd::toLower (
    std::string & s ) [private]
```

10.35.3.50 file_new()

```
void seq64::mainwnd::file_new ( ) [inline], [private]
```

10.35.3.51 file_open()

```
void seq64::mainwnd::file_open ( ) [inline], [private]
```

10.35.3.52 file_save()

```
void seq64::mainwnd::file_save ( ) [inline], [private]
```

10.35.3.53 set_song_mute()

```
void seq64::mainwnd::set_song_mute (
    perform::mute_op_t op ) [inline], [private]
```

10.35.3.54 wid_box_to_slot()

```
int seq64::mainwnd::wid_box_to_slot (
    int col,
    int row ) const [private]
```

10.35.3.55 wid_slot_to_box()

```
bool seq64::mainwnd::wid_slot_to_box (
    int slot,
    int & col,
    int & row ) const [private]
```

10.35.3.56 file_import_dialog()

```
void seq64::mainwnd::file_import_dialog ( ) [private]
```

Note that every track of the MIDI file will be imported, even if the track is only a label track (without any MIDI events), or a very long track.

The main difference between the Open operation and the Import operation seems to be that the latter can read MIDI files into a screen-set greater than screen-set 0. No, that's not true, so far. No matter what the current screen-set setting, the import is appended after the current data in screen-set 0. Then, if it overflows that screen-set, the overflow goes into the next screen-set.

It might be nice to have the option of importing a MIDI file into a specific screen-set, for better organization, as well as being able to offset the sequence number.

Also, it is important to note that `perf().clear_all()` is not called by this routine, as we are merely adding to what might already be there.

10.35.3.57 options_dialog()

```
void seq64::mainwnd::options_dialog ( ) [private]
```

10.35.3.58 jack_dialog()

```
void seq64::mainwnd::jack_dialog ( ) [private]
```

10.35.3.59 about_dialog()

```
void seq64::mainwnd::about_dialog ( ) [private]
```

I (Chris) took the liberty of tacking my name at the end, and hope to have done eventually enough work to warrant having it there. Hmmmmm....

10.35.3.60 build_info_dialog()

```
void seq64::mainwnd::build_info_dialog ( ) [private]
```

It is similar to the "--version" option on the command line. The AboutDialog doesn't seem to have a way to left-align the text, so we're trying the MessageDialog.

10.35.3.61 query_save_changes()

```
int seq64::mainwnd::query_save_changes ( ) [private]
```

10.35.3.62 new_open_error_dialog()

```
void seq64::mainwnd::new_open_error_dialog ( ) [private]
```

10.35.3.63 file_save_as()

```
void seq64::mainwnd::file_save_as (
    SaveOption option = FILE_SAVE_AS_NORMAL ) [private]
```

Please note that Sequencer64 will not adopt the "c_seq32_midl" type of file, because it already saves its files in a format that other sequencers should be able to read.

Stazed on the intent of the export functionality:

The original intent was to be able to play an exported song in something like TiMIDItY. After I completed things I realized that there could be an editing benefit as well. I like to record from my MIDI keyboard, improvised to a drum beat, on a long sequence (64 measures). Some is junk, but there are usually parts that I can use. In original seq24, to cut out the good or bad stuff, you would have to search the sequence by listening, then cut and move or copy and paste to a new sequence. It could be done but was always tedious. The paste box for the sequence sometimes made it difficult to find the correct note location, measure, and beat. Also, on a long sequence, you need to zoom out to see the copy location as it played, but zoom in for the precise paste location. In addition if you wanted to change the measure of the notes, it became a trial and error of copy/paste, listen, move, listen, move....

With the added Song editor feature of split trigger to mouse and copy paste trigger to mouse, you can now do all the editing from the song editor. Listen to the sequence, cut out the good or bad parts and reassemble. Move or copy all good trigger parts to the left start and delete all the bad stuff. Now you can use the song export to create the new sequence. Just mute all other tracks and export. Re-import and the new cleaned sequence is already done. Also I use it for importing drum beats from a single '32/'42 file that contains dozens of different styles with intros and endings. I like to sync two instances of '32 or '42 together with jack, then play/experiment with the different beats. If I find something I like, create the song trigger for the part I like in the drum file, export and import.

I actually do not use the song export for anything but editing.

Note that the split trigger variant of Stazed, where it doesn't just split the section in half, is not yet implemented (2016-08-05).

Parameters

<i>option</i>	Indicates how to save or export the MIDI sequences. The default value of this parameter is FILE_SAVE_AS_NORMAL. The export options allow one to save the file as if the triggers were employed, or with a lot of the Sequencer64-specific information removed.
---------------	--

10.35.3.64 file_exit()

```
void seq64::mainwnd::file_exit ( ) [private]
```

10.35.3.65 new_file()

```
void seq64::mainwnd::new_file ( ) [private]
```

Not sure that we need to clear the modified flag here, especially since it is now centralized in the perform object. Let `perf().clear_all()` handle it now.

Question Should we do a save check here, a la Kepler34?

10.35.3.66 save_file()

```
bool seq64::mainwnd::save_file ( ) [private]
```

Here we specify the current value of `m_ppqn`, which was set when reading the MIDI file. We also let `midifile` tell the perform that saving worked, so that the "is modified" flag can be cleared. The `midifile` class is already a friend of `perform`.

Note that we do not support saving files in the Cakewalk WRK format.

10.35.3.67 choose_file()

```
void seq64::mainwnd::choose_file ( ) [private]
```

Change Note layk 2016-10-11 Issue #43 Added filters for upper-case MIDI-file extensions.

10.35.3.68 is_save()

```
bool seq64::mainwnd::is_save ( ) [private]
```

10.35.3.69 install_signal_handlers()

```
bool seq64::mainwnd::install_signal_handlers ( ) [private]
```

10.35.3.70 signal_action()

```
bool seq64::mainwnd::signal_action (
    Glib::IOCondition condition ) [private]
```

Returns

Returns true if the signalling was able to be completed, even if it was an unexpected signal.

10.35.3.71 edit_field_has_focus()

```
bool seq64::mainwnd::edit_field_has_focus ( ) const [private]
```

Todo We may have to revisit this one to add the Adjustment objects and the extra objects created in multi-wid mode.

Returns

Returns true if one of the three editable/modifiable fields has the keyboard focus.

10.35.3.72 populate_menu_file()

```
void seq64::mainwnd::populate_menu_file ( ) [private]
```

Provided to make the constructor more readable and manageable.

10.35.3.73 populate_menu_edit()

```
void seq64::mainwnd::populate_menu_edit ( ) [private]
```

Provided to make the constructor more readable and manageable.

10.35.3.74 populate_menu_help()

```
void seq64::mainwnd::populate_menu_help ( ) [private]
```

Provided to make the constructor more readable and manageable.

10.35.3.75 populate_menu_view()

```
void seq64::mainwnd::populate_menu_view ( ) [private]
```

It repeats the song editor edit command, just to help those whose muscle memory is already seq32-oriented. Provided to make the constructor more readable and manageable. View menu items and their hot keys.

10.35.3.76 set_status_text()

```
void seq64::mainwnd::set_status_text (
    const std::string & text ) [private]
```


Parameters

<i>text</i>	Provides the short (6 characters in the default state) string to set the label.
-------------	---

10.35.3.77 on_delete_event()

```
bool seq64::mainwnd::on_delete_event (
    GdkEventAny * ev ) [private]
```

Any changed data is saved. If the pattern is playing, then it is stopped. We now use [perform::is_pattern_playing\(\)](#).

10.35.3.78 on_key_press_event()

```
bool seq64::mainwnd::on_key_press_event (
    GdkEventKey * ev ) [private]
```

It also handles the control-key and modifier-key combinations matching the entries in its list of if statements.

Also, we now effectively press the CAPS LOCK key for the user if in group-learn mode, via the [keystroke::shift_lock\(\)](#) function. Toggle the sequence mute/unmute setting using keyboard keys. However, do not do this if the Ctrl key is being pressed. Ctrl-E, for example, brings up the Song Editor, and should not toggle the sequence controlled by the "e" key. Will also see if the Alt key could/should be intercepted.

Also, try to avoid using the hotkeys if an editable field has focus.

Also, if the pattern-edit keys (will be minus and equals by default) are enabled and have been pressed, then bring up one of the editors (pattern or event) when the slot shortcut key is pressed.

Finally, as a new feature, if the pattern-shift key (the forward slash by default) is pressed, increment the flag that indicates an extended sequence (value + 32 [c_seqs_in_set] or value + 64) will be toggled, instead of the normal sequence.

10.35.3.79 on_key_release_event()

```
bool seq64::mainwnd::on_key_release_event (
    GdkEventKey * ev ) [private]
```

Is this worth turning into a switch statement? Or offloading to a perform member function? The latter. Also, we now effectively press the CAPS LOCK key for the user if in group-learn mode. The function that does this is [keystroke::shift_lock\(\)](#).

Todo Test this functionality in old and new application.

Returns

Always returns false. This matches seq24 behavior.

10.35.3.80 on_realize()

```
void seq64::mainwnd::on_realize ( ) [private]
```

without first moving the mouse to the main window. Weird with a beard! Might be a Fluxbox-related issue.

10.35.3.81 on_grouplearnchange()

```
void seq64::mainwnd::on_grouplearnchange (
    bool state ) [private], [virtual]
```

This handler responds to a learn-mode change from [perf\(\)](#).

Reimplemented from [seq64::performcallback](#).

10.35.4 Field Documentation

10.35.4.1 sm_sigpipe

```
int seq64::mainwnd::sm_sigpipe [static], [private]
```

This static member provides a couple of pipes for signalling/messaging.

10.35.4.2 sm_widmax

```
const int seq64::mainwnd::sm_widmax [static], [private]
```

More checks, but less incrementing and array-offset calculations.

10.35.4.3 m_menubar

```
Gtk::MenuBar* seq64::mainwnd::m_menubar [private]
```

The whole menu bar.

10.35.4.4 m_menu_file

```
Gtk::Menu* seq64::mainwnd::m_menu_file [private]
```

10.35.4.5 m_menu_recent

Gtk::Menu* seq64::mainwnd::m_menu_recent [private]

10.35.4.6 m_menu_edit

Gtk::Menu* seq64::mainwnd::m_menu_edit [private]

10.35.4.7 m_menu_view

Gtk::Menu* seq64::mainwnd::m_menu_view [private]

10.35.4.8 m_menu_help

Gtk::Menu* seq64::mainwnd::m_menu_help [private]

10.35.4.9 m_status_label

Gtk::Label* seq64::mainwnd::m_status_label [private]

10.35.4.10 m_ppqn

int seq64::mainwnd::m_ppqn [private]

We need it early here to be able to pass it along to child objects.

10.35.4.11 m_mainwid_grid

Gtk::Table* seq64::mainwnd::m_mainwid_grid [private]

This item is not used if only one mainwid is configured.

10.35.4.12 m_mainwid_frames

Gtk::Frame* seq64::mainwnd::m_mainwid_frames[SEQ64_MAINWIDS_MAX] [private]

Each frame will hold a mainwid object, and the text part of the frame will show the set number for that mainwid. It's only 6 pointers, no need to fret over dynamic allocation.

10.35.4.13 m_mainwid_adjustors

```
Gtk::Adjustment* seq64::mainwnd::m_mainwid_adjustors[SEQ64_MAINWIDS_MAX] [private]
```

10.35.4.14 m_mainwid_spinners

```
Gtk::SpinButton* seq64::mainwnd::m_mainwid_spinners[SEQ64_MAINWIDS_MAX] [private]
```

10.35.4.15 m_mainwid_blocks

```
mainwid* seq64::mainwnd::m_mainwid_blocks[SEQ64_MAINWIDS_MAX] [private]
```

10.35.4.16 m_mainwid_rows

```
int seq64::mainwnd::m_mainwid_rows [private]
```

Defaults to 1.

10.35.4.17 m_mainwid_columns

```
int seq64::mainwnd::m_mainwid_columns [private]
```

Defaults to 1.

10.35.4.18 m_mainwid_count

```
int seq64::mainwnd::m_mainwid_count [private]
```

Saves multiplications and static value checks.

10.35.4.19 m_mainwid_independent

```
bool seq64::mainwnd::m_mainwid_independent [private]
```

10.35.4.20 m_main_wid

```
mainwid* seq64::mainwnd::m_main_wid [private]
```

We end up sharing this object with perfedit, perfnames, and seqedit in order to allow the seqedit object to notify the mainwid (indirectly) of the currently-edited sequence.

If the SEQ64_MULTI_MAINWID build option is in force, this pointer is used for highlighting and activating the mainwid that was last clicked. It starts out as the upper left mainwid.

10.35.4.21 m_adjust_ss

```
Gtk::Adjustment* seq64::mainwnd::m_adjust_ss [private]
```

Screenset adjustment.

10.35.4.22 m_spinbutton_ss

```
Gtk::SpinButton* seq64::mainwnd::m_spinbutton_ss [private]
```

10.35.4.23 m_current_screenset

```
int seq64::mainwnd::m_current_screenset [private]
```

10.35.4.24 m_main_time

```
maintime* seq64::mainwnd::m_main_time [private]
```

10.35.4.25 m_perf_edit

```
perfedit* seq64::mainwnd::m_perf_edit [private]
```

10.35.4.26 m_perf_edit_2

```
perfedit* seq64::mainwnd::m_perf_edit_2 [private]
```

The second makes it easy to line up two different patterns that cannot be seen together on one performance editor.

10.35.4.27 m_options

```
options* seq64::mainwnd::m_options [private]
```

10.35.4.28 m_main_cursor

```
Gdk::Cursor seq64::mainwnd::m_main_cursor [private]
```

10.35.4.29 m_image_play

```
Gtk::Image* seq64::mainwnd::m_image_play [private]
```

10.35.4.30 m_button_panic

```
Gtk::Button* seq64::mainwnd::m_button_panic [private]
```

10.35.4.31 m_button_learn

```
Gtk::Button* seq64::mainwnd::m_button_learn [private]
```

10.35.4.32 m_button_stop

```
Gtk::Button* seq64::mainwnd::m_button_stop [private]
```

10.35.4.33 m_button_play

```
Gtk::Button* seq64::mainwnd::m_button_play [private]
```

If configured to support pause, it also supports the pause pixmap and functionality.

10.35.4.34 m_button_tempo_log

```
Gtk::Button* seq64::mainwnd::m_button_tempo_log [private]
```

The user clicks on it to log the current tempo value at the current time as a Set Tempo event.

10.35.4.35 m_button_tempo_record

```
Gtk::ToggleButton* seq64::mainwnd::m_button_tempo_record [private]
```

One should be able to left click on it to record the current tempo as a tempo event, and right-click to enable auto-record.

10.35.4.36 m_is_tempo_recording

```
bool seq64::mainwnd::m_is_tempo_recording [private]
```

10.35.4.37 m_button_perfedit

```
Gtk::Button* seq64::mainwnd::m_button_perfedit [private]
```

10.35.4.38 m_button_jack

```
Gtk::Button* seq64::mainwnd::m_button_jack [private]
```

10.35.4.39 m_button_song_record

```
Gtk::ToggleButton* seq64::mainwnd::m_button_song_record [private]
```

10.35.4.40 m_button_song_snap

```
Gtk::ToggleButton* seq64::mainwnd::m_button_song_snap [private]
```

10.35.4.41 m_is_song_recording

```
bool seq64::mainwnd::m_is_song_recording [private]
```

10.35.4.42 m_is_snap_recording

```
bool seq64::mainwnd::m_is_snap_recording [private]
```

10.35.4.43 m_tick_time

```
Gtk::Label* seq64::mainwnd::m_tick_time [private]
```

Long overdue, actually!

10.35.4.44 m_button_time_type

```
Gtk::Button* seq64::mainwnd::m_button_time_type [private]
```

10.35.4.45 m_tick_time_as_bbt

```
bool seq64::mainwnd::m_tick_time_as_bbt [private]
```

The default is true: bar:beats:ticks.

10.35.4.46 m_adjust_bpm

```
Gtk::Adjustment* seq64::mainwnd::m_adjust_bpm [private]
```

BPM adjustment object.

10.35.4.47 m_spinbutton_bpm

```
Gtk::SpinButton* seq64::mainwnd::m_spinbutton_bpm [private]
```

10.35.4.48 m_button_queue

```
Gtk::ToggleButton* seq64::mainwnd::m_button_queue [private]
```

10.35.4.49 m_adjust_load_offset

```
Gtk::Adjustment* seq64::mainwnd::m_adjust_load_offset [private]
```

These controls are used in the File / Import dialog to change where the imported file will be loaded in the sequences space, which ranges from 0 to 1024 in blocks of 32 patterns. Load number for import.

10.35.4.50 m_spinbutton_load_offset

```
Gtk::SpinButton* seq64::mainwnd::m_spinbutton_load_offset [private]
```

10.35.4.51 m_entry_notes

```
Gtk::Entry* seq64::mainwnd::m_entry_notes [private]
```

This is just a long text-edit field that can be used to enter a long name or a short description of the current screenset.

10.35.4.52 m_is_running

```
bool seq64::mainwnd::m_is_running [private]
```

10.35.4.53 m_timeout_connect

```
sigc::connection seq64::mainwnd::m_timeout_connect [private]
```

10.35.4.54 m_menu_mode

```
bool seq64::mainwnd::m_menu_mode [private]
```

This is a "stazed" feature that might be generally useful. This value is true if the menu-bar is to be enabled.

10.35.4.55 m_call_seq_edit

```
bool seq64::mainwnd::m_call_seq_edit [private]
```

Currently, the hard-wired key for this function is the equals key.

10.35.4.56 m_call_seq_shift

```
int seq64::mainwnd::m_call_seq_shift [private]
```

It causes 32 (c_seqs_in_set) to be added to the hot key. Actually, let's make it an integer that can range from 0 (off) to 1 to 2 (m_seqs_in_set / c_seqs_in_set).

10.35.4.57 m_call_seq_eventedit

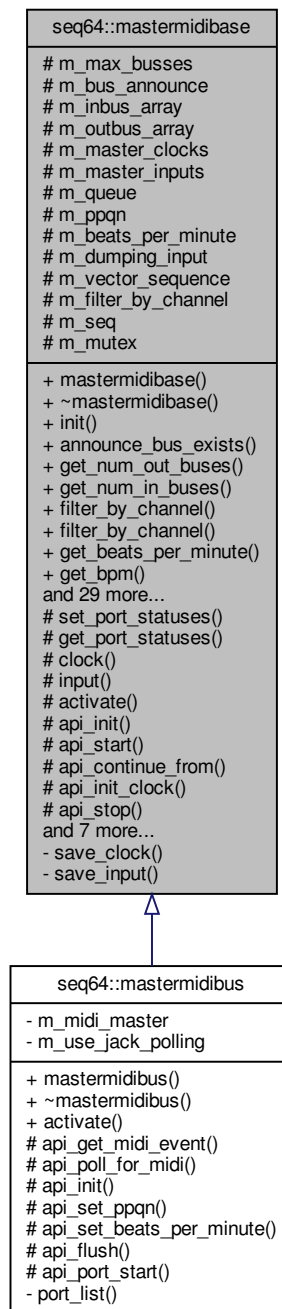
```
bool seq64::mainwnd::m_call_seq_eventedit [private]
```

Currently, the hard-wired key for this function is the minus key.

10.36 seq64::mastermidibase Class Reference

The class that "supervises" all of the midibus objects?

Inheritance diagram for seq64::mastermidibase:



Public Member Functions

- [mastermidibase](#) (int ppqn=SEQ64_USE_DEFAULT_PPQN, [midibpm](#) bpm=SEQ64_DEFAULT_BPM)

The mastermidibase default constructor fills the array with our busses.

- virtual [~mastermidibase](#) ()

The virtual destructor deletes all of the output busses, clears out the ALSA events, stops and frees the queue, and closes ALSA for this application.

- virtual void `init` (int ppqn, `midibpm` bpm)
Initialize the mastermidibus using the implementation-specific API function.
- bool `announce_bus_exists` () const
Indicates that we have an announce buss entry to skip when filling in the device list with "user" entries.
- int `get_num_out_buses` () const
'Getter' function for member m_num_out_buses
- int `get_num_in_buses` () const
'Getter' function for member m_num_in_buses
- bool `filter_by_channel` () const
'Getter' function for member m_filter_by_channel
- void `filter_by_channel` (bool flag)
'Setter' function for member m_filter_by_channel
- `midibpm` `get_beats_per_minute` () const
'Getter' function for member m_beats_per_minute
- `midibpm` `get_bpm` () const
'Getter' function for member m_beats_per_minute This is a second version.
- int `get_ppqn` () const
'Getter' function for member m_ppqn
- bool `is_dumping` () const
'Getter' function for member m_dumping_input
- `sequence` * `get_sequence` () const
'Getter' function for member m_seq Used only in `perform::input_func()` when not filtering MIDI input by channel.
- void `start` ()
Starts all of the configured output busses up to m_num_out_buses.
- void `stop` ()
Stops each of the MIDI output busses.
- void `port_start` (int client, int port)
Start the given MIDI port.
- void `port_exit` (int client, int port)
Turn off the given port for the given client.
- void `play` (`bussbyte` bus, `event` *e24, `midibyte` channel)
Handle the playing of MIDI events on the MIDI buss given by the parameter, as long as it is a legal buss number.
- void `continue_from` (`midipulse` tick)
Gets the MIDI output busses running again.
- void `init_clock` (`midipulse` tick)
Initializes the clock of each of the MIDI output busses.
- void `emit_clock` (`midipulse` tick)
Generates the MIDI clock for each of the output busses.
- void `sysex` (`event` *event)
Handle the sending of SYSEX events.
- void `print` () const
Print some information about the available MIDI input and output busses.
- void `flush` ()
Flushes our local queue events out The implementation-specific API function is called.
- void `panic` ()
Stops all notes on all channels on all busses.
- void `set_sequence_input` (bool state, `sequence` *seq)
Set the input sequence object, and set the m_dumping_input value to the given state.
- void `dump_midi_input` (`event` in)
This function augments the recording functionality by looking for a sequence that has a matching channel number, logging the event to that sequence, and then immediately exiting.

- `std::string get_midi_out_bus_name (bussbyte bus)`
Get the MIDI output buss name for the given (legal) buss number.
- `std::string get_midi_in_bus_name (bussbyte bus)`
Get the MIDI input buss name for the given (legal) buss number.
- `int poll_for_midi ()`
Initiate a poll() on the existing poll descriptors.
- `bool is_more_input ()`
Test the sequencer to see if any more input is pending.
- `bool get_midi_event (event *in)`
Grab a MIDI event via the currently-selected MIDI API.
- `bool set_clock (bussbyte bus, clock_e clock_type)`
Set the clock for the given (legal) buss number.
- `bool set_input (bussbyte bus, bool inputing)`
Set the status of the given input buss, if a legal buss number.
- `bool get_input (bussbyte bus)`
Get the input for the given (legal) buss number.
- `bool is_input_system_port (bussbyte bus)`
Get the system-buss status for the given (legal) buss number.
- `clock_e get_clock (bussbyte bus)`
Gets the clock setting for the given (legal) buss number.
- `void set_ppqn (int ppqn)`
Set the PPQN value (parts per quarter note).
- `void set_beats_per_minute (midibpm bpm)`
Set the BPM value (beats per minute).

Protected Member Functions

- `void set_port_statuses (const std::vector< clock_e > &clocks, const std::vector< bool > &inputs)`
'Setter' function for member m_master_clocks, m_master_inputs.
- `void get_port_statuses (std::vector< clock_e > &clocks, std::vector< bool > &inputs)`
'Getter' function for member m_master_clocks, m_master_inputs.
- `clock_e clock (int bus)`
- `bool input (int bus)`
- `virtual bool activate ()`
Initializes and ctivates the busses, in a partly API-dependent manner.
- `virtual void api_init (int ppqn, midibpm bpm)=0`
- `virtual void api_start ()`
Provides MIDI API-specific functionality for the start() function.
- `virtual void api_continue_from (midipulse)`
Provides MIDI API-specific functionality for the continue_from() function.
- `virtual void api_init_clock (midipulse)`
Provides MIDI API-specific functionality for the init_clock() function.
- `virtual void api_stop ()`
Provides MIDI API-specific functionality for the stop() function.
- `virtual void api_set_ppqn (int)`
Provides MIDI API-specific functionality for the set_ppqn() function.
- `virtual void api_set_beats_per_minute (midibpm)`
Provides MIDI API-specific functionality for the set_beats_per_minute() function.
- `virtual void api_flush ()`
Provides MIDI API-specific functionality for the flush() function.

- virtual void `api_clock()`
Provides MIDI API-specific functionality for the `clock()` function.
- virtual void `api_port_start(int, int)`
- virtual bool `api_get_midi_event(event *inev)=0`
- virtual int `api_poll_for_midi()`
Provides a default implementation of `api_poll_for_midi()`.

Protected Attributes

- int `m_max_busses`
The maximum number of busses supported.
- `midibus * m_bus_announce`
MIDI buss announcer.
- `busarray m_inbus_array`
Encapsulates information about the input busses.
- `busarray m_outbus_array`
Encapsulates information about the output busses.
- `std::vector< clock_e > m_master_clocks`
Saves the clock settings obtained from the "rc" (options) file so that they can be loaded into the mastermidibus once it is created.
- `std::vector< bool > m_master_inputs`
Saves the input settings obtained from the "[midi-input]" section of the "rc" (options) file, so that they can be loaded into the mastermidibus once it is created.
- int `m_queue`
The ID of the MIDI queue.
- int `m_ppqn`
Resolution in parts per quarter note.
- `midibpm m_beats_per_minute`
BPM (beats per minute).
- bool `m_dumping_input`
For dumping MIDI input to a sequence for recording.
- `std::vector< sequence * > m_vector_sequence`
Used for the new "stazed" feature of filtering MIDI channels so that a sequence gets only the channels meant for it.
- bool `m_filter_by_channel`
If true, the `m_vector_sequence` container is used to divert incoming data to the sequence that has the channel it is meant for.
- `sequence * m_seq`
Points to the sequence object.
- `mutex m_mutex`
The locking mutex.

Private Member Functions

- bool `save_clock(bussbyte bus, clock_e clock)`
Saves the given clock value in `m_master_clocks[bus]`.
- bool `save_input(bussbyte bus, bool inputing)`
Saves the input status (as selected in the MIDI Input tab).

Friends

- class [perform](#)
- class [midi_alsa_info](#)

10.36.1 Constructor & Destructor Documentation

10.36.1.1 mastermidibase()

```
seq64::mastermidibase::mastermidibase (
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

Parameters

<i>ppqn</i>	Provides the PPQN value for this object. However, in most cases, the default, SEQ64_USE_DEFAULT_PPQN should be specified. Then the caller of this constructor should call mastermidibase::set_ppqn() to set up the proper PPQN value.
<i>bpm</i>	Provides the beats per minute value, which defaults to <code>c_beats_per_minute</code> .

10.36.1.2 ~mastermidibase()

```
seq64::mastermidibase::~~mastermidibase ( ) [virtual]
```

Valgrind indicates we might have issues caused by the following functions:

- `snd_config_hook_load()`
- `snd_config_update_r()` via `snd_seq_open()`
- `_dl_init()` and other GNU function
- `init_gtkmm_internals()` [version 2.4]

10.36.2 Member Function Documentation

10.36.2.1 init()

```
virtual void seq64::mastermidibase::init (
    int ppqn,
    midibpm bpm ) [inline], [virtual]
```

A return value would be nice.

Parameters

<i>ppqn</i>	The PPQN value to which to initialize the master MIDI buss.
<i>bpm</i>	The beats/minute value to which to initialize the master MIDI buss.

10.36.2.2 `announce_bus_exists()`

```
bool seq64::mastermidibase::announce_bus_exists ( ) const [inline]
```

Another potentially equivalent test is `is_input_system_port(bus)`.

Returns

Returns true if `m_bus_announce` is not the null pointer.

10.36.2.3 `get_num_out_buses()`

```
int seq64::mastermidibase::get_num_out_buses ( ) const [inline]
```

10.36.2.4 `get_num_in_buses()`

```
int seq64::mastermidibase::get_num_in_buses ( ) const [inline]
```

10.36.2.5 `filter_by_channel()` [1/2]

```
bool seq64::mastermidibase::filter_by_channel ( ) const [inline]
```

10.36.2.6 `filter_by_channel()` [2/2]

```
void seq64::mastermidibase::filter_by_channel (
    bool flag ) [inline]
```


10.36.2.7 get_beats_per_minute()

```
midibpm seq64::mastermidibase::get_beats_per_minute ( ) const [inline]
```

10.36.2.8 get_bpm()

```
midibpm seq64::mastermidibase::get_bpm ( ) const [inline]
```

10.36.2.9 get_ppqn()

```
int seq64::mastermidibase::get_ppqn ( ) const [inline]
```

10.36.2.10 is_dumping()

```
bool seq64::mastermidibase::is_dumping ( ) const [inline]
```

10.36.2.11 get_sequence()

```
sequence* seq64::mastermidibase::get_sequence ( ) const [inline]
```

10.36.2.12 start()

```
void seq64::mastermidibase::start ( )
```

Calls the implementation-specific API function for starting.

Threadsafe

10.36.2.13 stop()

```
void seq64::mastermidibase::stop ( )
```

Then calls the implementation-specific API function to finalize the stoppage. (See the ALSA implementation in the seq_alsamidi library, for example. It is the original Sequencer64 implementation.)

Threadsafe

10.36.2.14 port_start()

```
void seq64::mastermidibase::port_start (
    int client,
    int port )
```

This function is called by [api_get_midi_event\(\)](#) when the ALSA event SND_SEQ_EVENT_PORT_START is received. Unlike [port_exit\(\)](#), the [port_start\(\)](#) function does rely on API-specific code, so we do need to create a virtual [api_port_start\(\)](#) function to implement the port-start event.

Threadsafe Quite a lot is done during the lock for the ALSA implimentation.

Parameters

<i>client</i>	Provides the client number, which is actually an ALSA concept.
<i>port</i>	Provides the client port, which is actually an ALSA concept.

10.36.2.15 port_exit()

```
void seq64::mastermidibase::port_exit (
    int client,
    int port )
```

Both the input and output busses for the given client are stopped: that is, set to inactive.

This function is called by [api_get_midi_event\(\)](#) when the ALSA event SND_SEQ_EVENT_PORT_EXIT is received. Since [port_exit\(\)](#) has no direct API-specific code in it, we do not need to create a virtual [api_port_exit\(\)](#) function to implement the port-exit event.

Threadsafe**Parameters**

<i>client</i>	The client to be matched and acted on. This value is actually an ALSA concept.
<i>port</i>	The port to be acted on. Both parameter must be matched before the buss is made inactive. This value is actually an ALSA concept.

10.36.2.16 play()

```
void seq64::mastermidibase::play (
    bussbyte bus,
    event * e24,
    midibyte channel )
```

There's currently no implementation-specific API function here.

Threadsafe**Parameters**

<i>bus</i>	The buss to start play on. Ooh, we just noticed that value should be checked before usage!
<i>e24</i>	The seq24 event to play on the buss. For speed, we don't bother to check the pointer.
<i>channel</i>	The channel on which to play the event.

10.36.2.17 continue_from()

```
void seq64::mastermidibase::continue_from (
    midipulse tick )
```

This function calls the implementation-specific API function, and then calls [midibus::continue_from\(\)](#) for all of the MIDI output busses.

Threadsafe

Parameters

<i>tick</i>	Provides the tick value to continue from.
-------------	---

10.36.2.18 init_clock()

```
void seq64::mastermidibase::init_clock (
    midipulse tick )
```

Calls the implementation-specific API function, and then calls [midibus::init_clock\(\)](#) for each of the MIDI output busses.

Threadsafe

Parameters

<i>tick</i>	Provides the tick value with which to initialize the buss clock.
-------------	--

10.36.2.19 emit_clock()

```
void seq64::mastermidibase::emit_clock (
    midipulse tick )
```

Also calls the [api_clock\(\)](#) function, which does nothing for the *original* ALSA implementation and the PortMidi implementation.

Threadsafe

Parameters

<i>tick</i>	Provides the tick value with which to set the buss clock.
-------------	---

10.36.2.20 sysex()

```
void seq64::mastermidibase::sysex (
    event * ev )
```

The event is sent to all MIDI output busses. Then `flush()` is called.

There's currently no implementation-specific API function for this call.

Threadsafe

Parameters

<code>ev</code>	Provides the event pointer to be set.
-----------------	---------------------------------------

10.36.2.21 print()

```
void seq64::mastermidibase::print ( ) const
```

10.36.2.22 flush()

```
void seq64::mastermidibase::flush ( )
```

For example, ALSA provides a function to "drain" the output.

Threadsafe

10.36.2.23 panic()

```
void seq64::mastermidibase::panic ( )
```

Adapted from Oli Kester's Kepler34 project.

10.36.2.24 set_sequence_input()

```
void seq64::mastermidibase::set_sequence_input (
    bool state,
    sequence * seq )
```

The portmidi version only sets `m_seq` and `m_dumping_input`, but it seems like all the code below would apply to any mastermidibus.

Threadsafe

Parameters

<i>state</i>	Provides the dumping-input (recording) state to be set. This value, as used in seqedit, can represent the state of the thru button or the record button.
<i>seq</i>	Provides the sequence object to be logged as the mastermidibase's sequence. Can also be used to set a null pointer, to disable the sequence setting.

10.36.2.25 dump_midi_input()

```
void seq64::mastermidibase::dump_midi_input (
    event ev )
```

It should be called only if m_filter_by_channel is set.

If we have more than one sequence recording, and the channel-match feature [the [sequence::channels_match\(\)](#) function] is disabled, then only the first sequence will get the events. So now we add an addition call to the new [sequence::channel_match\(\)](#) function.

Parameters

<i>ev</i>	The event that was recorded, passed as a copy. (Do we really need a copy?)
-----------	--

10.36.2.26 get_midi_out_bus_name()

```
std::string seq64::mastermidibase::get_midi_out_bus_name (
    bussbyte bus )
```

This function is used for display purposes, and is also written to the options ("rc") file.

This function adds the retrieval of client and port numbers that are not needed in the portmidi implementation, but seem generally useful to support in all implementations.

Also, if the client name is already part of the port name, as in "client:client port 0", then we remove the "client:" portion to make the listing look cleaner.

Parameters

<i>bus</i>	Provides the output buss number. Checked before usage. Actually should now be an index number
------------	---

Returns

Returns the buss name as a standard C++ string. Also contains an indication that the buss is disconnected or unconnected. If the buss number is illegal, this string is empty.

10.36.2.27 get_midi_in_bus_name()

```
std::string seq64::mastermidibase::get_midi_in_bus_name (
    bussbyte bus )
```

This function adds the retrieval of client and port numbers that are not needed in the portmidi implementation, but seem generally useful to support in all implementations.

Parameters

<i>bus</i>	Provides the input buss number.
------------	---------------------------------

Returns

Returns the buss name as a standard C++ string. Also contains an indication that the buss is disconnected or unconnected.

10.36.2.28 poll_for_midi()

```
int seq64::mastermidibase::poll_for_midi ( )
```

This base-class implementation could be made identical to portmidi's [poll_for_midi\(\)](#) function, maybe. But currently it is better just call the implementation-specific API function.

Warning

Do we need to use a mutex lock? No! It causes a deadlock!!!

Returns

Returns the result of the poll, or 0 if the API is not supported.

10.36.2.29 is_more_input()

```
bool seq64::mastermidibase::is_more_input ( )
```

Calls the implementation-specific API function.

Note that the ALSA implementation calls a single "input-pending" function, while the PortMidi implementation loops through all of the input midibus objects, calling the [poll_for_midi\(\)](#) function of each.

Threadsafe**Returns**

Returns true if ALSA is supported, and the returned size is greater than 0, or false otherwise.

10.36.2.30 get_midi_event()

```
bool seq64::mastermidibase::get_midi_event (
    event * ev )
```

Parameters

<i>ev</i>	The event to be set based on the found input event.
-----------	---

10.36.2.31 set_clock()

```
bool seq64::mastermidibase::set_clock (
    bussbyte bus,
    clock_e clocktype )
```

The legality checks are a little loose, however.

There's currently no implementation-specific API function here.

Threadsafe

Parameters

<i>bus</i>	The buss to start play on. Checked before usage.
<i>clocktype</i>	The type of clock to be set, either "off", "pos", or "mod", as noted in the midibus_common module.

10.36.2.32 set_input()

```
bool seq64::mastermidibase::set_input (
    bussbyte bus,
    bool inputting )
```

Why is another buss-count constant, and a global one at that, being used? And I thought there was only one input buss anyway! Well, there is only one ALSA input buss, but more can be used with JACK, apparently.

There's currently no implementation-specific API function here.

Threadsafe

Parameters

<i>bus</i>	Provides the buss number.
<i>inputting</i>	True if the input bus will be inputting MIDI data.

Returns

Returns true if the input buss array item could be set and then saved into the status container.

10.36.2.33 get_input()

```
bool seq64::mastermidibase::get_input (
    bussbyte bus )
```

There's currently no implementation-specific API function here.

Parameters

<i>bus</i>	Provides the buss number.
------------	---------------------------

Returns

Returns the value of the busarray::get_input(bus) call.

10.36.2.34 is_input_system_port()

```
bool seq64::mastermidibase::is_input_system_port (
    bussbyte bus )
```

Parameters

<i>bus</i>	Provides the buss number.
------------	---------------------------

Returns

Returns the value of the busarray::get_input(bus) call.

10.36.2.35 get_clock()

```
clock_e seq64::mastermidibase::get_clock (
    bussbyte bus )
```

There's currently no implementation-specific API function here.

Parameters

<i>bus</i>	Provides the buss number to read. Checked before usage.
------------	---

Returns

If the buss number is legal, and the buss is active, then its clock setting is returned. Otherwise, e_clock_↔ disabled is returned.

10.36.2.36 set_ppqn()

```
void seq64::mastermidibase::set_ppqn (
    int ppqn )
```

Then call the implementation-specific API function to complete the PPQN setting.

Threadsafe

Parameters

<i>ppqn</i>	The PPQN value to be set.
-------------	---------------------------

10.36.2.37 set_beats_per_minute()

```
void seq64::mastermidibase::set_beats_per_minute (
    midibpm bpm )
```

Then call the implementation-specific API function to complete the BPM setting.

Threadsafe

Parameters

<i>bpm</i>	Provides the beats-per-minute value to set.
------------	---

10.36.2.38 set_port_statuses()

```
void seq64::mastermidibase::set_port_statuses (
    const std::vector< clock_e > & clocks,
    const std::vector< bool > & inputs ) [inline], [protected]
```

Used in the perform class to pass the settings read from the "rc" file to here. There is an converse function defined below.

10.36.2.39 get_port_statuses()

```
void seq64::mastermidibase::get_port_statuses (
    std::vector< clock_e > & clocks,
    std::vector< bool > & inputs ) [inline], [protected]
```

Used in the perform class to pass the settings read from the "rc" file to here. There is an converse function defined above.

10.36.2.40 clock()

```
clock_e seq64::mastermidibase::clock (
    int bus ) [inline], [protected]
```

10.36.2.41 input()

```
bool seq64::mastermidibase::input (
    int bus ) [inline], [protected]
```

10.36.2.42 activate()

```
virtual bool seq64::mastermidibase::activate ( ) [inline], [protected], [virtual]
```

Currently re-implemented only in the rtmidi JACK API.

Reimplemented in [seq64::mastermidibus](#).

10.36.2.43 api_init()

```
virtual void seq64::mastermidibase::api_init (
    int ppqn,
    midibpm bpm ) [protected], [pure virtual]
```

Implemented in [seq64::mastermidibus](#).

10.36.2.44 api_start()

```
virtual void seq64::mastermidibase::api_start ( ) [inline], [protected], [virtual]
```

10.36.2.45 api_continue_from()

```
virtual void seq64::mastermidibase::api_continue_from (
    midipulse ) [inline], [protected], [virtual]
```

10.36.2.46 api_init_clock()

```
virtual void seq64::mastermidibase::api_init_clock (
    midipulse ) [inline], [protected], [virtual]
```

10.36.2.47 api_stop()

```
virtual void seq64::mastermidibase::api_stop ( ) [inline], [protected], [virtual]
```

10.36.2.48 api_set_ppqn()

```
virtual void seq64::mastermidibase::api_set_ppqn (
    int ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#).

10.36.2.49 api_set_beats_per_minute()

```
virtual void seq64::mastermidibase::api_set_beats_per_minute (
    midibpm ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#).

10.36.2.50 api_flush()

```
virtual void seq64::mastermidibase::api_flush ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#).

10.36.2.51 api_clock()

```
virtual void seq64::mastermidibase::api_clock ( ) [inline], [protected], [virtual]
```

10.36.2.52 `api_port_start()`

```
virtual void seq64::mastermidibase::api_port_start (
    int ,
    int ) [inline], [protected], [virtual]
```

10.36.2.53 `api_get_midi_event()`

```
virtual bool seq64::mastermidibase::api_get_midi_event (
    event * inev ) [protected], [pure virtual]
```

Implemented in [seq64::mastermidibus](#).

10.36.2.54 `api_poll_for_midi()`

```
int seq64::mastermidibase::api_poll_for_midi ( ) [protected], [virtual]
```

This implementation adds a millisecond of sleep time unless more than two events are pending. Some input devices may need to override this function.

For a quick threadsafe check, call [is_more_input\(\)](#) instead. But see the warning in the non-API [poll_for_midi\(\)](#) function.

Returns

Returns the number of events found by the first successful poll of the array of input busses (`m_inbus_array`).

Reimplemented in [seq64::mastermidibus](#).

10.36.2.55 `save_clock()`

```
bool seq64::mastermidibase::save_clock (
    bussbyte bus,
    clock_e clock ) [private]
```

Parameters

<i>bus</i>	Provides the desired buss to be set.
<i>clock</i>	Provides the clocking value to set.

Returns

Returns true if the buss value is valid.

10.36.2.56 save_input()

```
bool seq64::mastermidibase::save_input (
    bussbyte bus,
    bool inputing ) [private]
```

Now, we were checking this bus number against the size of the vector as gotten from the perform object, which it got the from the "rc" file's [midi-input] section. However, the "rc" file won't necessarily match what is on the system now. So we might have to adjust.

Do we also have to adjust the perform's vector?

Parameters

<i>bus</i>	Provides the buss number.
<i>inputing</i>	True if the input bus will be inputting MIDI data.

Returns

Returns true, always.

10.36.3 Friends And Related Function Documentation

10.36.3.1 perform

```
friend class perform [friend]
```

10.36.3.2 midi_alsa_info

```
friend class midi_alsa_info [friend]
```

10.36.4 Field Documentation

10.36.4.1 m_max_busses

```
int seq64::mastermidibase::m_max_busses [protected]
```

Set to c_max_busses (SEQ64_DEFAULT_BUSS_MAX = 32) for now.

10.36.4.2 m_bus_announce

```
midibus* seq64::mastermidibase::m_bus_announce [protected]
```

10.36.4.3 m_inbus_array

```
busarray seq64::mastermidibase::m_inbus_array [protected]
```

10.36.4.4 m_outbus_array

```
busarray seq64::mastermidibase::m_outbus_array [protected]
```

10.36.4.5 m_master_clocks

```
std::vector<clock_e> seq64::mastermidibase::m_master_clocks [protected]
```

10.36.4.6 m_master_inputs

```
std::vector<bool> seq64::mastermidibase::m_master_inputs [protected]
```

However, these items will be modified if the actual enumerated input ports do not match the port read from the "rc" file.

10.36.4.7 m_queue

```
int seq64::mastermidibase::m_queue [protected]
```

10.36.4.8 m_ppqn

```
int seq64::mastermidibase::m_ppqn [protected]
```

10.36.4.9 m_beats_per_minute

```
midibpm seq64::mastermidibase::m_beats_per_minute [protected]
```

We had to lengthen this name; way too easy to confuse it with "bpm" for "beats per measure".

10.36.4.10 m_dumping_input

```
bool seq64::mastermidibase::m_dumping_input [protected]
```

This value is set to true when a sequence editor window is open and the user has clicked the "record MIDI" or "thru MIDI" button.

10.36.4.11 m_vector_sequence

```
std::vector<sequence *> seq64::mastermidibase::m_vector_sequence [protected]
```

We want to make this a run-time, non-legacy option.

10.36.4.12 m_filter_by_channel

```
bool seq64::mastermidibase::m_filter_by_channel [protected]
```

10.36.4.13 m_seq

```
sequence* seq64::mastermidibase::m_seq [protected]
```

10.36.4.14 m_mutex

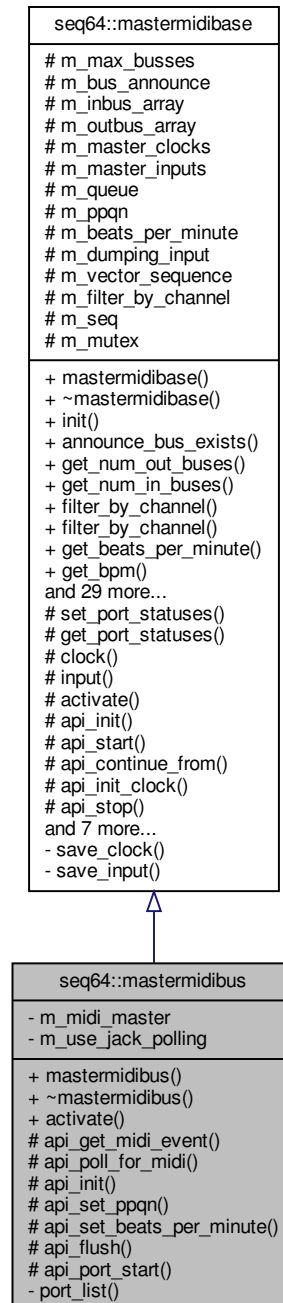
```
mutex seq64::mastermidibase::m_mutex [protected]
```

This object is passed to an automutex object that lends exception-safety to the mutex locking.

10.37 seq64::mastermidibus Class Reference

The class that "supervises" all of the midibus objects.

Inheritance diagram for seq64::mastermidibus:



Public Member Functions

- [mastermidibus](#) (int ppqn=SEQ64_USE_DEFAULT_PPQN, [midibpm](#) bpm=SEQ64_DEFAULT_BPM)
- virtual [~mastermidibus](#) ()
- virtual bool [activate](#) ()

Protected Member Functions

- virtual bool `api_get_midi_event` (`event *in`)
- virtual int `api_poll_for_midi` ()
- virtual void `api_init` (int ppqn, midibpm bpm)
- virtual void `api_set_ppqn` (int p)
Provides MIDI API-specific functionality for the `set_ppqn()` function.
- virtual void `api_set_beats_per_minute` (midibpm b)
Provides MIDI API-specific functionality for the `set_beats_per_minute()` function.
- virtual void `api_flush` ()
- virtual void `api_port_start` (mastermidibus &masterbus, int bus, int port)

Private Member Functions

- void `port_list` (const std::string &tag)

Private Attributes

- `rtmidi_info m_midi_master`
Holds the basic MIDI input and output information for later re-use in the construction of midibus objects.
- bool `m_use_jack_polling`
Indicates we are running with JACK MIDI enabled, and need to use each port's ability to poll for and get MIDI events, rather than use ALSA's method of calling functions from the "MIDI master" object.

Friends

- class `midi_alsa_info`
- class `midi_jack_info`

Additional Inherited Members

10.37.1 Detailed Description

This implementation uses the PortMidi library, which supports Linux and Windows, but not JACK or Mac OSX.

10.37.2 Constructor & Destructor Documentation

10.37.2.1 mastermidibus()

```
seq64::mastermidibus::mastermidibus (
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

10.37.2.2 ~mastermidibus()

```
virtual seq64::mastermidibus::~~mastermidibus ( ) [virtual]
```

10.37.3 Member Function Documentation

10.37.3.1 activate()

```
virtual bool seq64::mastermidibus::activate ( ) [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

10.37.3.2 api_get_midi_event()

```
virtual bool seq64::mastermidibus::api_get_midi_event (
    event * in ) [protected], [virtual]
```

Implements [seq64::mastermidibase](#).

10.37.3.3 api_poll_for_midi()

```
virtual int seq64::mastermidibus::api_poll_for_midi ( ) [protected], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

10.37.3.4 api_init()

```
virtual void seq64::mastermidibus::api_init (
    int ppqn,
    midibpm bpm ) [protected], [virtual]
```

Implements [seq64::mastermidibase](#).

10.37.3.5 api_set_ppqn()

```
virtual void seq64::mastermidibus::api_set_ppqn (
    int p ) [inline], [protected], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

10.37.3.6 api_set_beats_per_minute()

```
virtual void seq64::mastermidibus::api_set_beats_per_minute (
    midibpm b ) [inline], [protected], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

10.37.3.7 api_flush()

```
virtual void seq64::mastermidibus::api_flush ( ) [inline], [protected], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

10.37.3.8 api_port_start()

```
virtual void seq64::mastermidibus::api_port_start (
    mastermidibus & masterbus,
    int bus,
    int port ) [inline], [protected], [virtual]
```

10.37.3.9 port_list()

```
void seq64::mastermidibus::port_list (
    const std::string & tag ) [private]
```

10.37.4 Friends And Related Function Documentation

10.37.4.1 midi_alsa_info

```
friend class midi_alsa_info [friend]
```

10.37.4.2 midi_jack_info

```
friend class midi_jack_info [friend]
```

10.37.5 Field Documentation

10.37.5.1 m_midi_master

```
rtmidi_info seq64::mastermidibus::m_midi_master [private]
```

10.37.5.2 m_use_jack_polling

```
bool seq64::mastermidibus::m_use_jack_polling [private]
```

10.38 seq64::editable_event::meta_length_t Struct Reference

Provides a type that contains the pair of values needed to get the Meta event's data length.

Data Fields

- unsigned short [event_value](#)
Holds a midibyte value (0x00 to 0xFF) or SEQ64_END_OF_MIDIBYTE_TABLE to indicate the end of an array of [name_value_t](#) items.
- unsigned short [event_length](#)
Holds the length expected for the Meta event, or 0 if it does not apply to the Meta event.

10.38.1 Field Documentation

10.38.1.1 event_value

```
unsigned short seq64::editable_event::meta_length_t::event_value
```

This field has the same meaning as the event_value of the [name_value_t](#) type.

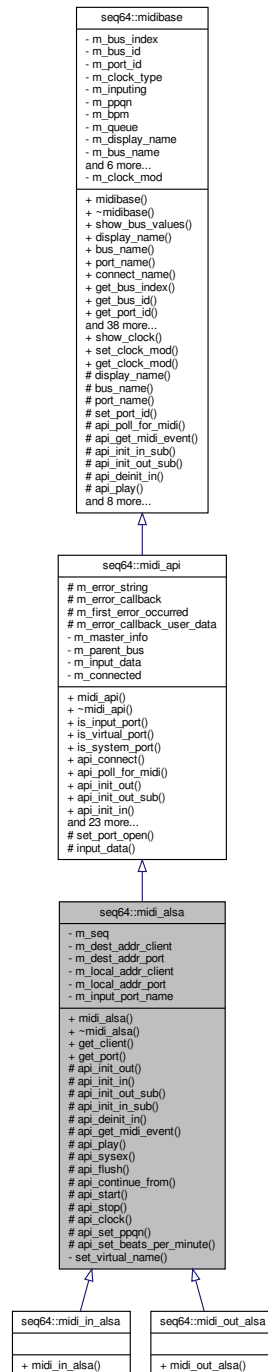
10.38.1.2 event_length

```
unsigned short seq64::editable_event::meta_length_t::event_length
```

10.39 seq64::midi_alsa Class Reference

This class implements the ALSA version of the [midi_api](#).

Inheritance diagram for seq64::midi_alsa:



Public Member Functions

- [midi_alsa](#) (midibus &parentbus, [midi_info](#) &masterinfo)

- virtual `~midi_alsa ()`
- virtual int `get_client ()` const
'Getter' function for member m_dest_addr_client The address of client.
- virtual int `get_port ()` const
'Getter' function for member m_dest_addr_port Can we replace it with `get_port_id()`?

Protected Member Functions

- virtual bool `api_init_out ()`
- virtual bool `api_init_in ()`
- virtual bool `api_init_out_sub ()`
- virtual bool `api_init_in_sub ()`
- virtual bool `api_deinit_in ()`
- virtual bool `api_get_midi_event (event *)`
ALSA get MIDI events via the `midi_alsa_info` object at present.
- virtual void `api_play (event *e24, midibyte channel)`
- virtual void `api_sysex (event *e24)`
- virtual void `api_flush ()`
- virtual void `api_continue_from (midipulse tick, midipulse beats)`
- virtual void `api_start ()`
- virtual void `api_stop ()`
- virtual void `api_clock (midipulse tick)`
- virtual void `api_set_ppqn (int ppqn)`
- virtual void `api_set_beats_per_minute (midibpm bpm)`

Private Member Functions

- bool `set_virtual_name (int portid, const std::string &portname)`

Private Attributes

- `snd_seq_t *const m_seq`
ALSA sequencer client handle.
- const int `m_dest_addr_client`
Destination address of client.
- const int `m_dest_addr_port`
Destination port of client.
- const int `m_local_addr_client`
Local address of client.
- int `m_local_addr_port`
Local port of client.
- const std::string `m_input_port_name`
Holds the port name for the ALSA MIDI input port.

Additional Inherited Members

10.39.1 Constructor & Destructor Documentation

10.39.1.1 midi_alsa()

```
seq64::midi_alsa::midi_alsa (
    midibus & parentbus,
    midi_info & masterinfo )
```

10.39.1.2 ~midi_alsa()

```
virtual seq64::midi_alsa::~~midi_alsa ( ) [virtual]
```

10.39.2 Member Function Documentation

10.39.2.1 get_client()

```
virtual int seq64::midi_alsa::get_client ( ) const [inline], [virtual]
```

Can we replace it with `get_client_id()`?

10.39.2.2 get_port()

```
virtual int seq64::midi_alsa::get_port ( ) const [inline], [virtual]
```

10.39.2.3 api_init_out()

```
virtual bool seq64::midi_alsa::api_init_out ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.4 api_init_in()

```
virtual bool seq64::midi_alsa::api_init_in ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.5 `api_init_out_sub()`

```
virtual bool seq64::midi_alsa::api_init_out_sub ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.6 `api_init_in_sub()`

```
virtual bool seq64::midi_alsa::api_init_in_sub ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.7 `api_deinit_in()`

```
virtual bool seq64::midi_alsa::api_deinit_in ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.8 `api_get_midi_event()`

```
virtual bool seq64::midi_alsa::api_get_midi_event (
    event * ) [inline], [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.9 `api_play()`

```
virtual void seq64::midi_alsa::api_play (
    event * e24,
    midibyte channel ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.10 `api_sysex()`

```
virtual void seq64::midi_alsa::api_sysex (
    event * e24 ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.11 api_flush()

```
virtual void seq64::midi_alsa::api_flush ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.12 api_continue_from()

```
virtual void seq64::midi_alsa::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.13 api_start()

```
virtual void seq64::midi_alsa::api_start ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.14 api_stop()

```
virtual void seq64::midi_alsa::api_stop ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.15 api_clock()

```
virtual void seq64::midi_alsa::api_clock (
    midipulse tick ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.16 api_set_ppqn()

```
virtual void seq64::midi_alsa::api_set_ppqn (
    int ppqn ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.17 `api_set_beats_per_minute()`

```
virtual void seq64::midi_alsa::api_set_beats_per_minute (
    midibpm bpm ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.39.2.18 `set_virtual_name()`

```
bool seq64::midi_alsa::set_virtual_name (
    int portid,
    const std::string & portname ) [private]
```

10.39.3 Field Documentation

10.39.3.1 `m_seq`

```
snd_seq_t* const seq64::midi_alsa::m_seq [private]
```

10.39.3.2 `m_dest_addr_client`

```
const int seq64::midi_alsa::m_dest_addr_client [private]
```

Could potentially be replaced by [midibase::m_bus_id](#).

10.39.3.3 `m_dest_addr_port`

```
const int seq64::midi_alsa::m_dest_addr_port [private]
```

Could potentially be replaced by [midibase::m_port_id](#).

10.39.3.4 `m_local_addr_client`

```
const int seq64::midi_alsa::m_local_addr_client [private]
```

10.39.3.5 m_local_addr_port

```
int seq64::midi_alsa::m_local_addr_port [private]
```

10.39.3.6 m_input_port_name

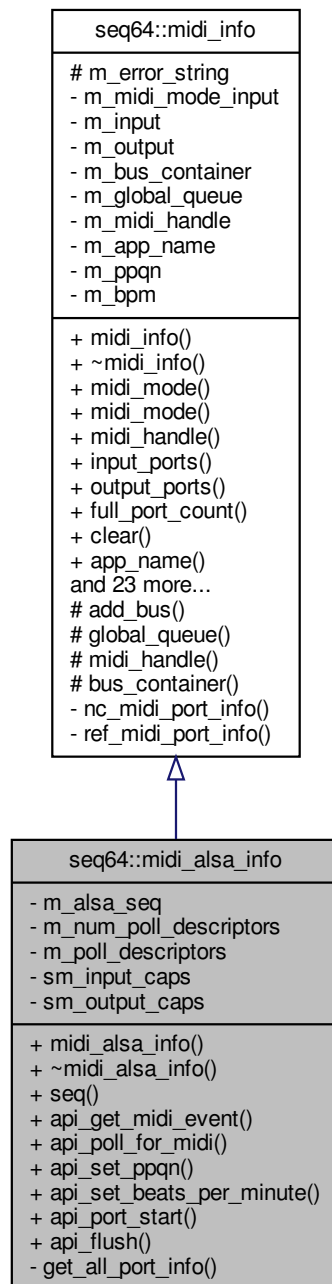
```
const std::string seq64::midi_alsa::m_input_port_name [private]
```

It is derived from the (optionally configured) official client name for the application with the word "in" appended.

10.40 seq64::midi_alsa_info Class Reference

The class for handling ALSA MIDI input.

Inheritance diagram for seq64::midi_alsa_info:



Public Member Functions

- [midi_alsa_info](#) (const std::string &appname, int [ppqn](#)=SEQ64_DEFAULT_PPQN, [midibpm bpm](#)=SEQ64_DEFAULT_BPM)
- virtual [~midi_alsa_info](#) ()
- `snd_seq_t * seq ()`

'Getter' function for member `m_alsa_seq` This is the platform-specific version of [midi_handle\(\)](#).

- virtual bool [api_get_midi_event](#) (event *inev)
- virtual int [api_poll_for_midi](#) ()
- virtual void [api_set_ppqn](#) (int p)
- virtual void [api_set_beats_per_minute](#) (midibpm b)
- virtual void [api_port_start](#) (mastermidibus &masterbus, int bus, int port)
- virtual void [api_flush](#) ()

Private Member Functions

- virtual int [get_all_port_info](#) ()

Private Attributes

- snd_seq_t * [m_alsa_seq](#)
Holds the ALSA sequencer client pointer so that it can be used by the midibus objects.
- int [m_num_poll_descriptors](#)
The number of descriptors for polling.
- struct pollfd * [m_poll_descriptors](#)
Points to the list of descriptors for polling.

Static Private Attributes

- static unsigned [sm_input_caps](#)
Flags that denote queries for input (read) ports.
- static unsigned [sm_output_caps](#)
Flags that denote queries for output (write) ports.

Additional Inherited Members

10.40.1 Constructor & Destructor Documentation

10.40.1.1 midi_alsa_info()

```
seq64::midi_alsa_info::midi_alsa_info (
    const std::string & appname,
    int ppqn = SEQ64_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

10.40.1.2 ~midi_alsa_info()

```
virtual seq64::midi_alsa_info::~~midi_alsa_info ( ) [virtual]
```

10.40.2 Member Function Documentation

10.40.2.1 seq()

```
snd_seq_t* seq64::midi_alsa_info::seq ( ) [inline]
```

10.40.2.2 api_get_midi_event()

```
virtual bool seq64::midi_alsa_info::api_get_midi_event (
    event * inev ) [virtual]
```

Implements [seq64::midi_info](#).

10.40.2.3 api_poll_for_midi()

```
virtual int seq64::midi_alsa_info::api_poll_for_midi ( ) [virtual]
```

Implements [seq64::midi_info](#).

10.40.2.4 api_set_ppqn()

```
virtual void seq64::midi_alsa_info::api_set_ppqn (
    int p ) [virtual]
```

Reimplemented from [seq64::midi_info](#).

10.40.2.5 api_set_beats_per_minute()

```
virtual void seq64::midi_alsa_info::api_set_beats_per_minute (
    midibpm b ) [virtual]
```

Reimplemented from [seq64::midi_info](#).

10.40.2.6 api_port_start()

```
virtual void seq64::midi_alsa_info::api_port_start (
    mastermidibus & masterbus,
    int bus,
    int port ) [virtual]
```

Reimplemented from [seq64::midi_info](#).

10.40.2.7 api_flush()

```
virtual void seq64::midi_alsa_info::api_flush ( ) [virtual]
```

Implements [seq64::midi_info](#).

10.40.2.8 get_all_port_info()

```
virtual int seq64::midi_alsa_info::get_all_port_info ( ) [private], [virtual]
```

Implements [seq64::midi_info](#).

10.40.3 Field Documentation

10.40.3.1 sm_input_caps

```
unsigned seq64::midi_alsa_info::sm_input_caps [static], [private]
```

10.40.3.2 sm_output_caps

```
unsigned seq64::midi_alsa_info::sm_output_caps [static], [private]
```

10.40.3.3 m_alsa_seq

```
snd_seq_t* seq64::midi_alsa_info::m_alsa_seq [private]
```

This is actually an opaque pointer; there is no way to get the actual fields in this structure; they can only be accessed through functions in the ALSA API.

10.40.3.4 m_num_poll_descriptors

```
int seq64::midi_alsa_info::m_num_poll_descriptors [private]
```

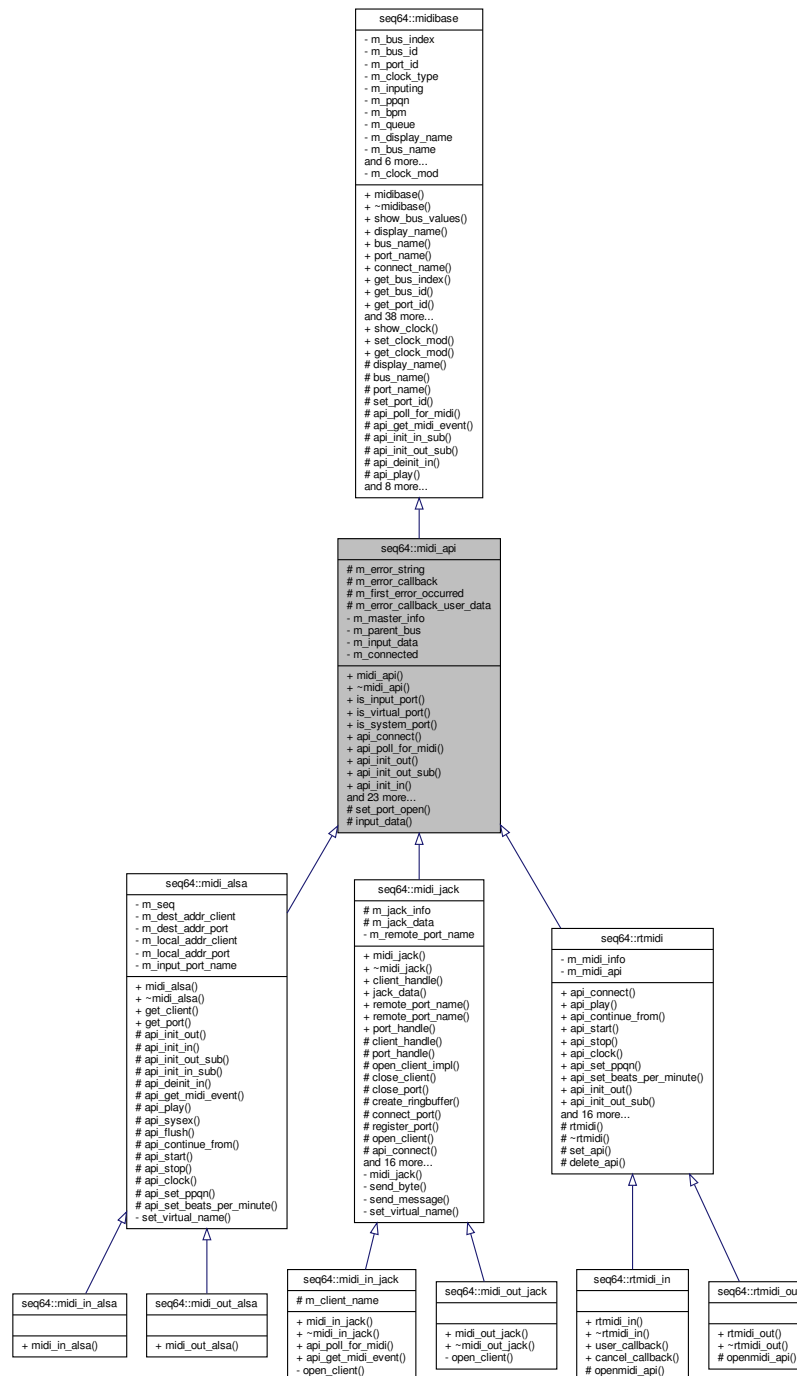
10.40.3.5 m_poll_descriptors

```
struct pollfd* seq64::midi_alsa_info::m_poll_descriptors [private]
```

10.41 seq64::midi_api Class Reference

Subclasses of `midi_in_api` and `midi_out_api` contain all API- and OS-specific code necessary to fully implement the `rtmidi` API.

Inheritance diagram for seq64::midi_api:



Public Member Functions

- [midi_api](#) (midibus &parentbus, [midi_info](#) &masterinfo)
- virtual [~midi_api](#) ()
- bool [is_input_port](#) () const
- bool [is_virtual_port](#) () const
- bool [is_system_port](#) () const

- virtual bool `api_connect ()`
No code; only `midi_jack` overrides this function at present.
- virtual int `api_poll_for_midi ()`
- virtual bool `api_init_out ()=0`
- virtual bool `api_init_out_sub ()=0`
- virtual bool `api_init_in ()=0`
- virtual bool `api_init_in_sub ()=0`
- virtual bool `api_deinit_in ()=0`
- virtual bool `api_get_midi_event (event *)=0`
- virtual void `api_play (event *e24, midibyte channel)=0`
- virtual void `api_sysex (event *e24)=0`
- virtual void `api_continue_from (midipulse tick, midipulse beats)=0`
- virtual void `api_start ()=0`
- virtual void `api_stop ()=0`
- virtual void `api_flush ()=0`
- virtual void `api_clock (midipulse tick)=0`
- virtual void `api_set_ppqn (int ppqn)=0`
- virtual void `api_set_beats_per_minute (midibpm bpm)=0`
- virtual std::string `api_get_bus_name ()`
- virtual std::string `api_get_port_name ()`
- bool `is_port_open ()` const
'Getter' function for member `m_connected`
- `midi_info & master_info ()`
'Getter' function for member `m_master_info`
- const `midi_info & master_info ()` const
'Getter' function for member `m_master_info` The const version.
- `midibus & parent_bus ()`
'Getter' function for member `m_parent_bus`
- const `midibus & parent_bus ()` const
'Getter' function for member `m_parent_bus` The const version.
- void `master_midi_mode (bool input)`
- void `error (rterror::Type type, const std::string &errorstring)`
- void `user_callback (rtmidi_callback_t callback, void *userdata)`
- void `cancel_callback ()`

Protected Member Functions

- void `set_port_open ()`
'Setter' function for member `m_connected`
- `rtmidi_in_data * input_data ()`
'Getter' function for member `&m_input_data`

Protected Attributes

- std::string `m_error_string`
Holds the last error message, if in force.
- `rterror_callback m_error_callback`
Holds the error callback function pointer, if any.
- bool `m_first_error_occurred`
Indicates that the first error has happened.
- void * `m_error_callback_user_data`
Holds data needed by the error-callback.

Private Attributes

- [midi_info](#) & [m_master_info](#)
Contains information about the ports (system or client) enumerated by the API.
- [midibus](#) & [m_parent_bus](#)
Contains a reference to the parent midibus/midibase object.
- [rtmidi_in_data](#) [m_input_data](#)
Although this really is useful only for MIDI input objects, the split of the [midi_api](#) is not as convenient for re-use as is the split for derived classes like [midi_in_jack](#)/[midi_out_jack](#).
- bool [m_connected](#)
Set to true if the port was opened, activated, and connected without issue.

Additional Inherited Members

10.41.1 Detailed Description

Note that [midi_in_api](#) and [midi_out_api](#) are abstract base classes and cannot be explicitly instantiated. [rtmidi_in](#) and [rtmidi_out](#) will create instances of a [midi_in_api](#) or [midi_out_api](#) subclass.

10.41.2 Constructor & Destructor Documentation

10.41.2.1 [midi_api\(\)](#)

```
seq64::midi_api::midi_api (
    midibus & parentbus,
    midi\_info & masterinfo )
```

10.41.2.2 [~midi_api\(\)](#)

```
virtual seq64::midi_api::~~midi_api ( ) [virtual]
```

10.41.3 Member Function Documentation

10.41.3.1 [is_input_port\(\)](#)

```
bool seq64::midi_api::is_input_port ( ) const
```

10.41.3.2 is_virtual_port()

```
bool seq64::midi_api::is_virtual_port ( ) const
```

10.41.3.3 is_system_port()

```
bool seq64::midi_api::is_system_port ( ) const
```

10.41.3.4 api_connect()

```
virtual bool seq64::midi_api::api_connect ( ) [inline], [virtual]
```

Reimplemented in [seq64::midi_jack](#), and [seq64::rtmidi](#).

10.41.3.5 api_poll_for_midi()

```
virtual int seq64::midi_api::api_poll_for_midi ( ) [virtual]
```

Reimplemented from [seq64::midibase](#).

Reimplemented in [seq64::midi_in_jack](#), and [seq64::rtmidi](#).

10.41.3.6 api_init_out()

```
virtual bool seq64::midi_api::api_init_out ( ) [pure virtual]
```

Implements [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.7 api_init_out_sub()

```
virtual bool seq64::midi_api::api_init_out_sub ( ) [pure virtual]
```

Reimplemented from [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.8 api_init_in()

```
virtual bool seq64::midi_api::api_init_in ( ) [pure virtual]
```

Implements [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.9 api_init_in_sub()

```
virtual bool seq64::midi_api::api_init_in_sub ( ) [pure virtual]
```

Reimplemented from [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.10 api_deinit_in()

```
virtual bool seq64::midi_api::api_deinit_in ( ) [pure virtual]
```

Reimplemented from [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.11 api_get_midi_event()

```
virtual bool seq64::midi_api::api_get_midi_event (
    event * ) [pure virtual]
```

Reimplemented from [seq64::midibase](#).

Implemented in [seq64::midi_in_jack](#), [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.12 api_play()

```
virtual void seq64::midi_api::api_play (
    event * e24,
    midibyte channel ) [pure virtual]
```

Implements [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.13 `api_sysex()`

```
virtual void seq64::midi_api::api_sysex (
    event * e24 ) [pure virtual]
```

Reimplemented from [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.14 `api_continue_from()`

```
virtual void seq64::midi_api::api_continue_from (
    midipulse tick,
    midipulse beats ) [pure virtual]
```

Implements [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.15 `api_start()`

```
virtual void seq64::midi_api::api_start ( ) [pure virtual]
```

Implements [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.16 `api_stop()`

```
virtual void seq64::midi_api::api_stop ( ) [pure virtual]
```

Implements [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.17 `api_flush()`

```
virtual void seq64::midi_api::api_flush ( ) [pure virtual]
```

Reimplemented from [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.18 api_clock()

```
virtual void seq64::midi_api::api_clock (
    midipulse tick ) [pure virtual]
```

Implements [seq64::midibase](#).

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.19 api_set_ppqn()

```
virtual void seq64::midi_api::api_set_ppqn (
    int ppqn ) [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.20 api_set_beats_per_minute()

```
virtual void seq64::midi_api::api_set_beats_per_minute (
    midibpm bpm ) [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), and [seq64::rtmidi](#).

10.41.3.21 api_get_bus_name()

```
virtual std::string seq64::midi_api::api_get_bus_name ( ) [inline], [virtual]
```

10.41.3.22 api_get_port_name()

```
virtual std::string seq64::midi_api::api_get_port_name ( ) [inline], [virtual]
```

Reimplemented in [seq64::midi_jack](#).

10.41.3.23 is_port_open()

```
bool seq64::midi_api::is_port_open ( ) const [inline]
```

10.41.3.24 master_info() [1/2]

```
midi_info& seq64::midi_api::master_info ( ) [inline]
```

10.41.3.25 master_info() [2/2]

```
const midi_info& seq64::midi_api::master_info ( ) const [inline]
```

10.41.3.26 parent_bus() [1/2]

```
midibus& seq64::midi_api::parent_bus ( ) [inline]
```

10.41.3.27 parent_bus() [2/2]

```
const midibus& seq64::midi_api::parent_bus ( ) const [inline]
```

10.41.3.28 master_midi_mode()

```
void seq64::midi_api::master_midi_mode (
    bool input )
```

10.41.3.29 error()

```
void seq64::midi_api::error (
    rterror::Type type,
    const std::string & errorstring )
```

10.41.3.30 user_callback()

```
void seq64::midi_api::user_callback (
    rtmidi_callback_t callback,
    void * userdata )
```


10.41.3.31 cancel_callback()

```
void seq64::midi_api::cancel_callback ( )
```

10.41.3.32 set_port_open()

```
void seq64::midi_api::set_port_open ( ) [inline], [protected]
```

10.41.3.33 input_data()

```
rtmidi_in_data* seq64::midi_api::input_data ( ) [inline], [protected]
```

10.41.4 Field Documentation

10.41.4.1 m_master_info

```
rtmidi_info& seq64::midi_api::m_master_info [private]
```

10.41.4.2 m_parent_bus

```
rtmidi_bus& seq64::midi_api::m_parent_bus [private]
```

This object is needed to get parameters that are peculiar to the port as it is actually set up, rather than information from the [rtmidi_info](#) object.

10.41.4.3 m_input_data

```
rtmidi_in_data seq64::midi_api::m_input_data [private]
```

10.41.4.4 m_connected

```
bool seq64::midi_api::m_connected [private]
```

10.41.4.5 m_error_string

```
std::string seq64::midi_api::m_error_string [protected]
```

This is an original RtMidi concept.

10.41.4.6 m_error_callback

```
rterror_callback seq64::midi_api::m_error_callback [protected]
```

This is an original RtMidi concept.

10.41.4.7 m_first_error_occurred

```
bool seq64::midi_api::m_first_error_occurred [protected]
```

This is an original RtMidi concept. I have to confess I am not sure how it is/should be used, yet.

10.41.4.8 m_error_callback_user_data

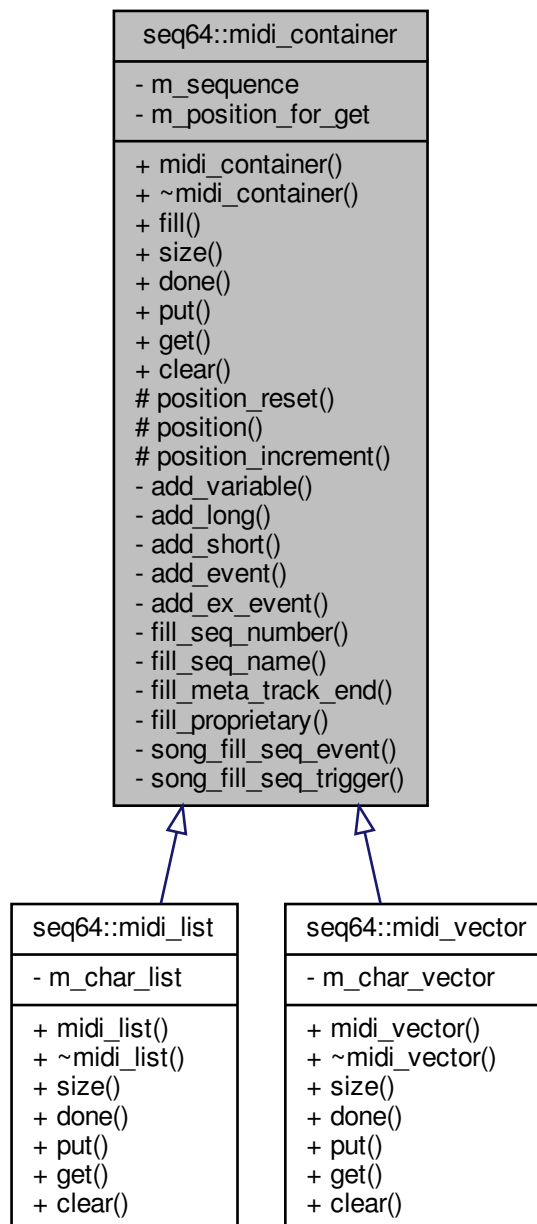
```
void* seq64::midi_api::m_error_callback_user_data [protected]
```

This is an original RtMidi concept. I have to confess I am not sure how it is/should be used, yet.

10.42 seq64::midi_container Class Reference

This class is the abstract base class for a container of MIDI track information.

Inheritance diagram for seq64::midi_container:



Public Member Functions

- `midi_container` (`sequence` &seq)
Fills in the few members of this class.
- virtual `~midi_container` ()
A rote constructor needed for a base class.
- void `fill` (int tracknumber, const `perform` &p, bool doseqspec=true)

This function fills the given track (sequence) with MIDI data from the current sequence, preparatory to writing it to a file.

- virtual std::size_t [size](#) () const
Returns the size of the container, in midibytes.
- virtual bool [done](#) () const
Instead of checking for the size of the container when "emptying" it [see the [midifile::write\(\)](#) function], use this function, which is overridden to match the type of container being used.
- virtual void [put](#) (midibyte b)=0
Provides a way to add a MIDI byte into the container.
- virtual midibyte [get](#) () const =0
Provide a way to get the next byte from the container.
- virtual void [clear](#) ()=0
Provides a way to clear the container.

Protected Member Functions

- unsigned [position_reset](#) () const
'Setter' function for member m_position_for_get Sets the position to 0 and then returns that value.
- unsigned [position](#) () const
'Getter' function for member m_position_for_get Returns the current position.
- void [position_increment](#) () const
'Getter' function for member m_position_for_get Increments the current position.

Private Member Functions

- void [add_variable](#) (midipulse v)
This function masks off the lower 8 bits of the long parameter, then shifts it right 7, and, if there are still set bits, it encodes it into the buffer in reverse order.
- void [add_long](#) (midipulse x)
Adds a long value (a MIDI pulse/tick value) to the container.
- void [add_short](#) (midishort x)
Adds a short value (two bytes) to the container.
- void [add_event](#) (const event &e, midipulse deltatime)
Adds an event to the container.
- void [add_ex_event](#) (const event &e, midipulse deltatime)
Adds the bytes of a SysEx or Meta MIDI event.
- void [fill_seq_number](#) (int seq)
Fills in the sequence number.
- void [fill_seq_name](#) (const std::string &name)
Fills in the sequence name.
- void [fill_meta_track_end](#) (midipulse deltatime)
- void [fill_proprietary](#) ()
Fills in the Sequencer64-specific information for the current sequence: The MIDI buss number, the time-signature, and the MIDI channel.
- midipulse [song_fill_seq_event](#) (const trigger &trig, midipulse prev_timestamp)
Fills in sequence events based on the trigger and events in the sequence associated with this [midi_container](#).
- void [song_fill_seq_trigger](#) (const trigger &trig, midipulse len, midipulse prev_timestamp)
Fills in the trigger for the whole sequence.

Private Attributes

- [sequence](#) & [m_sequence](#)
Provide a hook into a sequence so that we can exchange data with a sequence object.
- unsigned [m_position_for_get](#)
Provides the position in the container when making a series of [get\(\)](#) calls on the container.

Friends

- class [midifile](#)

10.42.1 Detailed Description

It is the base class for [midi_list](#) and [midi_vector](#).

10.42.2 Constructor & Destructor Documentation

10.42.2.1 midi_container()

```
seq64::midi_container::midi_container (
    sequence & seq )
```

Parameters

seq	Provides a reference to the sequence/track for which this container holds MIDI data.
---------------------	--

10.42.2.2 ~midi_container()

```
virtual seq64::midi_container::~~midi_container ( ) [inline], [virtual]
```

10.42.3 Member Function Documentation

10.42.3.1 fill()

```
void seq64::midi_container::fill (
    int track,
```

```
const perform & p,
bool doseqspec = true )
```

Note that some of the events might not come out in the same order they were stored in (we see that with program-change events). This function replaces `sequence::fill_list()`.

Now, for sequence 0, an alternate format for writing the sequencer number chunk is "FF 00 00". But that format can only occur in the first track, and the rest of the tracks then don't need a sequence number, since it is assumed to increment. This application doesn't use that shortcut.

We have noticed differences in saving files in `sets=4x8` versus `sets=8x8`, and pre-sorting the event list gets rid of some of the differences, except for the last, multi-line SeqSpec. Some event-reordering still seems to occur, though.

Stazed:

The "stazed" (seq32) code implements a function like this one using a function `sequence::fill_proprietary_list()` that we don't need for our implementation... it is part of our `midi_container::fill()` function.

Triggers:

Triggers are added by first calling `add_variable(0)`, which is needed because why?

Then `0xFF 0x7F` is written, followed by the length value, which is the number of triggers at 3 long integers per trigger, plus the 4-byte code for triggers, `c_triggers_new = 0x24240008`.

Meta and SysEx Events:

These events can now be detected and added to the list of bytes to dump. However, historically Seq24 has forced Time Signature and Set Tempo events to be written to the container, and has ignored these events (after the first occurrence). So we need to figure out what to do here yet; we need to distinguish between forcing these events and them being part of the edit.

Not threadsafe The sequence object bound to this container needs to provide the locking mechanism when calling this function.

Parameters

<i>track</i>	Provides the track number, re 0. This number is masked into the track information.
<i>p</i>	The performance object that will hold some of the parameters needed when filling the MIDI container.
<i>doseqspec</i>	If true (the default), writes out the SeqSpec information. If false, we want to write out a regular MIDI track without this information; it writes a smaller file.

To allow other sequencers to read Seq24/Sequencer64 files, we should provide the Time Signature and Tempo meta events, in the 0th (first) track (sequence). These events must precede any "real" MIDI events. They are not included if the legacy-format option is in force. We also need to skip this if tempo track support is in force.

10.42.3.2 size()

```
virtual std::size_t seq64::midi_container::size ( ) const [inline], [virtual]
```

Must be overridden in the derived class, though not pure.

Reimplemented in [seq64::midi_list](#), and [seq64::midi_vector](#).

10.42.3.3 done()

```
virtual bool seq64::midi_container::done ( ) const [inline], [virtual]
```

Reimplemented in [seq64::midi_vector](#), and [seq64::midi_list](#).

10.42.3.4 put()

```
virtual void seq64::midi_container::put (
    midibyte b ) [pure virtual]
```

The original seq24 container used an `std::list` and a `push_front` operation.

Implemented in [seq64::midi_vector](#), and [seq64::midi_list](#).

10.42.3.5 get()

```
virtual midibyte seq64::midi_container::get ( ) const [pure virtual]
```

It also increments `m_position_for_get`.

Implemented in [seq64::midi_vector](#), and [seq64::midi_list](#).

10.42.3.6 clear()

```
virtual void seq64::midi_container::clear ( ) [pure virtual]
```

Implemented in [seq64::midi_vector](#), and [seq64::midi_list](#).

10.42.3.7 position_reset()

```
unsigned seq64::midi_container::position_reset ( ) const [inline], [protected]
```

10.42.3.8 position()

```
unsigned seq64::midi_container::position ( ) const [inline], [protected]
```

10.42.3.9 position_increment()

```
void seq64::midi_container::position_increment ( ) const [inline], [protected]
```

10.42.3.10 add_variable()

```
void seq64::midi_container::add_variable (
    midipulse v ) [private]
```

This function "replaces" sequence::add_list_var().

Parameters

v	The data value to be added to the current event in the MIDI container.
---	--

10.42.3.11 add_long()

```
void seq64::midi_container::add_long (
    midipulse x ) [private]
```

What is the difference between this function and add_list_var()? This function "replaces" sequence::add_long_list(). This was a *global* internal function called addLongList(). Let's at least make it a private member now, and hew to the naming conventions of this class.

Parameters

x	Provides the timestamp (pulse value) to be added to the container.
---	--

10.42.3.12 add_short()

```
void seq64::midi_container::add_short (
    midishort x ) [private]
```


Parameters

<i>x</i>	Provides the timestamp (pulse value) to be added to the container.
----------	--

10.42.3.13 add_event()

```
void seq64::midi_container::add_event (
    const event & e,
    midipulse deltatime ) [private]
```

It handles regular MIDI events separately from "extended" (our term) MIDI events (SysEx and Meta events).

For normal MIDI events, if the sequence's MIDI channel is `EVENT_NULL_CHANNEL == 0xFF`, then it is the copy of an SMF 0 sequence that the [midi_splitter](#) created. We want to be able to save it along with the other tracks, but won't be able to read it back if all the channels are bad. So we just use the channel from the event.

SysEx and Meta events are detected and passed to the new [add_ex_event\(\)](#) function for proper dumping.

Parameters

<i>e</i>	Provides the event to be added to the container.
<i>deltatime</i>	Provides the time-location of the event.

10.42.3.14 add_ex_event()

```
void seq64::midi_container::add_ex_event (
    const event & e,
    midipulse deltatime ) [private]
```

Parameters

<i>e</i>	Provides the MIDI event to add. The caller must ensure that this is either SysEx or Meta event, using the event::is_ex_data() function.
<i>deltatime</i>	Provides the time of the event, which is encoded into the event.

10.42.3.15 fill_seq_number()

```
void seq64::midi_container::fill_seq_number (
    int seq ) [private]
```

Writes `0xFF 0x00 0x02 ss ss`, where `ss ss` is the variable-length value for the sequence number. This function is used in the new [midifile::write_song\(\)](#) function, which should be ready to go by the time you're reading this. Compare this function to the beginning of [midi_container::fill\(\)](#).

Warning

This is an optional event, which must occur only at the start of a track, before any non-zero delta-time. For Format 2 MIDI files, this is used to identify each track. If omitted, the sequences are numbered sequentially in the order the tracks appear. For Format 1 files, this event should occur on the first track only. So, are we writing a hybrid format?

Parameters

<i>seq</i>	The sequence/track number to write.
------------	-------------------------------------

10.42.3.16 fill_seq_name()

```
void seq64::midi_container::fill_seq_name (
    const std::string & name ) [private]
```

Writes 0xFF 0x03, and then the track name. This function is used in the new [midifile::write_song\(\)](#) function, which should be ready to go by the time you're reading this.

Compare this function to the beginning of [midi_container::fill\(\)](#).

Parameters

<i>name</i>	The sequence/track name to set. We could get this item from <code>m_sequence</code> , but the parameter allows the flexibility to change the name.
-------------	--

10.42.3.17 fill_meta_track_end()

```
void seq64::midi_container::fill_meta_track_end (
    midipulse deltetime ) [private]
```

10.42.3.18 fill_proprietary()

```
void seq64::midi_container::fill_proprietary ( ) [private]
```

Then, if we're not using the legacy output format, we add the "events" for the musical key, musical scale, and the background sequence for the current sequence. Finally, if transpose support has been compiled into the program, we add that information as well. New feature: save more sequence-specific values, if not legacy format and not saved globally. We use a single byte for the key and scale, and a long for the background sequence. We save these values only if they are different from the defaults; in most cases they will have been left alone by the user. We save per-sequence values here only if the global-background-sequence feature is not in force.

For the new "transposable" flag (tagged by the value `c_transpose`) we really only care about saving the value of "false", because otherwise we can assume the value is true for the given sequence, and save space by not saving it... generally only drum patterns will not be transposable.

However, for now, write it anyway for consistency with Seq32.

10.42.3.19 song_fill_seq_event()

```
midipulse seq64::midi_container::song_fill_seq_event (
    const trigger & trig,
    midipulse prev_timestamp ) [private]
```

Parameters

<i>trig</i>	The current trigger to be processed.
<i>prev_timestamp</i>	The time-stamp of the previous event.

Returns

The next time-stamp value is returned.

10.42.3.20 song_fill_seq_trigger()

```
void seq64::midi_container::song_fill_seq_trigger (
    const trigger & trig,
    midipulse length,
    midipulse prev_timestamp ) [private]
```

For a song-performance, there will be only one trigger, covering the beginning to the end of the fully unlooped track.

Parameters

<i>trig</i>	The current trigger to be processed.
<i>length</i>	Provides the total length of the sequence.
<i>prev_timestamp</i>	The time-stamp of the previous event, which is actually the first event.

10.42.4 Friends And Related Function Documentation

10.42.4.1 midifile

```
friend class midifile [friend]
```

10.42.5 Field Documentation

10.42.5.1 m_sequence

```
sequence& seq64::midi_container::m_sequence [private]
```

10.42.5.2 m_position_for_get

```
unsigned seq64::midi_container::m_position_for_get [mutable], [private]
```

10.43 seq64::midi_control Class Reference

This class (formerly a struct) contains the control information for sequences that make up a live set.

Public Types

- enum [action](#) {
[action_toggle](#),
[action_on](#),
[action_off](#) }

Public Member Functions

- [midi_control](#) ()
This default constructor creates a "zero" object.
- bool [active](#) () const
- bool [inverse_active](#) () const
- int [status](#) () const
- int [data](#) () const
- int [min_value](#) () const
- int [max_value](#) () const
- void [set](#) (int values[6])
Not so sure if this really saves trouble for the caller.
- void [set](#) (midibyte values[6])
Not so sure if this really saves trouble for the caller.
- bool [match](#) (midibyte status, midibyte data) const
Handles a common check in the perform module.
- bool [in_range](#) (midibyte data) const
Handles a common check in the perform module.

Private Attributes

- bool [m_active](#)
Provides the value for active.
- bool [m_inverse_active](#)
Provides the value for inverse-active.
- int [m_status](#)
Provides the value for the status.
- int [m_data](#)
Provides the value for the data.
- int [m_min_value](#)
Provides the minimum value for the controller.
- int [m_max_value](#)
Provides the value for the controller.

10.43.1 Detailed Description

Note that, although we've converted this to a full-fledged class, the ordering of variables and the data arrays used to fill them is very significant. See the midifile and optionsfile modules.

The perform module sets up the three following arrays for each of the MIDI controls that can be defined in the "rc" file:

```
m_midi_cc_toggle[]
m_midi_cc_on[]
m_midi_cc_off[]
```

These three arrays are specified in the "rc" by a line like the following:

```
n [0 0 0 0 0 0] [0 0 0 0 0 0] [0 0 0 0 0 0]
```

where n ranges from 0 to 73 or 83. Lines 0 to 31 provide controller values for the "pattern group", one line for each of the 32 pattern slots. Lines 32 to 63 provide controller values for the "mute in group", one line for each of the 32 pattern slots. The rest of the lines provide entries for control of: BPM up, BPM down, Screen-set up, Screen-set down, Mod Replaces, Mod Snapshot, Mod Queue, Mod gmute (group mute), Mod glearn (group learn), and Screen-set Play. Additional controls are currently in the works.

In each of the bracketed sections, the values correspond to the members in this order: `m_active`, `m_inverse_active`, `m_status`, `m_data`, `m_min_value`, and `m_max_value`.

Why are the status, data, and min/max values long? A character or midibyte would be enough. We'll fix that later, once we have tested this stuff. We do need to convert them from long to int, though, and do that in the scanning and output done by optionsfile.

10.43.2 Member Enumeration Documentation

10.43.2.1 action

```
enum seq64::midi_control::action
```

Enumerator

action_toggle	Provides the kind of MIDI control event found, used in the new perform::handle_midi_control_ex() function. Toggles the status of the given control. For the "playback" status, indicates the "pause" functionality.
action_on	Turns on the status of the given control. For the "playback" status, indicates the "start" functionality.
action_off	Turns off the status of the given control. For the "playback" status, indicates the "stop" functionality.

10.43.3 Constructor & Destructor Documentation

10.43.3.1 midi_control()

```
seq64::midi_control::midi_control ( ) [inline]
```

Every member is either false or zero.

10.43.4 Member Function Documentation

10.43.4.1 active()

```
bool seq64::midi_control::active ( ) const [inline]
```

10.43.4.2 inverse_active()

```
bool seq64::midi_control::inverse_active ( ) const [inline]
```

10.43.4.3 status()

```
int seq64::midi_control::status ( ) const [inline]
```

10.43.4.4 data()

```
int seq64::midi_control::data ( ) const [inline]
```

10.43.4.5 min_value()

```
int seq64::midi_control::min_value ( ) const [inline]
```

10.43.4.6 max_value()

```
int seq64::midi_control::max_value ( ) const [inline]
```

10.43.4.7 set() [1/2]

```
void seq64::midi_control::set (
    int values[6] ) [inline]
```

It fits in with the big-ass sscanf() call in optionsfile.

Parameters

<i>values</i>	Provides the six values, in an integer array, to set into the members in this order: m_active, m_inverse_active, m_status, m_data, m_min_value, and m_max_value.
---------------	--

10.43.4.8 set() [2/2]

```
void seq64::midi_control::set (
    midibyte values[6] ) [inline]
```

It fits in with the usage in midifile.

Parameters

<i>values</i>	Provides the six values, in a byte array, to set into the members in this order: m_active, m_inverse_active, m_status, m_data, m_min_value, and m_max_value.
---------------	--

10.43.4.9 match()

```
bool seq64::midi_control::match (
    midibyte status,
    midibyte data ) const [inline]
```

Parameters

<i>status</i>	Provides the status byte, which is checked against m_status.
<i>data</i>	Provides the data byte, which is checked against m_data.

10.43.4.10 in_range()

```
bool seq64::midi_control::in_range (
    midibyte data ) const [inline]
```

10.43.5 Field Documentation

10.43.5.1 m_active

```
bool seq64::midi_control::m_active [private]
```

10.43.5.2 m_inverse_active

```
bool seq64::midi_control::m_inverse_active [private]
```

10.43.5.3 m_status

```
int seq64::midi_control::m_status [private]
```

Big question is, is the channel included here? Yes. So the next question is, is it ignored? We don't think so.

10.43.5.4 m_data

```
int seq64::midi_control::m_data [private]
```


10.43.5.5 m_min_value

```
int seq64::midi_control::m_min_value [private]
```

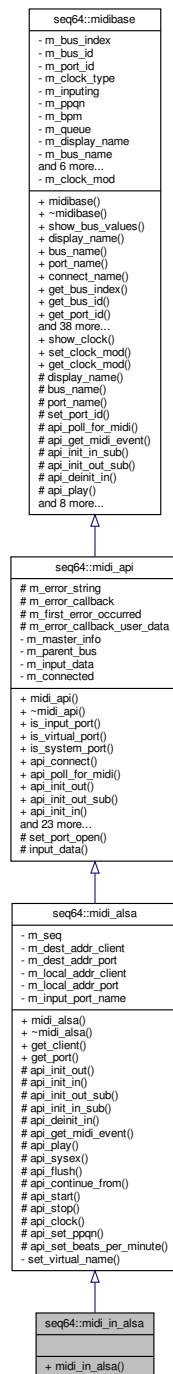
10.43.5.6 m_max_value

```
int seq64::midi_control::m_max_value [private]
```

10.44 seq64::midi_in_alsa Class Reference

This class implements the ALSA version of a MIDI input object.

Inheritance diagram for seq64::midi_in_alsa:



Public Member Functions

- [midi_in_alsa](#) ([midibus](#) & [parentbus](#), [midi_info](#) & [masterinfo](#))

Additional Inherited Members

10.44.1 Constructor & Destructor Documentation

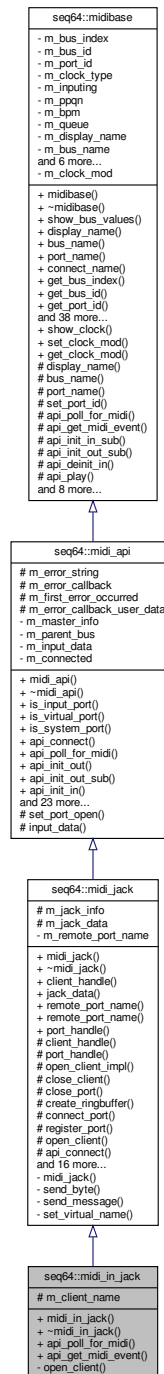
10.44.1.1 midi_in_alsa()

```
seq64::midi_in_alsa::midi_in_alsa (
    midibus & parentbus,
    midi_info & masterinfo )
```

10.45 seq64::midi_in_jack Class Reference

The class for handling JACK MIDI input.

Inheritance diagram for seq64::midi_in_jack:



Public Member Functions

- `midi_in_jack` (`midibus` &parentbus, `midi_info` &masterinfo)
- virtual `~midi_in_jack` ()
- virtual int `api_poll_for_midi` ()
- virtual bool `api_get_midi_event` (event *)

Protected Attributes

- std::string [m_client_name](#)

Private Member Functions

- virtual bool [open_client](#) ()

This function is virtual, so we don't call it in the constructor, using [open_client_impl\(\)](#) directly instead.

Additional Inherited Members

10.45.1 Constructor & Destructor Documentation

10.45.1.1 midi_in_jack()

```
seq64::midi_in_jack::midi_in_jack (
    midibus & parentbus,
    midi\_info & masterinfo )
```

10.45.1.2 ~midi_in_jack()

```
virtual seq64::midi_in_jack::~~midi_in_jack ( ) [virtual]
```

10.45.2 Member Function Documentation

10.45.2.1 api_poll_for_midi()

```
virtual int seq64::midi_in_jack::api_poll_for_midi ( ) [virtual]
```

Reimplemented from [seq64::midi_api](#).

10.45.2.2 api_get_midi_event()

```
virtual bool seq64::midi_in_jack::api_get_midi_event (
    event * ) [virtual]
```

Reimplemented from [seq64::midi_jack](#).

10.45.2.3 open_client()

```
virtual bool seq64::midi_in_jack::open_client ( ) [inline], [private], [virtual]
```

This function replaces the RtMidi function "connect()".

Implements [seq64::midi_jack](#).

10.45.3 Field Documentation

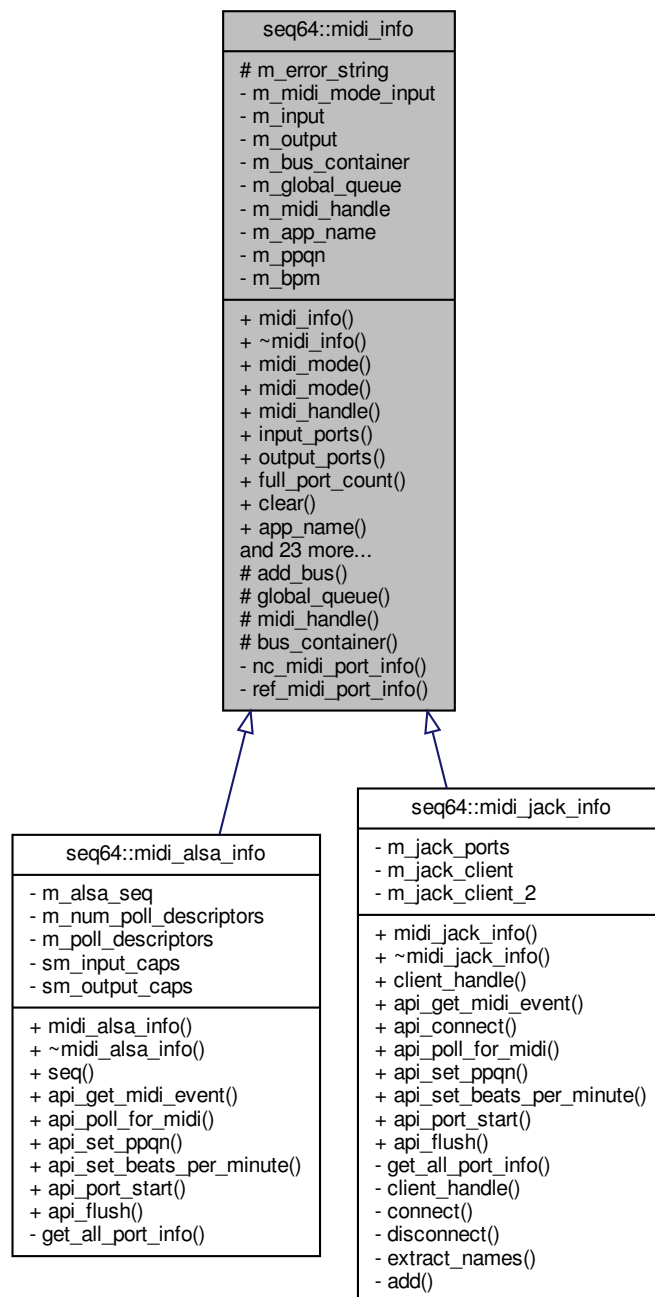
10.45.3.1 m_client_name

```
std::string seq64::midi_in_jack::m_client_name [protected]
```

10.46 seq64::midi_info Class Reference

The class for holding basic information on the MIDI input and output ports currently present in the system.

Inheritance diagram for seq64::midi_info:



Public Member Functions

- `midi_info` (const std::string &appname, int ppqn=SEQ64_DEFAULT_PPQN, midibpm bpm=SEQ64_DEFAULT_BPM)
- virtual `~midi_info` ()
- bool `midi_mode` () const
'Getter' function for member m_midi_mode_input

- void [midi_mode](#) (bool flag)
'Setter' function for member m_midi_mode_input
- void * [midi_handle](#) ()
'Getter' function for member m_midi_handle
- [midi_port_info](#) & [input_ports](#) ()
'Getter' function for member m_input
- [midi_port_info](#) & [output_ports](#) ()
'Getter' function for member m_output
- int [full_port_count](#) () const
'Getter' function for member Total port count.
- void [clear](#) ()
- const std::string & [app_name](#) () const
'Getter' function for member m_app_name
- int [ppqn](#) () const
'Getter' function for member m_ppqn, simple version, also see [api_set_ppqn\(\)](#).
- [midibpm bpm](#) () const
'Getter' function for member m_bpm, simple version, also see [api_set_beats_per_minute\(\)](#).
- virtual void [api_set_ppqn](#) (int p)
Special setter.
- virtual void [api_set_beats_per_minute](#) ([midibpm](#) b)
Special setter.
- virtual void [api_port_start](#) ([mastermidibus](#) &, int, int)
An ALSA-specific function at the moment.
- virtual bool [api_get_midi_event](#) ([event](#) *inev)=0
- virtual int [api_poll_for_midi](#) ()=0
- virtual void [api_flush](#) ()=0
- virtual bool [api_connect](#) ()
Used only in the [midi_jack_info](#) class.
- virtual int [get_port_count](#) () const
- virtual int [get_bus_id](#) (int index) const
- virtual std::string [get_bus_name](#) (int index) const
- virtual int [get_port_id](#) (int index) const
- virtual std::string [get_port_name](#) (int index) const
- virtual bool [get_input](#) (int index) const
- virtual bool [get_virtual](#) (int index) const
- virtual bool [get_system](#) (int index) const
- virtual int [queue_number](#) (int index) const
- std::string [connect_name](#) (int index) const
- std::string [port_list](#) () const
- int [global_queue](#) () const
- void [error](#) ([rerror::Type](#) type, const std::string &errorstring)
A basic error reporting function for [midi_info](#) classes.
- virtual int [get_all_port_info](#) ()=0

Protected Member Functions

- void [add_bus](#) (const [midibus](#) *m)
Adds the midibus to a quick list of all ports for use in the [api_connect\(\)](#) call in mastermidibus.
- void [global_queue](#) (int q)
'Setter' function for member m_global_queue
- void [midi_handle](#) (void *h)
'Setter' function for member m_midi_handle
- std::vector< [midibus](#) * > & [bus_container](#) ()
'Getter' function for member m_bus_container

Protected Attributes

- `std::string m_error_string`
Error string for the `midi_info` interface.

Private Member Functions

- `const midi_port_info & nc_midi_port_info () const`
'Getter' function for member `m_input` or `m_output` Used for retrieving values from the input or output containers.
- `midi_port_info & ref_midi_port_info ()`
'Getter' function for member `m_input` or `m_output`

Private Attributes

- `bool m_midi_mode_input`
Indicates which mode we're in, input or output.
- `midi_port_info m_input`
Holds data on the ALSA/JACK/Core/WinMM inputs.
- `midi_port_info m_output`
Holds data on the ALSA/JACK/Core/WinMM outputs.
- `std::vector< midibus * > m_bus_container`
Holds pointers to the ports that were created, so that, after activation, we can call the `connect_port()` function on those that are not virtual.
- `int m_global_queue`
The ID of the ALSA MIDI queue.
- `void * m_midi_handle`
Provides a handle to the main ALSA or JACK implementation object.
- `const std::string m_app_name`
Holds this value for passing along, to reduce the number of arguments needed.
- `int m_ppqn`
Hold this value for passing along to some ports that get created.
- `midibpm m_bpm`
Hold this value for passing along to some ports that get created.

Friends

- class `rtmidi_info`

10.46.1 Constructor & Destructor Documentation

10.46.1.1 midi_info()

```
seq64::midi_info::midi_info (
    const std::string & appname,
    int ppqn = SEQ64_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

10.46.1.2 ~midi_info()

```
virtual seq64::midi_info::~~midi_info ( ) [inline], [virtual]
```

10.46.2 Member Function Documentation

10.46.2.1 midi_mode() [1/2]

```
bool seq64::midi_info::midi_mode ( ) const [inline]
```

10.46.2.2 midi_mode() [2/2]

```
void seq64::midi_info::midi_mode (
    bool flag ) [inline]
```

10.46.2.3 midi_handle() [1/2]

```
void* seq64::midi_info::midi_handle ( ) [inline]
```

10.46.2.4 input_ports()

```
midi\_port\_info& seq64::midi_info::input_ports ( ) [inline]
```

10.46.2.5 output_ports()

```
midi\_port\_info& seq64::midi_info::output_ports ( ) [inline]
```

10.46.2.6 full_port_count()

```
int seq64::midi_info::full_port_count ( ) const [inline]
```

10.46.2.7 clear()

```
void seq64::midi_info::clear ( ) [inline]
```

10.46.2.8 app_name()

```
const std::string& seq64::midi_info::app_name ( ) const [inline]
```

10.46.2.9 ppqn()

```
int seq64::midi_info::ppqn ( ) const [inline]
```

10.46.2.10 bpm()

```
midibpm seq64::midi_info::bpm ( ) const [inline]
```

10.46.2.11 api_set_ppqn()

```
virtual void seq64::midi_info::api_set_ppqn (
    int p ) [inline], [virtual]
```

Reimplemented in [seq64::midi_alsa_info](#), and [seq64::midi_jack_info](#).

10.46.2.12 api_set_beats_per_minute()

```
virtual void seq64::midi_info::api_set_beats_per_minute (
    midibpm b ) [inline], [virtual]
```

Reimplemented in [seq64::midi_alsa_info](#), and [seq64::midi_jack_info](#).

10.46.2.13 api_port_start()

```
virtual void seq64::midi_info::api_port_start (
    mastermidibus & ,
    int ,
    int ) [inline], [virtual]
```

Reimplemented in [seq64::midi_alsa_info](#), and [seq64::midi_jack_info](#).

10.46.2.14 `api_get_midi_event()`

```
virtual bool seq64::midi_info::api_get_midi_event (
    event * inev ) [pure virtual]
```

Implemented in [seq64::midi_alsa_info](#), and [seq64::midi_jack_info](#).

10.46.2.15 `api_poll_for_midi()`

```
virtual int seq64::midi_info::api_poll_for_midi ( ) [pure virtual]
```

Implemented in [seq64::midi_alsa_info](#), and [seq64::midi_jack_info](#).

10.46.2.16 `api_flush()`

```
virtual void seq64::midi_info::api_flush ( ) [pure virtual]
```

Implemented in [seq64::midi_alsa_info](#), and [seq64::midi_jack_info](#).

10.46.2.17 `api_connect()`

```
virtual bool seq64::midi_info::api_connect ( ) [inline], [virtual]
```

Reimplemented in [seq64::midi_jack_info](#).

10.46.2.18 `get_port_count()`

```
virtual int seq64::midi_info::get_port_count ( ) const [inline], [virtual]
```

10.46.2.19 `get_bus_id()`

```
virtual int seq64::midi_info::get_bus_id (
    int index ) const [inline], [virtual]
```

10.46.2.20 get_bus_name()

```
virtual std::string seq64::midi_info::get_bus_name (
    int index ) const [inline], [virtual]
```

10.46.2.21 get_port_id()

```
virtual int seq64::midi_info::get_port_id (
    int index ) const [inline], [virtual]
```

10.46.2.22 get_port_name()

```
virtual std::string seq64::midi_info::get_port_name (
    int index ) const [inline], [virtual]
```

10.46.2.23 get_input()

```
virtual bool seq64::midi_info::get_input (
    int index ) const [inline], [virtual]
```

10.46.2.24 get_virtual()

```
virtual bool seq64::midi_info::get_virtual (
    int index ) const [inline], [virtual]
```

10.46.2.25 get_system()

```
virtual bool seq64::midi_info::get_system (
    int index ) const [inline], [virtual]
```

10.46.2.26 queue_number()

```
virtual int seq64::midi_info::queue_number (
    int index ) const [inline], [virtual]
```

10.46.2.27 connect_name()

```
std::string seq64::midi_info::connect_name (
    int index ) const [inline]
```

10.46.2.28 port_list()

```
std::string seq64::midi_info::port_list ( ) const
```

10.46.2.29 global_queue() [1/2]

```
int seq64::midi_info::global_queue ( ) const [inline]
```

10.46.2.30 error()

```
void seq64::midi_info::error (
    rterror::Type type,
    const std::string & errorstring )
```

10.46.2.31 get_all_port_info()

```
virtual int seq64::midi_info::get_all_port_info ( ) [pure virtual]
```

Implemented in [seq64::midi_alsa_info](#), and [seq64::midi_jack_info](#).

10.46.2.32 add_bus()

```
void seq64::midi_info::add_bus (
    const midibus * m ) [inline], [protected]
```

We could add the midibus pointer to the [midi_port_info](#) structure, but that information is strictly for representing data obtained by querying the system via the selected API.

10.46.2.33 global_queue() [2/2]

```
void seq64::midi_info::global_queue (
    int q ) [inline], [protected]
```

10.46.2.34 midi_handle() [2/2]

```
void seq64::midi_info::midi_handle (
    void * h )    [inline], [protected]
```

10.46.2.35 bus_container()

```
std::vector<midibus *>& seq64::midi_info::bus_container ( )    [inline], [protected]
```

10.46.2.36 nc_midi_port_info()

```
const midi_port_info& seq64::midi_info::nc_midi_port_info ( ) const    [inline], [private]
```

The caller must insure the proper container by calling the [midi_mode\(\)](#) function with the value of true (SEQ64_MIDI_INPUT_PORT) or false (SEQ64_MIDI_OUTPUT_PORT) first. Ugly stuff. I hate it.

10.46.2.37 ref_midi_port_info()

```
midi_port_info& seq64::midi_info::ref_midi_port_info ( )    [inline], [private]
```

10.46.3 Friends And Related Function Documentation

10.46.3.1 rtmidi_info

```
friend class rtmidi_info    [friend]
```

10.46.4 Field Documentation

10.46.4.1 m_midi_mode_input

```
bool seq64::midi_info::m_midi_mode_input    [private]
```

We have to pick the mode we need to be in with the [set_mode\(\)](#) function before we do a series of operations. This clumsy two-step is needed in order to preserve the [midi_api](#) interface.

10.46.4.2 m_input

```
midi_port_info seq64::midi_info::m_input [private]
```

10.46.4.3 m_output

```
midi_port_info seq64::midi_info::m_output [private]
```

10.46.4.4 m_bus_container

```
std::vector<midibus *> seq64::midi_info::m_bus_container [private]
```

See the [add_bus\(\)](#) and [bus_container\(\)](#) member functions.

10.46.4.5 m_global_queue

```
int seq64::midi_info::m_global_queue [private]
```

10.46.4.6 m_midi_handle

```
void* seq64::midi_info::m_midi_handle [private]
```

Created by the class derived from [midi_info](#).

10.46.4.7 m_app_name

```
const std::string seq64::midi_info::m_app_name [private]
```

This value is the main application name as determined at ./configure time.

10.46.4.8 m_ppqn

```
int seq64::midi_info::m_ppqn [private]
```

Some APIs can use this value.

10.46.4.9 m_bpm

```
midibpm seq64::midi_info::m_bpm [private]
```

Some APIs can use this value.

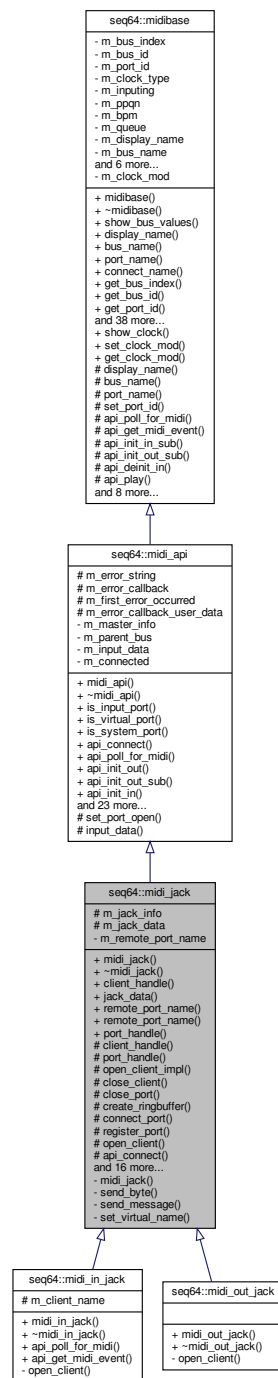
10.46.4.10 m_error_string

```
std::string seq64::midi_info::m_error_string [protected]
```

10.47 seq64::midi_jack Class Reference

This class implements with JACK version of the [midi_alsa](#) object.

Inheritance diagram for seq64::midi_jack:



Public Member Functions

- [midi_jack](#) ([midibus](#) &[parentbus](#), [midi_info](#) &[masterinfo](#))
- virtual [~midi_jack](#) ()
- [jack_client_t](#) * [client_handle](#) ()
'Getter' function for member m_jack_client This is the platform-specific version of midi_handle().
- [midi_jack_data](#) & [jack_data](#) ()
'Getter' function for member m_jack_data
- const std::string & [remote_port_name](#) () const
'Getter' function for member m_remote_port_name
- void [remote_port_name](#) (const std::string &s)
'Setter' function for member m_remote_port_name
- [jack_port_t](#) * [port_handle](#) ()
'Getter' function for member m_jack_port This is the platform-specific version of midi_handle().

Protected Member Functions

- void [client_handle](#) ([jack_client_t](#) *handle)
'Setter' function for member m_jack_data.m_jack_client
- void [port_handle](#) ([jack_port_t](#) *handle)
'Setter' function for member m_jack_data.m_jack_port
- bool [open_client_impl](#) (bool input)
- void [close_client](#) ()
- void [close_port](#) ()
- bool [create_ringbuffer](#) (size_t rbsize)
- bool [connect_port](#) (bool input, const std::string &sourceportname, const std::string &destportname)
- bool [register_port](#) (bool input, const std::string &portname)
- virtual bool [open_client](#) ()=0
- virtual bool [api_connect](#) ()
- virtual bool [api_init_out](#) ()
- virtual bool [api_init_in](#) ()
- virtual bool [api_init_out_sub](#) ()
- virtual bool [api_init_in_sub](#) ()
- virtual bool [api_deinit_in](#) ()
- virtual bool [api_get_midi_event](#) ([event](#) *)
- virtual void [api_play](#) ([event](#) *e24, [midibyte](#) channel)
- virtual void [api_sysex](#) ([event](#) *e24)
- virtual void [api_flush](#) ()
- virtual void [api_continue_from](#) ([midipulse](#) tick, [midipulse](#) beats)
- virtual void [api_start](#) ()
- virtual void [api_stop](#) ()
- virtual void [api_clock](#) ([midipulse](#) tick)
- virtual void [api_set_ppqn](#) (int ppqn)
- virtual void [api_set_beats_per_minute](#) ([midibpm](#) bpm)
- virtual std::string [api_get_port_name](#) ()

Protected Attributes

- [midi_jack_info](#) & [m_jack_info](#)
This reference is needed in order for this [midi_jack](#) object to add itself to the main [midi_jack_info](#) list when running in single-JACK client mode.
- [midi_jack_data](#) [m_jack_data](#)
Holds the data needed for JACK processing.

Private Member Functions

- [midi_jack](#) ()
- void [send_byte](#) ([midibyte](#) evbyte)
- bool [send_message](#) (const [midi_message](#) &message)
- bool [set_virtual_name](#) (int portid, const std::string &portname)

Private Attributes

- std::string [m_remote_port_name](#)

Preserves the original name of the remote port, so it can be used later for connection.

Friends

- class [midi_jack_info](#)

Additional Inherited Members

10.47.1 Constructor & Destructor Documentation

10.47.1.1 [midi_jack\(\)](#) [1/2]

```
seq64::midi_jack::midi_jack ( ) [private]
```

10.47.1.2 [midi_jack\(\)](#) [2/2]

```
seq64::midi_jack::midi_jack (
    midibus & parentbus,
    midi\_info & masterinfo )
```

10.47.1.3 [~midi_jack\(\)](#)

```
virtual seq64::midi_jack::~~midi_jack ( ) [virtual]
```

10.47.2 Member Function Documentation

10.47.2.1 client_handle() [1/2]

```
jack_client_t* seq64::midi_jack::client_handle ( ) [inline]
```

10.47.2.2 jack_data()

```
midi_jack_data& seq64::midi_jack::jack_data ( ) [inline]
```

10.47.2.3 remote_port_name() [1/2]

```
const std::string& seq64::midi_jack::remote_port_name ( ) const [inline]
```

10.47.2.4 remote_port_name() [2/2]

```
void seq64::midi_jack::remote_port_name (
    const std::string & s ) [inline]
```

10.47.2.5 port_handle() [1/2]

```
jack_port_t* seq64::midi_jack::port_handle ( ) [inline]
```

10.47.2.6 client_handle() [2/2]

```
void seq64::midi_jack::client_handle (
    jack_client_t * handle ) [inline], [protected]
```

10.47.2.7 port_handle() [2/2]

```
void seq64::midi_jack::port_handle (
    jack_port_t * handle ) [inline], [protected]
```

10.47.2.8 open_client_impl()

```
bool seq64::midi_jack::open_client_impl (
    bool input ) [protected]
```

10.47.2.9 close_client()

```
void seq64::midi_jack::close_client ( ) [protected]
```

10.47.2.10 close_port()

```
void seq64::midi_jack::close_port ( ) [protected]
```

10.47.2.11 create_ringbuffer()

```
bool seq64::midi_jack::create_ringbuffer (
    size_t rbsize ) [protected]
```

10.47.2.12 connect_port()

```
bool seq64::midi_jack::connect_port (
    bool input,
    const std::string & sourceportname,
    const std::string & destportname ) [protected]
```

10.47.2.13 register_port()

```
bool seq64::midi_jack::register_port (
    bool input,
    const std::string & portname ) [protected]
```

10.47.2.14 open_client()

```
virtual bool seq64::midi_jack::open_client ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi_out_jack](#), and [seq64::midi_in_jack](#).

10.47.2.15 `api_connect()`

```
virtual bool seq64::midi_jack::api_connect ( ) [protected], [virtual]
```

Reimplemented from [seq64::midi_api](#).

10.47.2.16 `api_init_out()`

```
virtual bool seq64::midi_jack::api_init_out ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.17 `api_init_in()`

```
virtual bool seq64::midi_jack::api_init_in ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.18 `api_init_out_sub()`

```
virtual bool seq64::midi_jack::api_init_out_sub ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.19 `api_init_in_sub()`

```
virtual bool seq64::midi_jack::api_init_in_sub ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.20 `api_deinit_in()`

```
virtual bool seq64::midi_jack::api_deinit_in ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.21 api_get_midi_event()

```
virtual bool seq64::midi_jack::api_get_midi_event (
    event * ) [inline], [protected], [virtual]
```

Returns

Returns false, since this is an input function that is implemented fully only by [midi_in_jack](#).

Implements [seq64::midi_api](#).

Reimplemented in [seq64::midi_in_jack](#).

10.47.2.22 api_play()

```
virtual void seq64::midi_jack::api_play (
    event * e24,
    midibyte channel ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.23 api_sysex()

```
virtual void seq64::midi_jack::api_sysex (
    event * e24 ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.24 api_flush()

```
virtual void seq64::midi_jack::api_flush ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.25 api_continue_from()

```
virtual void seq64::midi_jack::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.26 `api_start()`

```
virtual void seq64::midi_jack::api_start ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.27 `api_stop()`

```
virtual void seq64::midi_jack::api_stop ( ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.28 `api_clock()`

```
virtual void seq64::midi_jack::api_clock (
    midipulse tick ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.29 `api_set_ppqn()`

```
virtual void seq64::midi_jack::api_set_ppqn (
    int ppqn ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.30 `api_set_beats_per_minute()`

```
virtual void seq64::midi_jack::api_set_beats_per_minute (
    midibpm bpm ) [protected], [virtual]
```

Implements [seq64::midi_api](#).

10.47.2.31 `api_get_port_name()`

```
virtual std::string seq64::midi_jack::api_get_port_name ( ) [protected], [virtual]
```

Reimplemented from [seq64::midi_api](#).

10.47.2.32 send_byte()

```
void seq64::midi_jack::send_byte (
    midi_byte evbyte ) [private]
```

10.47.2.33 send_message()

```
bool seq64::midi_jack::send_message (
    const midi_message & message ) [private]
```

10.47.2.34 set_virtual_name()

```
bool seq64::midi_jack::set_virtual_name (
    int portid,
    const std::string & portname ) [private]
```

10.47.3 Friends And Related Function Documentation

10.47.3.1 midi_jack_info

```
friend class midi_jack_info [friend]
```

10.47.4 Field Documentation

10.47.4.1 m_remote_port_name

```
std::string seq64::midi_jack::m_remote_port_name [private]
```

10.47.4.2 m_jack_info

```
midi_jack_info& seq64::midi_jack::m_jack_info [protected]
```

10.47.4.3 m_jack_data

```
midi_jack_data seq64::midi_jack::m_jack_data [protected]
```

Please do not confuse this item with the m_midi_handle of the [midi_api](#) base class. This object holds a JACK-client pointer and a JACK-port pointer.

10.48 seq64::midi_jack_data Struct Reference

Contains the JACK MIDI API data as a kind of scratchpad for this object.

Public Member Functions

- [midi_jack_data](#) ()
Constructor [midi_jack_data](#)
- [~midi_jack_data](#) ()
This destructor currently does nothing.
- bool [valid_buffer](#) () const
Tests that the buffer is good.

Data Fields

- jack_client_t * [m_jack_client](#)
Holds the JACK sequencer client pointer so that it can be used by the midibus objects.
- jack_port_t * [m_jack_port](#)
Holds the JACK port information of the JACK client.
- jack_ringbuffer_t * [m_jack_buffsize](#)
Holds the size of data for communicating between the client ring-buffer and the JACK port's internal buffer.
- jack_ringbuffer_t * [m_jack_buffmessage](#)
Holds the data for communicating between the client ring-buffer and the JACK port's internal buffer.
- jack_time_t [m_jack_lasttime](#)
The last time-stamp obtained.
- [rtmidi_in_data](#) * [m_jack_rtmidiin](#)
Holds special data peculiar to the client and its MIDI input processing.

10.48.1 Detailed Description

This guy needs a constructor taking parameters for an [rtmidi_in_data](#) pointer.

10.48.2 Constructor & Destructor Documentation

10.48.2.1 midi_jack_data()

```
seq64::midi_jack_data::midi_jack_data ( ) [inline]
```

10.48.2.2 ~midi_jack_data()

```
seq64::midi_jack_data::~~midi_jack_data ( ) [inline]
```

We rely on the enclosing class to close out the things that it created.

10.48.3 Member Function Documentation

10.48.3.1 valid_buffer()

```
bool seq64::midi_jack_data::valid_buffer ( ) const [inline]
```

10.48.4 Field Documentation

10.48.4.1 m_jack_client

```
jack_client_t* seq64::midi_jack_data::m_jack_client
```

This is actually an opaque pointer; there is no way to get the actual fields in this structure; they can only be accessed through functions in the JACK API. Note that it is also stored as a void pointer in [midi_info::m_midi_handle](#). This item can either be the single JACK client created by the [midi_jack_info](#) object, or a JACK client created by the [midi_jack](#) object in the "multi-client" mode (which is not yet complete or usable).

10.48.4.2 m_jack_port

```
jack_port_t* seq64::midi_jack_data::m_jack_port
```

10.48.4.3 m_jack_buffsize

```
jack_ringbuffer_t* seq64::midi_jack_data::m_jack_buffsize
```

10.48.4.4 m_jack_buffmessage

```
jack_ringbuffer_t* seq64::midi_jack_data::m_jack_buffmessage
```

10.48.4.5 m_jack_lasttime

```
jack_time_t seq64::midi_jack_data::m_jack_lasttime
```

Use for calculating the delta time, I would imagine.

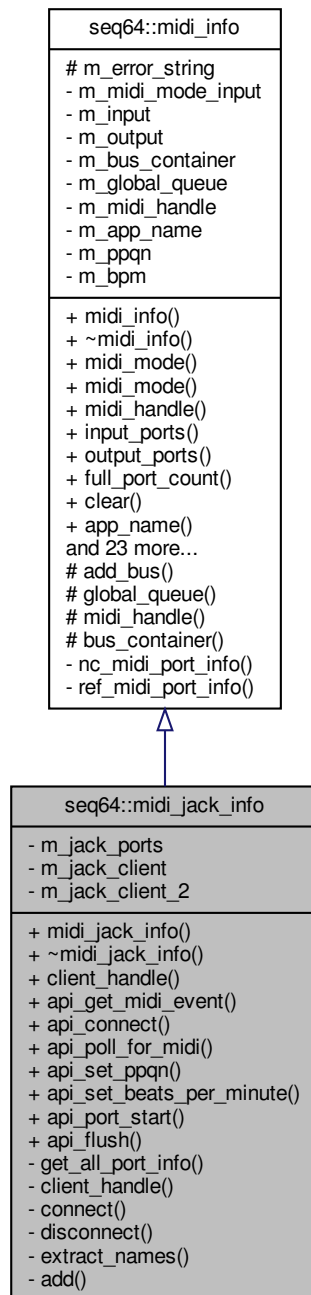
10.48.4.6 m_jack_rtmidiin

```
rtmidi_in_data* seq64::midi_jack_data::m_jack_rtmidiin
```

10.49 seq64::midi_jack_info Class Reference

The class for handling JACK MIDI port enumeration.

Inheritance diagram for seq64::midi_jack_info:



Public Member Functions

- `midi_jack_info` (const std::string &appname, int ppqn=SEQ64_DEFAULT_PPQN, midibpm bpm=SEQ64_DEFAULT_BPM)
- virtual `~midi_jack_info` ()
- `jack_client_t * client_handle` ()

'Getter' function for member `m_jack_client` This is the platform-specific version of `midi_handle()`.

- virtual bool `api_get_midi_event` (`event *inev`)
- virtual bool `api_connect` ()
- virtual int `api_poll_for_midi` ()
- virtual void `api_set_ppqn` (int p)
- virtual void `api_set_beats_per_minute` (midibpm b)
- virtual void `api_port_start` (mastermidibus &masterbus, int bus, int port)
- virtual void `api_flush` ()

Private Member Functions

- virtual int `get_all_port_info` ()
- void `client_handle` (jack_client_t *j)
'Getter' function for member m_jack_client This is the platform-specific version of `midi_handle()`.
- jack_client_t * `connect` ()
- void `disconnect` ()
- void `extract_names` (const std::string &fullname, std::string &clientname, std::string &portname)
- bool `add` (midi_jack &mj)
Adds a pointer to a JACK port.

Private Attributes

- std::vector< midi_jack * > `m_jack_ports`
Holds the port data.
- jack_client_t * `m_jack_client`
Holds the JACK sequencer client pointer so that it can be used by the midibus objects.
- jack_client_t * `m_jack_client_2`
Holds the JACK input client pointer if multi-client mode is in force.

Friends

- class `midi_jack`
- int `jack_process_io` (jack_nframes_t nframes, void *arg)

Additional Inherited Members

10.49.1 Constructor & Destructor Documentation

10.49.1.1 midi_jack_info()

```
seq64::midi_jack_info::midi_jack_info (
    const std::string & appname,
    int ppqn = SEQ64_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

10.49.1.2 ~midi_jack_info()

```
virtual seq64::midi_jack_info::~~midi_jack_info ( ) [virtual]
```

10.49.2 Member Function Documentation

10.49.2.1 client_handle() [1/2]

```
jack_client_t* seq64::midi_jack_info::client_handle ( ) [inline]
```

10.49.2.2 api_get_midi_event()

```
virtual bool seq64::midi_jack_info::api_get_midi_event (
    event * inev ) [virtual]
```

Implements [seq64::midi_info](#).

10.49.2.3 api_connect()

```
virtual bool seq64::midi_jack_info::api_connect ( ) [virtual]
```

Reimplemented from [seq64::midi_info](#).

10.49.2.4 api_poll_for_midi()

```
virtual int seq64::midi_jack_info::api_poll_for_midi ( ) [virtual]
```

Implements [seq64::midi_info](#).

10.49.2.5 api_set_ppqn()

```
virtual void seq64::midi_jack_info::api_set_ppqn (
    int p ) [virtual]
```

Reimplemented from [seq64::midi_info](#).

10.49.2.6 `api_set_beats_per_minute()`

```
virtual void seq64::midi_jack_info::api_set_beats_per_minute (
    midibpm b ) [virtual]
```

Reimplemented from [seq64::midi_info](#).

10.49.2.7 `api_port_start()`

```
virtual void seq64::midi_jack_info::api_port_start (
    mastermidibus & masterbus,
    int bus,
    int port ) [virtual]
```

Reimplemented from [seq64::midi_info](#).

10.49.2.8 `api_flush()`

```
virtual void seq64::midi_jack_info::api_flush ( ) [virtual]
```

Implements [seq64::midi_info](#).

10.49.2.9 `get_all_port_info()`

```
virtual int seq64::midi_jack_info::get_all_port_info ( ) [private], [virtual]
```

Implements [seq64::midi_info](#).

10.49.2.10 `client_handle()` [2/2]

```
void seq64::midi_jack_info::client_handle (
    jack_client_t * j ) [inline], [private]
```

10.49.2.11 `connect()`

```
jack_client_t* seq64::midi_jack_info::connect ( ) [private]
```


10.49.2.12 disconnect()

```
void seq64::midi_jack_info::disconnect ( ) [private]
```

10.49.2.13 extract_names()

```
void seq64::midi_jack_info::extract_names (
    const std::string & fullname,
    std::string & clientname,
    std::string & portname ) [private]
```

10.49.2.14 add()

```
bool seq64::midi_jack_info::add (
    midi\_jack & mj ) [inline], [private]
```

10.49.3 Friends And Related Function Documentation

10.49.3.1 midi_jack

```
friend class midi\_jack [friend]
```

10.49.3.2 jack_process_io

```
int jack_process_io (
    jack_nframes_t nframes,
    void * arg ) [friend]
```

10.49.4 Field Documentation

10.49.4.1 m_jack_ports

```
std::vector<midi\_jack *> seq64::midi_jack_info::m_jack_ports [private]
```

Not for use with the multi-client option. This list is iterated in the input and output portions of the JACK process callback.

10.49.4.2 m_jack_client

```
jack_client_t* seq64::midi_jack_info::m_jack_client [private]
```

This is actually an opaque pointer; there is no way to get the actual fields in this structure; they can only be accessed through functions in the JACK API. Note that it is also stored as a void pointer in [midi_info::m_midi_handle](#).

In multi-client mode, this pointer is the output client pointer.

10.49.4.3 m_jack_client_2

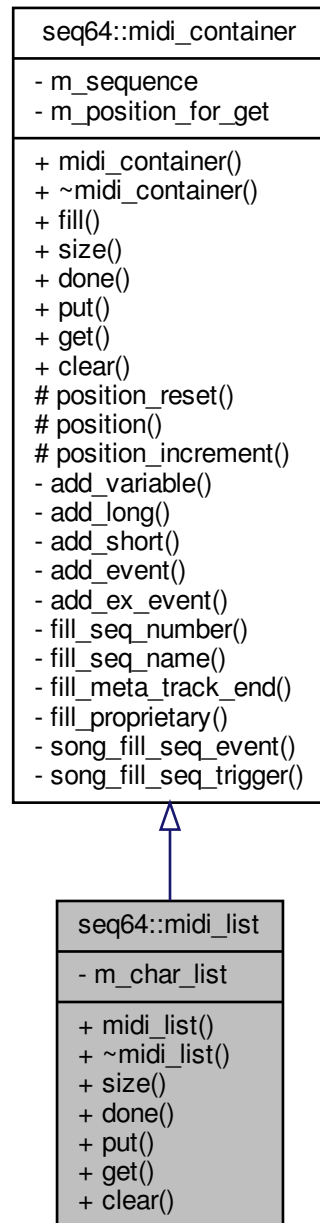
```
jack_client_t* seq64::midi_jack_info::m_jack_client_2 [private]
```

Otherwise, it is an unused null pointer.

10.50 seq64::midi_list Class Reference

This class is the `std::list` implementation of the [midi_container](#).

Inheritance diagram for seq64::midi_list:



Public Member Functions

- `midi_list (sequence &seq)`
This constructor fills in the members.
- `virtual ~midi_list ()`
A rote constructor needed for a base class.
- `virtual std::size_t size () const`

Returns the size of the container, in midibytes.

- virtual bool `done` () const

For popping data from the MIDI list, we are done when the container is empty.

- virtual void `put` (midibyte b)

Provides a way to add a MIDI byte into the list.

- virtual midibyte `get` () const

Provide a way to get the next byte from the container.

- virtual void `clear` ()

Provides a way to clear the container.

Private Types

- typedef std::list< midibyte > CharList

Provides the type of this container.

Private Attributes

- CharList m_char_list

The container itself.

Additional Inherited Members

10.50.1 Member Typedef Documentation

10.50.1.1 CharList

```
typedef std::list<midibyte> seq64::midi_list::CharList [private]
```

This type is basically the same as the `midifile::m_char_list` container in the `midifile` module.

10.50.2 Constructor & Destructor Documentation

10.50.2.1 midi_list()

```
seq64::midi_list::midi_list (
    sequence & seq )
```

Parameters

<code>seq</code>	The sequence/track object that is using this container.
------------------	---

10.50.2.2 ~midi_list()

```
virtual seq64::midi_list::~~midi_list ( ) [inline], [virtual]
```

10.50.3 Member Function Documentation

10.50.3.1 size()

```
virtual std::size_t seq64::midi_list::size ( ) const [inline], [virtual]
```

Reimplemented from [seq64::midi_container](#).

10.50.3.2 done()

```
virtual bool seq64::midi_list::done ( ) const [inline], [virtual]
```

Reimplemented from [seq64::midi_container](#).

10.50.3.3 put()

```
virtual void seq64::midi_list::put (
    midibyte b ) [inline], [virtual]
```

The original seq24 list used an std::list and a push_front operation.

Implements [seq64::midi_container](#).

10.50.3.4 get()

```
virtual midibyte seq64::midi_list::get ( ) const [inline], [virtual]
```

In this implementation, m_position_for_get is not used. The elements of the container are popped off backward! This modifies the character list, so it has to be mutable.

Implements [seq64::midi_container](#).

10.50.3.5 clear()

```
virtual void seq64::midi_list::clear ( ) [inline], [virtual]
```

Implements [seq64::midi_container](#).

10.50.4 Field Documentation

10.50.4.1 m_char_list

```
CharList seq64::midi_list::m_char_list [mutable], [private]
```

It has to be mutable because the const-function [get\(\)](#) actually modifies the container when getting a byte.

10.51 seq64::midi_measures Class Reference

Provides a data structure to hold the numeric equivalent of the measures string "measures:beats:divisions" ("m:b↔:d").

Public Member Functions

- [midi_measures](#) ()
Default constructor for [midi_measures](#).
- [midi_measures](#) (int [measures](#), int [beats](#), int [divisions](#))
Principal constructor for [midi_measures](#).
- int [measures](#) () const
'Getter' function for member m_measures
- void [measures](#) (int m)
'Setter' function for member m_measures
- int [beats](#) () const
'Getter' function for member m_beats
- void [beats](#) (int b)
'Setter' function for member m_beats
- int [divisions](#) () const
'Getter' function for member m_divisions
- void [divisions](#) (int d)
'Setter' function for member m_divisions

Private Attributes

- int [m_measures](#)
The integral number of measures in the measures-based time.
- int [m_beats](#)
The integral number of beats in the measures-based time.
- int [m_divisions](#)
The integral number of divisions/pulses in the measures-based time.

10.51.1 Detailed Description

More commonly known as "bars:beats:ticks", or "BBT".

10.51.2 Constructor & Destructor Documentation

10.51.2.1 midi_measures() [1/2]

```
seq64::midi_measures::midi_measures ( )
```

10.51.2.2 midi_measures() [2/2]

```
seq64::midi_measures::midi_measures (
    int measures,
    int beats,
    int divisions )
```

Parameters

<i>measures</i>	Copied into the m_measures member.
<i>beats</i>	Copied into the m_beats member.
<i>divisions</i>	Copied into the m_divisions member.

10.51.3 Member Function Documentation

10.51.3.1 measures() [1/2]

```
int seq64::midi_measures::measures ( ) const [inline]
```

10.51.3.2 measures() [2/2]

```
void seq64::midi_measures::measures (
    int m ) [inline]
```

Parameters

<i>m</i>	The value to which to set the number of measures. We can add validation later.
----------	--

10.51.3.3 beats() [1/2]

```
int seq64::midi_measures::beats ( ) const [inline]
```

10.51.3.4 beats() [2/2]

```
void seq64::midi_measures::beats (
    int b ) [inline]
```

Parameters

<i>b</i>	The value to which to set the number of beats. We can add validation later.
----------	---

10.51.3.5 divisions() [1/2]

```
int seq64::midi_measures::divisions ( ) const [inline]
```

10.51.3.6 divisions() [2/2]

```
void seq64::midi_measures::divisions (
    int d ) [inline]
```

Parameters

<i>d</i>	The value to which to set the number of divisions. We can add validation later.
----------	---

10.51.4 Field Documentation**10.51.4.1 m_measures**

```
int seq64::midi_measures::m_measures [private]
```


10.51.4.2 m_beats

```
int seq64::midi_measures::m_beats [private]
```

10.51.4.3 m_divisions

```
int seq64::midi_measures::m_divisions [private]
```

There are two possible translations of the two bytes of a division. If the top bit of the 16 bits is 0, then the time division is in "ticks per beat" (or "pulses per quarter note"). If the top bit is 1, then the time division is in "frames per second". This member deals only with the ticks/beat definition.

10.52 seq64::midi_message Class Reference

Provides a handy capsule for a MIDI message, based on the `std::vector<unsigned char>` data type from the RtMidi project.

Public Types

- typedef `std::vector< midibyte >` `container`
Holds the data of the MIDI message.

Public Member Functions

- `midibyte` `midibyte ()`
- `midibyte operator[]` (`int i`) `const`
- `const char *` `array () const`
- `int` `count () const`
- `bool` `empty () const`
- `void` `push (midibyte b)`
- `double` `timestamp () const`
- `void` `timestamp (double t)`
- `bool` `is_sysex () const`
- `void` `show () const`

Private Attributes

- `container` `m_bytes`
Holds the event status and data bytes.
- `double` `m_timestamp`
Holds the (optional) timestamp of the MIDI message.

10.52.1 Detailed Description

Please note that the ALSA module in sequencer64's rtmidi infrastructure uses the [seq64::event](#) rather than the [seq64::midi_message](#) object. For the moment, we will translate between them until we have the interactions between the old and new modules under control.

10.52.2 Member Typedef Documentation

10.52.2.1 container

```
typedef std::vector<midibyte> seq64::midi_message::container
```

Callers should use [midi_message::container](#) rather than using the vector directly. Bytes are added by the [push\(\)](#) function, and are safely accessed (with bounds-checking) by operator [].

10.52.3 Constructor & Destructor Documentation

10.52.3.1 midi_message()

```
seq64::midi_message::midi_message ( )
```

10.52.4 Member Function Documentation

10.52.4.1 operator[]()

```
midibyte seq64::midi_message::operator[] (
    int i ) const [inline]
```

10.52.4.2 array()

```
const char* seq64::midi_message::array ( ) const [inline]
```

10.52.4.3 count()

```
int seq64::midi_message::count ( ) const [inline]
```

10.52.4.4 empty()

```
bool seq64::midi_message::empty ( ) const [inline]
```

10.52.4.5 push()

```
void seq64::midi_message::push (
    midibyte b ) [inline]
```

10.52.4.6 timestamp() [1/2]

```
double seq64::midi_message::timestamp ( ) const [inline]
```

10.52.4.7 timestamp() [2/2]

```
void seq64::midi_message::timestamp (
    double t ) [inline]
```

10.52.4.8 is_sysex()

```
bool seq64::midi_message::is_sysex ( ) const [inline]
```

10.52.4.9 show()

```
void seq64::midi_message::show ( ) const
```

10.52.5 Field Documentation

10.52.5.1 m_bytes

```
container seq64::midi_message::m_bytes [private]
```

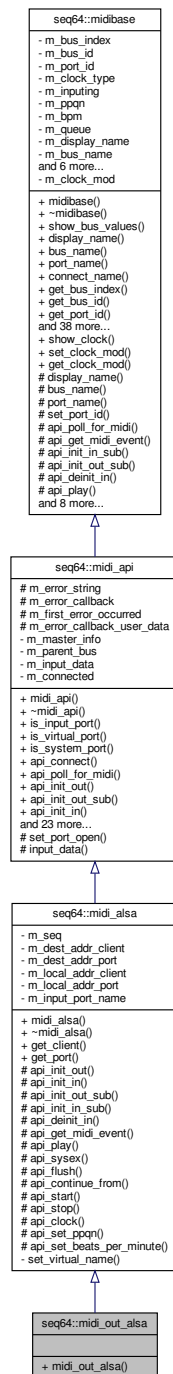
10.52.5.2 m_timestamp

```
double seq64::midi_message::m_timestamp [private]
```

10.53 seq64::midi_out_alsa Class Reference

This class implements the ALSA version of a MIDI output object.

Inheritance diagram for seq64::midi_out_alsa:



Public Member Functions

- [midi_out_alsa](#) ([midibus](#) &parentbus, [midi_info](#) &masterinfo)

Additional Inherited Members

10.53.1 Constructor & Destructor Documentation

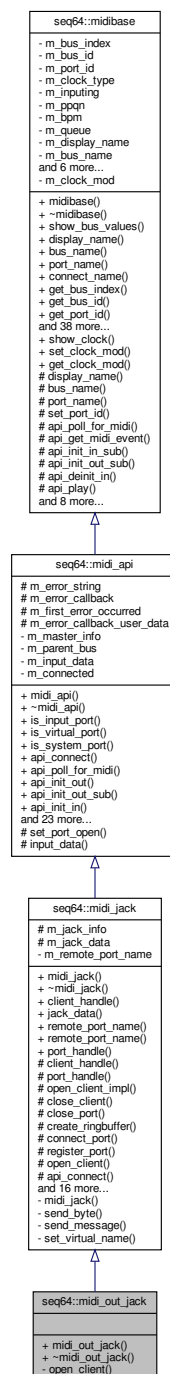
10.53.1.1 `midi_out_alsa()`

```
seq64::midi_out_alsa::midi_out_alsa (
    midibus & parentbus,
    midi_info & masterinfo )
```

10.54 `seq64::midi_out_jack` Class Reference

The JACK MIDI output API class.

Inheritance diagram for seq64::midi_out_jack:



Public Member Functions

- [midi_out_jack](#) ([midibus](#) &parentbus, [midi_info](#) &masterinfo)
- virtual [~midi_out_jack](#) ()

Private Member Functions

- virtual bool [open_client](#) ()

```
virtual bool send_message (const midi\_message & message);
```

Additional Inherited Members

10.54.1 Constructor & Destructor Documentation

10.54.1.1 midi_out_jack()

```
seq64::midi_out_jack::midi_out_jack (
    midibus & parentbus,
    midi\_info & masterinfo )
```

10.54.1.2 ~midi_out_jack()

```
virtual seq64::midi_out_jack::~~midi_out_jack ( ) [virtual]
```

10.54.2 Member Function Documentation

10.54.2.1 open_client()

```
virtual bool seq64::midi_out_jack::open_client ( ) [inline], [private], [virtual]
```

This function is virtual, so we don't call it in the constructor, using [open_client_impl\(\)](#) directly instead. This function replaces the RtMidi function "connect()".

Implements [seq64::midi_jack](#).

10.55 seq64::midi_port_info Class Reference

A class for holding port information.

Data Structures

- struct [port_info_t](#)

Hold the information for a single port.

Public Member Functions

- [midi_port_info](#) ()
- void [add](#) (int clientnumber, const std::string &clientname, int portnumber, const std::string &portname, bool makevirtual, bool makesystem, bool makeinput, int queuenumber=SEQ64_BAD_QUEUE_ID)
- void [add](#) (const [midibus](#) *m)
- void [clear](#) ()

This function is useful in replacing the discovered system ports with the manual/virtual ports added in "manual" mode.

- int [get_port_count](#) () const
- int [get_bus_id](#) (int index) const
- std::string [get_bus_name](#) (int index) const
- int [get_port_id](#) (int index) const
- std::string [get_port_name](#) (int index) const
- bool [get_input](#) (int index) const
- bool [get_virtual](#) (int index) const
- bool [get_system](#) (int index) const
- int [get_queue_number](#) (int index) const
- std::string [connect_name](#) (int index) const

Provides the bus name and port name in canonical JACK format: "busname:portname".

Private Attributes

- int [m_port_count](#)
Holds the number of ports counted.
- std::vector< [port_info_t](#) > [m_port_container](#)
Holds information on all of the ports that were "scanned".

10.55.1 Constructor & Destructor Documentation

10.55.1.1 midi_port_info()

```
seq64::midi_port_info::midi_port_info ( )
```

10.55.2 Member Function Documentation

10.55.2.1 add() [1/2]

```
void seq64::midi_port_info::add (
    int clientnumber,
    const std::string & clientname,
    int portnumber,
    const std::string & portname,
    bool makevirtual,
    bool makesystem,
    bool makeinput,
    int queuenumber = SEQ64_BAD_QUEUE_ID )
```

10.55.2.2 add() [2/2]

```
void seq64::midi_port_info::add (
    const midibus * m )
```

10.55.2.3 clear()

```
void seq64::midi_port_info::clear ( ) [inline]
```

10.55.2.4 get_port_count()

```
int seq64::midi_port_info::get_port_count ( ) const [inline]
```

10.55.2.5 get_bus_id()

```
int seq64::midi_port_info::get_bus_id (
    int index ) const [inline]
```

10.55.2.6 get_bus_name()

```
std::string seq64::midi_port_info::get_bus_name (
    int index ) const [inline]
```

10.55.2.7 get_port_id()

```
int seq64::midi_port_info::get_port_id (
    int index ) const [inline]
```

10.55.2.8 get_port_name()

```
std::string seq64::midi_port_info::get_port_name (
    int index ) const [inline]
```

10.55.2.9 get_input()

```
bool seq64::midi_port_info::get_input (
    int index ) const [inline]
```

10.55.2.10 get_virtual()

```
bool seq64::midi_port_info::get_virtual (
    int index ) const [inline]
```

10.55.2.11 get_system()

```
bool seq64::midi_port_info::get_system (
    int index ) const [inline]
```

10.55.2.12 get_queue_number()

```
int seq64::midi_port_info::get_queue_number (
    int index ) const [inline]
```

10.55.2.13 connect_name()

```
std::string seq64::midi_port_info::connect_name (
    int index ) const [inline]
```

This function is basically the same as [midibase::connect_name\(\)](#) function. If either the bus name or port name are empty, then an empty string is returned.

10.55.3 Field Documentation

10.55.3.1 m_port_count

```
int seq64::midi_port_info::m_port_count [private]
```

10.55.3.2 m_port_container

```
std::vector<port_info_t> seq64::midi_port_info::m_port_container [private]
```

10.56 seq64::midi_queue Class Reference

Provides a queue of [midi_message](#) structures.

Public Member Functions

- [midi_queue](#) ()
- [~midi_queue](#) ()
- bool [empty](#) () const
'Getter' function for member m_size == 0
- int [count](#) () const
'Getter' function for member m_size == 0
- bool [full](#) () const
- bool [add](#) (const [midi_message](#) &mmsg)
- void [pop](#) ()
- [midi_message pop_front](#) ()
- void [allocate](#) (unsigned queuesize=SEQ64_DEFAULT_QUEUE_SIZE)
- void [deallocate](#) ()
- const [midi_message](#) & [front](#) () const
'Getter' function for member m_ring[m_front]

Private Attributes

- unsigned [m_front](#)
- unsigned [m_back](#)
- unsigned [m_size](#)
- unsigned [m_ring_size](#)
- [midi_message](#) * [m_ring](#)

10.56.1 Detailed Description

This entity used to be a plain structure nested in the `midi_in_api` class. We made it a class to encapsulate some common operations to save a burden on the callers.

10.56.2 Constructor & Destructor Documentation

10.56.2.1 midi_queue()

```
seq64::midi_queue::midi_queue ( )
```

10.56.2.2 ~midi_queue()

```
seq64::midi_queue::~~midi_queue ( )
```

10.56.3 Member Function Documentation

10.56.3.1 empty()

```
bool seq64::midi_queue::empty ( ) const [inline]
```

10.56.3.2 count()

```
int seq64::midi_queue::count ( ) const [inline]
```

10.56.3.3 full()

```
bool seq64::midi_queue::full ( ) const [inline]
```

Returns

Returns true if the queue size is at maximum.

10.56.3.4 add()

```
bool seq64::midi_queue::add (
    const midi\_message & mmsg )
```

10.56.3.5 pop()

```
void seq64::midi_queue::pop ( )
```

10.56.3.6 pop_front()

```
midi_message seq64::midi_queue::pop_front ( )
```

10.56.3.7 allocate()

```
void seq64::midi_queue::allocate (
    unsigned queue_size = SEQ64_DEFAULT_QUEUE_SIZE )
```

10.56.3.8 deallocate()

```
void seq64::midi_queue::deallocate ( )
```

10.56.3.9 front()

```
const midi_message& seq64::midi_queue::front ( ) const [inline]
```

10.56.4 Field Documentation

10.56.4.1 m_front

```
unsigned seq64::midi_queue::m_front [private]
```

10.56.4.2 m_back

```
unsigned seq64::midi_queue::m_back [private]
```

10.56.4.3 m_size

```
unsigned seq64::midi_queue::m_size [private]
```

10.56.4.4 m_ring_size

```
unsigned seq64::midi_queue::m_ring_size [private]
```

10.56.4.5 m_ring

```
midi_message* seq64::midi_queue::m_ring [private]
```

10.57 seq64::midi_splitter Class Reference

This class handles the parsing and writing of MIDI files.

Public Member Functions

- [midi_splitter](#) (int [ppqn](#)=SEQ64_USE_DEFAULT_PPQN)
Principal constructor.
- [~midi_splitter](#) ()
A rote destructor.
- bool [log_main_sequence](#) ([sequence](#) &seq, int seqnum)
Logs the main sequence (an SMF 0 track) for later usage in splitting the track.
- void [initialize](#) ()
Resets the SMF 0 support variables in preparation for parsing a new MIDI file.
- void [increment](#) (int channel)
Processes a channel number by raising its flag in the `m_smf0_channels[]` array.
- bool [split](#) ([perform](#) &p, int screenset)
This function splits an SMF 0 file, splitting all of the channels in the sequence out into separate sequences, and adding each to the perform object.
- int [ppqn](#) () const
'Getter' function for member `m_ppqn` Provides a way to get the actual value of PPQN used in processing the sequences when `parse()` was called.
- int [count](#) () const
'Getter' function for member `m_smf0_channels_count`

Private Member Functions

- bool [split_channel](#) (const [sequence](#) &main_seq, [sequence](#) *seq, int channel)
This function splits the given sequence into a new sequence, for the given channel found in the SMF 0 track.

Private Attributes

- int `m_ppqn`
Provides the current value of the PPQN, which used to be constant and is now only the macro `DEFAULT_PPQN`.
- bool `m_use_default_ppqn`
Indicates that the default PPQN is in force.
- int `m_smf0_channels_count`
Provides support for SMF 0, indicates how many channels were found in the file in a single sequence.
- bool `m_smf0_channels` [16]
Provides support for SMF 0, holds a bool value that indicates the occurrence of a given channel.
- `sequence *` `m_smf0_main_sequence`
Provides support for SMF 0, points to the initial SMF 0 sequence, from which the single-channel sequences will be created.
- int `m_smf0_seq_number`
Provides support for SMF 0, holds the prospective sequence number of the main (SMF 0) sequence.

10.57.1 Detailed Description

In addition to the standard MIDI tracks, it also handles some "private" or "proprietary" tracks specific to Seq24. It does not, however, handle SYSEX events.

10.57.2 Constructor & Destructor Documentation

10.57.2.1 `midi_splitter()`

```
seq64::midi_splitter::midi_splitter (
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

Parameters

<i>ppqn</i>	<p>Provides the initial value of the PPQN setting. It is handled differently for parsing (reading) versus writing the MIDI file.</p> <ul style="list-style-type: none"> • Reading. <ul style="list-style-type: none"> – If set to <code>SEQ64_USE_DEFAULT_PPQN</code>, the legacy application behavior is used. The <code>m_ppqn</code> member is set to the default PPQN, <code>DEFAULT_PPQN</code>. The value read from the MIDI file, <code>ppqn</code>, is then use to scale the running-time of the sequence relative to <code>DEFAULT_PPQN</code>. – Otherwise, <code>m_ppqn</code> is set to the value read from the MIDI file. No scaling is done. Since the value gets written, specify <code>ppqn</code> as 0, an obviously bogus value, to get this behavior. • Writing. This value is written to the MIDI file in the header chunk of the song. Note that the caller must query for the PPQN set during parsing, and pass it to the constructor when preparing to write the file. See how it is done in the <code>mainwnd</code> class.
-------------	--

10.57.2.2 ~midi_splitter()

```
seq64::midi_splitter::~~midi_splitter ( )
```

10.57.3 Member Function Documentation

10.57.3.1 log_main_sequence()

```
bool seq64::midi_splitter::log_main_sequence (
    sequence & seq,
    int seqnum )
```

/param seq The main sequence to be logged.

/param seqnum The sequence number of the main sequence.

/return Returns true if the main sequence's address was logged, and false if it was already logged.

10.57.3.2 initialize()

```
void seq64::midi_splitter::initialize ( )
```

10.57.3.3 increment()

```
void seq64::midi_splitter::increment (
    int channel )
```

If it is the first entry for that channel, `m_smf0_channels_count` is incremented. We won't check the channel number, to save time, until someday we segfault :-D

Parameters

<i>channel</i>	The MIDI channel number. The caller is responsible to make sure it ranges from 0 to 15.
----------------	---

10.57.3.4 split()

```
bool seq64::midi_splitter::split (
    perform & p,
    int screenset )
```

Lastly, it adds the SMF 0 track as the last track; the user can then examine it before removing it. Is this worth the effort?

There is a little oddity, in that, if the SMF 0 track has events for only one channel, this code will still create a new sequence, as well as the main sequence. Not sure if this is worth extra code to just change the channels on the main sequence and put it into the correct track for the one channel it contains. In fact, we just want to keep it in pattern slot number 16, to keep it out of the way.

Parameters

<i>p</i>	Provides a reference to the perform object into which sequences/tracks are to be added.
<i>screenset</i>	The screen-set offset to be used when loading a sequence (track) from the file.

Returns

Returns true if the parsing succeeded. Returns false if no SMF 0 main sequence was logged.

10.57.3.5 ppqn()

```
int seq64::midi_splitter::ppqn ( ) const [inline]
```

The PPQN will be either the global ppqn (legacy behavior) or the value read from the file, depending on the ppqn parameter passed to the [midi_splitter](#) constructor.

10.57.3.6 count()

```
int seq64::midi_splitter::count ( ) const [inline]
```

10.57.3.7 split_channel()

```
bool seq64::midi_splitter::split_channel (
    const sequence & main_seq,
    sequence * s,
    int channel ) [private]
```

It is called for each possible channel, resulting in multiple passes over the SMF 0 track.

Note that the events that are read from the MIDI file have delta times. Sequencer64 converts these delta times to cumulative times. We need to preserve that here. Conversion back to delta times is needed only when saving the sequences to a file. This is done in [midi_container::fill\(\)](#).

We have to accumulate the delta times in order to be able to set the length of the sequence in pulses.

Luckily, we don't have to worry about copying triggers, since the imported SMF 0 track won't have any Seq24/↔ Sequencer24 triggers.

It doesn't set the sequence number of the sequence; that is set when the sequence is added to the perform object.

Parameters

<i>main_seq</i>	This parameter is the whole SMF 0 track that was read from the MIDI file. It contains all of the channel data that needs to be split into separate sequences.
<i>s</i>	Provides the new sequence that needs to have its settings made, and all of the selected channel events added to it.
<i>channel</i>	Provides the MIDI channel number (re 0) that marks the channel data the needs to be extracted and added to the new sequence. If this channel is 0, then we need to add certain Meta events to this sequence, as well. So far we support only Tempo Meta events.

Returns

Returns true if at least one event got added. If none were added, the caller should delete the sequence object represented by parameter *s*.

10.57.4 Field Documentation

10.57.4.1 m_ppqn

```
int seq64::midi_splitter::m_ppqn [private]
```

10.57.4.2 m_use_default_ppqn

```
bool seq64::midi_splitter::m_use_default_ppqn [private]
```

10.57.4.3 m_smf0_channels_count

```
int seq64::midi_splitter::m_smf0_channels_count [private]
```

SMF 1 file parsing will only warn about more than one channel found in a given sequence.

10.57.4.4 m_smf0_channels

```
bool seq64::midi_splitter::m_smf0_channels[16] [private]
```

Obviously, we don't have to worry about multiple MIDI busses.

10.57.4.5 m_smf0_main_sequence

```
sequence* seq64::midi_splitter::m_smf0_main_sequence [private]
```

10.57.4.6 m_smf0_seq_number

```
int seq64::midi_splitter::m_smf0_seq_number [private]
```

We want to be able to add that sequence last, for easier and cleaner removal of that sequence by the user.

10.58 seq64::midi_timing Class Reference

We anticipate the need to have a small structure holding the parameters needed to calculate MIDI times within an arbitrary song.

Public Member Functions

- [midi_timing](#) ()
Defaults constructor for [midi_timing](#).
- [midi_timing](#) (midibpm bpmminute, int bpmmeasure, int beatwidth, int ppqn)
Principal constructor for [midi_timing](#).
- [midibpm beats_per_minute](#) () const
'Getter' function for member m_beats_per_minute
- void [beats_per_minute](#) (midibpm b)
'Setter' function for member m_beats_per_minute
- int [beats_per_measure](#) () const
'Getter' function for member m_beats_per_measure
- void [beats_per_measure](#) (int b)
'Setter' function for member m_beats_per_measure
- int [beat_width](#) () const
'Getter' function for member m_beats_per_beat_width
- void [beat_width](#) (int bw)
'Setter' function for member m_beats_per_beat_width
- int [ppqn](#) () const
'Getter' function for member m_ppqn
- void [ppqn](#) (int p)
'Setter' function for member m_ppqn

Private Attributes

- [midibpm m_beats_per_minute](#)
This value should match the BPM value selected when editing the song.
- int [m_beats_per_measure](#)
This value should match the numerator value selected when editing the sequence.
- int [m_beat_width](#)
This value should match the denominator value selected when editing the sequence.
- int [m_ppqn](#)
This value provides the precision of the MIDI song.

10.58.1 Detailed Description

Although Seq24/Sequencer64 currently are heavily dependent on hard-wired values, that will be rectified eventually, so let us get ready for it.

10.58.2 Constructor & Destructor Documentation

10.58.2.1 midi_timing() [1/2]

```
seq64::midi_timing::midi_timing ( )
```

10.58.2.2 midi_timing() [2/2]

```
seq64::midi_timing::midi_timing (
    midibpm bpmminute,
    int bpmmeasure,
    int beatwidth,
    int ppqn )
```

Parameters

<i>bpmminute</i>	Copied into the m_beats_per_minute member.
<i>bpmmeasure</i>	Copied into the m_beats_per_measure member.
<i>beatwidth</i>	Copied into the m_beat_width member.
<i>ppqn</i>	Copied into the m_ppqn member.

10.58.3 Member Function Documentation

10.58.3.1 beats_per_minute() [1/2]

```
midibpm seq64::midi_timing::beats_per_minute ( ) const [inline]
```

10.58.3.2 beats_per_minute() [2/2]

```
void seq64::midi_timing::beats_per_minute (
    midibpm b ) [inline]
```

Parameters

<i>b</i>	The value to which to set the number of beats/minute. We can add validation later.
----------	--

10.58.3.3 beats_per_measure() [1/2]

```
int seq64::midi_timing::beats_per_measure ( ) const [inline]
```

10.58.3.4 beats_per_measure() [2/2]

```
void seq64::midi_timing::beats_per_measure (
    int b ) [inline]
```

Parameters

<i>b</i>	The value to which to set the number of beats/measure. We can add validation later.
----------	---

10.58.3.5 beat_width() [1/2]

```
int seq64::midi_timing::beat_width ( ) const [inline]
```

10.58.3.6 beat_width() [2/2]

```
void seq64::midi_timing::beat_width (
    int bw ) [inline]
```

Parameters

<i>bw</i>	The value to which to set the number of beats in the denominator of the time signature. We can add validation later.
-----------	--

10.58.3.7 ppqn() [1/2]

```
int seq64::midi_timing::ppqn ( ) const [inline]
```

10.58.3.8 ppqn() [2/2]

```
void seq64::midi_timing::ppqn (
    int p ) [inline]
```

Parameters

<i>p</i>	The value to which to set the PPQN member. We can add validation later.
----------	---

10.58.4 Field Documentation

10.58.4.1 m_beats_per_minute

```
midibpm seq64::midi_timing::m_beats_per_minute [private]
```

This value is most commonly set to 120, but is also read from the MIDI file. This value is needed if one want to calculate durations in true time units such as seconds, but is not needed to calculate the number of pulses/ticks/divisions.

10.58.4.2 m_beats_per_measure

```
int seq64::midi_timing::m_beats_per_measure [private]
```

This value is most commonly set to 4.

10.58.4.3 m_beat_width

```
int seq64::midi_timing::m_beat_width [private]
```

This value is most commonly set to 4, meaning that the fundamental beat unit is the quarter note.

10.58.4.4 m_ppqn

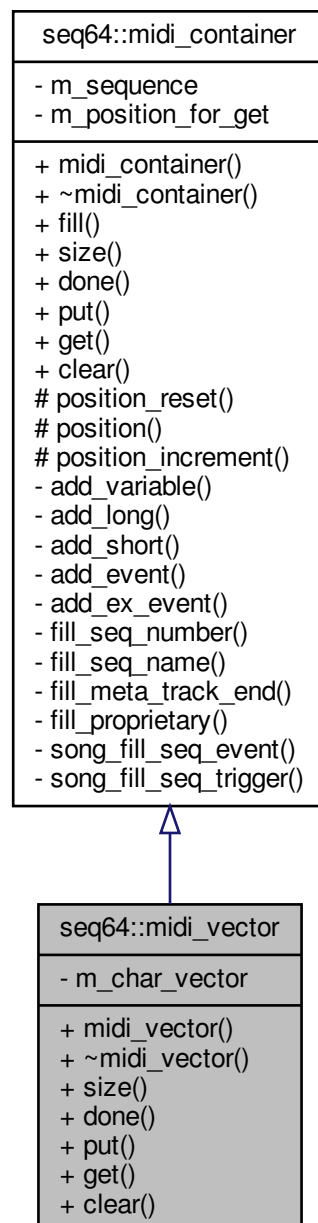
```
int seq64::midi_timing::m_ppqn [private]
```

This value is most commonly set to 192, but is also read from the MIDI file. We are still working getting "non-standard" values to work.

10.59 seq64::midi_vector Class Reference

This class is the `std::vector` implementation of the [midi_container](#).

Inheritance diagram for `seq64::midi_vector`:



Public Member Functions

- [midi_vector](#) ([sequence](#) &seq)

This constructor fills in the members of this class.

- virtual `~midi_vector()`

A rote constructor needed for a base class.

- virtual `std::size_t size()` const
- virtual `bool done()` const

For iterating through the data in the MIDI vector, we are done when we've gotten the last element of the container.

- virtual `void put(midibyte b)`

Provides a way to add a MIDI byte into the list.

- virtual `midibyte get()` const

Provide a way to get the next byte from the container.

- virtual `void clear()`

Provides a way to clear the container.

Private Types

- `typedef std::vector<midibyte> CharVector`

Provides the type of this container.

Private Attributes

- `CharVector m_char_vector`

The container itself.

Additional Inherited Members

10.59.1 Member Typedef Documentation

10.59.1.1 CharVector

```
typedef std::vector<midibyte> seq64::midi_vector::CharVector [private]
```

10.59.2 Constructor & Destructor Documentation

10.59.2.1 midi_vector()

```
seq64::midi_vector::midi_vector (
    sequence & seq )
```

Parameters

<code>seq</code>	Provides a reference to the sequence/track for which this container holds MIDI data.
------------------	--

10.59.2.2 ~midi_vector()

```
virtual seq64::midi_vector::~~midi_vector ( ) [inline], [virtual]
```

10.59.3 Member Function Documentation

10.59.3.1 size()

```
virtual std::size_t seq64::midi_vector::size ( ) const [inline], [virtual]
```

Returns

Returns the size of the container, in midibytes.

Reimplemented from [seq64::midi_container](#).

10.59.3.2 done()

```
virtual bool seq64::midi_vector::done ( ) const [inline], [virtual]
```

Returns

Returns true if the position is greater than or equal to the size of the character vector.

Reimplemented from [seq64::midi_container](#).

10.59.3.3 put()

```
virtual void seq64::midi_vector::put (
    midibyte b ) [inline], [virtual]
```

The original seq24 list used an std::list and a push_front operation.

Parameters

<i>b</i>	Provides the MIDI byte to push_back() into the character vector.
----------	--

Implements [seq64::midi_container](#).

10.59.3.4 get()

```
virtual midibyte seq64::midi_vector::get ( ) const [inline], [virtual]
```

In this implementation, `m_position_for_get` is used. As a side-effect, the position value is incremented.

Returns

Returns the next byte in the character vector.

Implements [seq64::midi_container](#).

10.59.3.5 clear()

```
virtual void seq64::midi_vector::clear ( ) [inline], [virtual]
```

Implements [seq64::midi_container](#).

10.59.4 Field Documentation

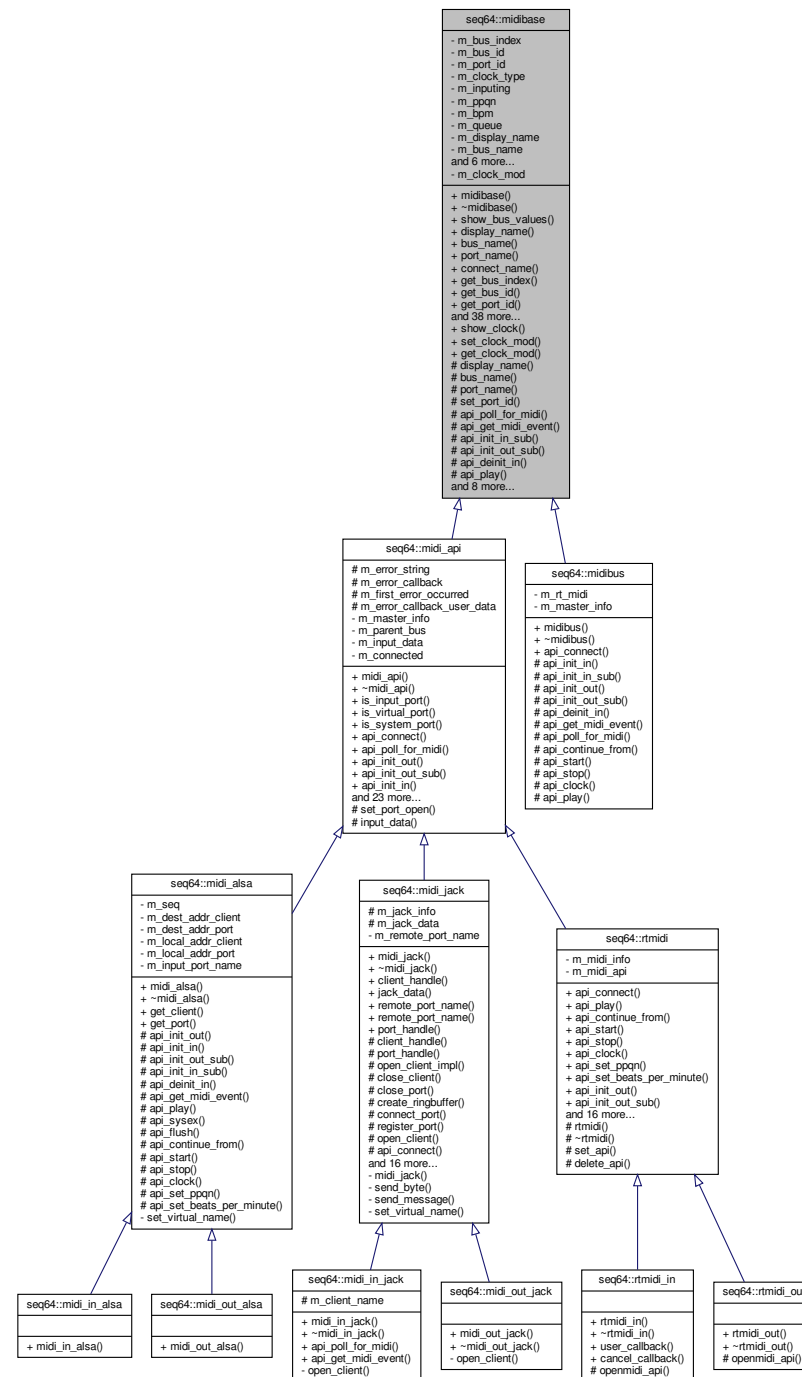
10.59.4.1 m_char_vector

```
CharVector seq64::midi_vector::m_char_vector [private]
```

10.60 seq64::midibase Class Reference

This class implements with ALSA version of the midibase object.

Inheritance diagram for seq64::midibase:



Public Member Functions

- [midibase](#) (const std::string &appname, const std::string &busname="", const std::string &portname="", int index=0, int bus_id=SEQ64_NO_BUS, int port_id=SEQ64_NO_PORT, int queue=SEQ64_NO_QUEUE,

int `ppqn`=SEQ64_USE_DEFAULT_PPQN, `midibpm bpm`=SEQ64_DEFAULT_BPM, bool `makevirtual`=false, bool `isinput`=false, bool `makesystem`=false)

Creates a normal MIDI port, which will correspond to an existing system MIDI port, such as one provided by Timidity or a running JACK application, or a virtual port, which has a name made up by the application.

- virtual `~midibase ()`
A rote empty destructor.
- void `show_bus_values ()`
Shows most midibase members.
- const std::string & `display_name ()` const
'Getter' function for member m_display_name
- const std::string & `bus_name ()` const
'Getter' function for member m_bus_name
- const std::string & `port_name ()` const
'Getter' function for member m_port_name
- std::string `connect_name ()` const
'Getter' function for member m_bus_name and m_port_name Concatenates the bus and port names into a string of the form "busname:portname".
- int `get_bus_index ()` const
'Getter' function for member m_bus_index
- int `get_bus_id ()` const
'Getter' function for member m_bus_id
- int `get_port_id ()` const
'Getter' function for member m_port_id
- int `ppqn ()` const
'Getter' function for member m_ppqn;
- `midibpm bpm ()` const
'Getter' function for member m_bpm;
- bool `match (int bus, int port)`
Checks if the given parameters match the current bus and port numbers.
- bool `is_virtual_port ()` const
'Getter' function for member m_is_virtual_port
- void `is_virtual_port (bool flag)`
*'Setter' function for member m_is_virtual_port This function is needed in the rtmidi library to set the is-virtual flag in the api_init_*_sub() functions, so that midi_alsa, midi_jack (and any other additional APIs that end up supported by our heavily-refactored rtmidi library), as well as the original midibus, can know that they represent a virtual port.*
- bool `is_input_port ()` const
'Getter' function for member m_is_input_port
- bool `is_output_port ()` const
'Getter' function for member ! m_is_input_port
- void `is_input_port (bool flag)`
'Setter' function for member m_is_input_port
- bool `is_system_port ()` const
'Getter' function for member m_is_system_port
- void `set_system_port_flag ()`
'Setter' function for member m_is_system_port Can only set it to true.
- void `set_clock (clock_e clocktype)`
'Setter' function for member m_clock_type We removed the redundant set_clock_status() function.
- `clock_e get_clock ()` const
'Getter' function for member m_clock_type
- bool `port_disabled ()` const
'Getter' function for member m_clock_type
- bool `clock_enabled ()` const

- 'Getter' function for member m_clock_type*
- bool `get_input` () const
- 'Getter' function for member m_inputing*
- void `set_input_status` (bool flag)
- 'Setter' function for member m_inputing*
- int `queue_number` () const
- 'Getter' function for member m_queue*
- void `set_bus_id` (int id)
- 'Setter' function for member m_bus_id Useful for setting the buss ID when using the `rtmidi_info` object to create a list of busses and ports.*
- void `set_name` (const std::string &appname, const std::string &busname, const std::string &portname)
- Sets the name of the buss by assembling the name components obtained from the system in a straightforward manner:*
- void `set_alt_name` (const std::string &appname, const std::string &busname, const std::string &portname)
- Sets the name of the buss in a different way.*
- void `set_multi_name` (const std::string &appname, const std::string &localbusname, const std::string &remoteportname)
- Sets the name of the buss in yet another different way, suitable for the multiclient mode of some APIs (such as JACK).*
- int `poll_for_midi` ()
- Polls for MIDI events.*
- bool `get_midi_event` (event *inev)
- Obtains a MIDI event.*
- bool `init_out` ()
- Initialize the MIDI output port.*
- bool `init_in` ()
- Initialize the MIDI input port.*
- bool `deinit_in` ()
- Deinitialize the MIDI input.*
- bool `init_out_sub` ()
- Initialize the output in a different way?*
- bool `init_in_sub` ()
- Initialize the output in a different way?*
- void `play` (event *e24, midibyte channel)
- This `play()` function takes a native event, encodes it to a MIDI sequencer event, sets the broadcasting to the subscribers, sets the direct-passing mode to send the event without queueing, and puts it in the queue.*
- void `sysex` (event *e24)
- Takes a native SYSEX event, encodes it to an ALSA event, and then puts it in the queue.*
- void `flush` ()
- Flushes our local queue events out into ALSA.*
- void `start` ()
- This function gets the MIDI clock a-runnin', if the clock type is not `e_clock_off` or `e_clock_disabled`.*
- void `stop` ()
- Stop the MIDI buss.*
- void `clock` (midipulse tick)
- Generates the MIDI clock, starting at the given tick value.*
- void `continue_from` (midipulse tick)
- Continue from the given tick.*
- void `init_clock` (midipulse tick)
- Initialize the clock, continuing from the given tick.*
- void `print` ()
- Prints m_name.*
- bool `set_input` (bool inputing)
- Set status to of "inputting" to the given value.*

Static Public Member Functions

- static void [show_clock](#) (const std::string &context, [midipulse](#) tick)
A static debug function, enabled only for trouble-shooting.
- static void [set_clock_mod](#) (int clockmod)
Set the clock mod to the given value, if legal.
- static int [get_clock_mod](#) ()
Get the clock mod value.

Protected Member Functions

- void [display_name](#) (const std::string &name)
'Setter' function for member m_display_name
- void [bus_name](#) (const std::string &name)
'Setter' function for member m_bus_name
- void [port_name](#) (const std::string &name)
'Setter' function for member m_port_name
- void [set_port_id](#) (int id)
'Setter' function for member m_port_id Useful for setting the port ID when using the [rtmidi_info](#) object to inspect and create a list of busses and ports.
- virtual int [api_poll_for_midi](#) ()
Now defined in the ALSA implementation, and used by mastermidibus.
- virtual bool [api_get_midi_event](#) ([event](#) *inev)
Used in the JACK implementation.
- virtual bool [api_init_in_sub](#) ()
Not defined in the PortMidi implementation.
- virtual bool [api_init_out_sub](#) ()
Not defined in the PortMidi implementation.
- virtual bool [api_deinit_in](#) ()
Not defined in the PortMidi implementation.
- virtual void [api_play](#) ([event](#) *e24, [midibyte](#) channel)=0
- virtual void [api_sysex](#) ([event](#) *)
Handles implementation details for SysEx messages.
- virtual void [api_flush](#) ()
Handles implementation details for the [flush\(\)](#) function.
- virtual bool [api_init_in](#) ()=0
- virtual bool [api_init_out](#) ()=0
- virtual void [api_continue_from](#) ([midipulse](#) tick, [midipulse](#) beats)=0
- virtual void [api_start](#) ()=0
- virtual void [api_stop](#) ()=0
- virtual void [api_clock](#) ([midipulse](#) tick)=0

Private Attributes

- const int [m_bus_index](#)
Provides the index of the midibase object in either the input list or the output list.
- int [m_bus_id](#)
The buss ID of the midibase object.
- int [m_port_id](#)
The port ID of the midibase object.

- [clock_e m_clock_type](#)
The type of clock to use.
- [bool m_inputting](#)
This flag indicates if an input bus has been selected for action as an input device (such as a MIDI controller).
- [int m_ppqn](#)
Provides the PPQN value in force, currently a constant.
- [midibpm m_bpm](#)
Provides the PPQN value in force, currently a constant.
- [int m_queue](#)
Another ID of the MIDI queue? This is an implementation-dependent value.
- [std::string m_display_name](#)
Holds the full display name of the bus, index, ID numbers, and item names.
- [std::string m_bus_name](#)
The name of the MIDI buss.
- [std::string m_port_name](#)
The name of the MIDI port.
- [midipulse m_lasttick](#)
The last (most recent? final?) tick.
- [bool m_is_virtual_port](#)
Indicates if the port is to be a virtual port.
- [bool m_is_input_port](#)
Indicates if the port is to be an input (versus output) port.
- [bool m_is_system_port](#)
Indicates if the port is a system port.
- [mutex m_mutex](#)
Locking mutex.

Static Private Attributes

- [static int m_clock_mod](#)
*This is another name for "16 * 4".*

Friends

- [class mastermidibus](#)
The master MIDI bus sets up the buss.

10.60.1 Constructor & Destructor Documentation

10.60.1.1 midibase()

```
seq64::midibase::midibase (
    const std::string & appname,
    const std::string & busname = "",
    const std::string & portname = "",
    int index = 0,
    int bus_id = SEQ64_NO_BUS,
    int port_id = SEQ64_NO_PORT,
    int queue = SEQ64_NO_QUEUE,
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM,
    bool makevirtual = false,
    bool isinput = false,
    bool makesystem = false )
```

Provides a constructor with client number, port number, name of client, name of port.

This constructor is the one that seems to be the one that is used for the MIDI input and output busses, when the [manual-alsa-ports] option is *not* in force. Also used for the announce buss, and in the [mastermidibase::port_start\(\)](#) function.

Parameters

<i>appname</i>	Provides the the name of the application. The derived class will determine this name.
<i>busname</i>	Provides the ALSA client name or the MIDI subsystem name (e.g. "TiMidity"). If empty, a name will be assembled by the derived class at port-setup time.
<i>portname</i>	Provides the port name. This item defaults to empty, which means the port name should be obtained via the API, or be assembled by the derived class at port-setup time.
<i>index</i>	Provides the ordinal of this buss/port, mostly for display purposes.
<i>bus_id</i> Indicates the port ID. Defaults to SEQ64_NO_PORT. If SEQ64_NO_PORT, the derived class will get the port ID at port-setup time.	
<i>queue</i> Provides the PPQN value. Defaults to SEQ64_USE_DEFAULT_PPQN.	
<i>bpm</i>	Provides the BPM value. Defaults to SEQ64_DEFAULT_BPM.
<i>makevirtual</i>	Indicates that the port represented by this object is to be virtual. Defaults to false. This could also be set via the init_in() , init_out() , init_in_sub() , or init_out_sub() routines. Doing it here seems okay.
<i>isinput</i>	Indicates that this midibus represents and input port, as opposed to an output port.
<i>makesystem</i>	Indicates that the port represented by this object is a system port. Currently true only for ALSA system ports (timer or announce ports).

10.60.1.2 ~midibase()

```
seq64::midibase::~~midibase ( ) [virtual]
```

10.60.2 Member Function Documentation

10.60.2.1 show_bus_values()

```
void seq64::midibase::show_bus_values ( )
```

10.60.2.2 show_clock()

```
void seq64::midibase::show_clock (
    const std::string & context,
    midipulse tick ) [static]
```

Parameters

<i>context</i>	Human readable context information (e.g. "ALSA").
<i>tick</i>	Provides the current tick value.

10.60.2.3 display_name() [1/2]

```
const std::string& seq64::midibase::display_name ( ) const [inline]
```

10.60.2.4 bus_name() [1/2]

```
const std::string& seq64::midibase::bus_name ( ) const [inline]
```

10.60.2.5 port_name() [1/2]

```
const std::string& seq64::midibase::port_name ( ) const [inline]
```

10.60.2.6 connect_name()

```
std::string seq64::midibase::connect_name ( ) const
```

If either name is empty, an empty string is returned.

10.60.2.7 get_bus_index()

```
int seq64::midibase::get_bus_index ( ) const [inline]
```

10.60.2.8 get_bus_id()

```
int seq64::midibase::get_bus_id ( ) const [inline]
```

10.60.2.9 get_port_id()

```
int seq64::midibase::get_port_id ( ) const [inline]
```

10.60.2.10 ppqn()

```
int seq64::midibase::ppqn ( ) const [inline]
```

10.60.2.11 bpm()

```
midibpm seq64::midibase::bpm ( ) const [inline]
```

10.60.2.12 match()

```
bool seq64::midibase::match (
    int bus,
    int port ) [inline]
```

10.60.2.13 is_virtual_port() [1/2]

```
bool seq64::midibase::is_virtual_port ( ) const [inline]
```

10.60.2.14 is_virtual_port() [2/2]

```
void seq64::midibase::is_virtual_port (
    bool flag ) [inline]
```

10.60.2.15 is_input_port() [1/2]

```
bool seq64::midibase::is_input_port ( ) const [inline]
```

10.60.2.16 is_output_port()

```
bool seq64::midibase::is_output_port ( ) const [inline]
```

10.60.2.17 is_input_port() [2/2]

```
void seq64::midibase::is_input_port (
    bool flag ) [inline]
```

10.60.2.18 is_system_port()

```
bool seq64::midibase::is_system_port ( ) const [inline]
```

10.60.2.19 set_system_port_flag()

```
void seq64::midibase::set_system_port_flag ( ) [inline]
```

10.60.2.20 set_clock()

```
void seq64::midibase::set_clock (
    clock_e clocktype ) [inline]
```

Parameters

<i>clocktype</i>	The value used to set the clock-type.
------------------	---------------------------------------

10.60.2.21 get_clock()

```
clock_e seq64::midibase::get_clock ( ) const [inline]
```

10.60.2.22 port_disabled()

```
bool seq64::midibase::port_disabled ( ) const [inline]
```

10.60.2.23 clock_enabled()

```
bool seq64::midibase::clock_enabled ( ) const [inline]
```

10.60.2.24 get_input()

```
bool seq64::midibase::get_input ( ) const [inline]
```

10.60.2.25 set_input_status()

```
void seq64::midibase::set_input_status (
    bool flag ) [inline]
```

10.60.2.26 queue_number()

```
int seq64::midibase::queue_number ( ) const [inline]
```

10.60.2.27 set_bus_id()

```
void seq64::midibase::set_bus_id (
    int id ) [inline]
```

Would be protected, but [midi_alsa](#) needs to change this value to reflect the user-client ID actually assigned by ALSA. (That value ranges from 128 to 191.)

10.60.2.28 set_name()

```
void seq64::midibase::set_name (
    const std::string & appname,
    const std::string & busname,
    const std::string & portname )
```

[0] 128:2 [seq64:seq64](#) port 2

We want to see if the user has configured a port name. If so, and this is an output port, then the buss name is overridden by the entry in the "usr" configuration file. Otherwise, we fall back to the parameters. Note that this has been tweaked versus Seq24, where the "usr" devices were also applied to the input ports. Also note that the "usr" device names should be kept short, and the actual buss name from the system is shown in brackets.

Parameters

<i>appname</i>	This is the name of the client, or application. Not to be confused with the ALSA client-name, which is actually a buss or subsystem name.
<i>busname</i>	Provides the name of the sub-system, such as "Midi Through" or "TiMidity".
<i>portname</i>	Provides the name of the port. In ALSA, this is something like "busname port X".

10.60.2.29 set_alt_name()

```
void seq64::midibase::set_alt_name (
    const std::string & appname,
    const std::string & busname,
    const std::string & portname )
```

If the port is virtual, this function just calls [set_name\(\)](#). Otherwise, it reassembles the name so that it refers to a port found on the system, but modified to make it a unique application port. For example:

```
[0] 128:0 yoshimi:midi in
```

is transformed to this:

```
[0] 128:0 seq64:yoshimi midi in
```

As a side-effect, the "short" portname is changed, from (for example) "midi in" to "yoshimi midi in".

Parameters

<i>appname</i>	This is the name of the client, or application. Not to be confused with the ALSA/JACK client-name, which is actually a buss or subsystem name.
<i>busname</i>	Provides the name of the sub-system, such as "Midi Through", "TiMidity", or "seq64".
<i>portname</i>	Provides the name of the port. In JACK, this should be the full port name, such as "qmidiarp:in".

10.60.2.30 set_multi_name()

```
void seq64::midibase::set_multi_name (
    const std::string & appname,
    const std::string & localbusname,
    const std::string & remoteportname )
```

If the port is virtual, this function just calls [set_name\(\)](#). Otherwise, it reassembles the name so that it refers to a port found on the system, but modified to make it a unique client port. For example:

```
[0] 128:0 yoshimi:midi in
```

is transformed to this:

```
[0] 128:0 seq64-yoshimi:midi in
```

The name in the latter is the original buss name, "seq64" plus the remote port's buss name (extracted from the long name), plus the remote port's short port name (extracted from the long name).

Internal parameter:

Parameters

<i>appname</i>	This is the name of the client, or application. Not to be confused with the ALSA client-name, which is actually a buss or subsystem name.
<i>localbusname</i>	Provides the name of the sub-system, such as "Midi Through", "TiMidity", "yoshimi", or "seq64". It is assumed this parameter has already been set properly.
<i>remoteportname</i>	Provides the name of the port. In JACK, this should be the long port name, such as "qmidiarp:in" or "yoshimi:midi in". It is assumed this parameter has already been set properly.

10.60.2.31 set_clock_mod()

```
static void seq64::midibase::set_clock_mod (
    int clockmod ) [inline], [static]
```

Parameters

<i>clockmod</i>	If this value is not equal to 0, it is used to set the static member <code>m_clock_mod</code> .
-----------------	---

10.60.2.32 `get_clock_mod()`

```
static int seq64::midibase::get_clock_mod ( ) [inline], [static]
```

10.60.2.33 `poll_for_midi()`

```
int seq64::midibase::poll_for_midi ( )
```

EXPERIMENTAL FIX FOR PORTMIDI BUG BUT NEEDED FOR ALL.

Returns

Returns a value greater than 0 if MIDI events are available. Otherwise 0 is returned, or -1 for some APIs (ALSA) when an internal error occurs.

10.60.2.34 `get_midi_event()`

```
bool seq64::midibase::get_midi_event (
    event * inev )
```

Parameters

<i>inev</i>	Points the event to be filled with the MIDI event data.
-------------	---

Returns

Returns true if an event was found, thus making the return parameter useful.

10.60.2.35 `init_out()`

```
bool seq64::midibase::init_out ( )
```

Returns

Returns true unless setting up MIDI failed in some way.

10.60.2.36 init_in()

```
bool seq64::midibase::init_in ( )
```

Returns

Returns true unless setting up MIDI failed in some way.

10.60.2.37 deinit_in()

```
bool seq64::midibase::deinit_in ( )
```

Set the input and the output ports. The destination port is actually our local port.

Returns

Returns true, unless an error occurs.

10.60.2.38 init_out_sub()

```
bool seq64::midibase::init_out_sub ( )
```

Returns

Returns true unless setting up the ALSA port failed in some way.

10.60.2.39 init_in_sub()

```
bool seq64::midibase::init_in_sub ( )
```

Returns

Returns true unless setting up the ALSA port failed in some way.

10.60.2.40 play()

```
void seq64::midibase::play (
    event * e24,
    midibyte channel )
```

Threadsafe

Parameters

<i>e24</i>	The event to be played on this bus. For speed, we don't bother to check the pointer.
<i>channel</i>	The channel of the playback.

10.60.2.41 sysex()

```
void seq64::midibase::sysex (
    event * e24 )
```

Parameters

<i>e24</i>	The event to be handled.
------------	--------------------------

10.60.2.42 flush()

```
void seq64::midibase::flush ( )
```

10.60.2.43 start()

```
void seq64::midibase::start ( )
```

10.60.2.44 stop()

```
void seq64::midibase::stop ( )
```

10.60.2.45 clock()

```
void seq64::midibase::clock (
    midipulse tick )
```

The number of ticks needed is calculated.

Threadsafe

Parameters

<i>tick</i>	Provides the starting tick.
-------------	-----------------------------

10.60.2.46 continue_from()

```
void seq64::midibase::continue_from (
    midipulse tick )
```

Tell the device that we are going to start at a certain position (starting_tick). If there is anything left, then wait for next beat (16th note) to start clocking.

Parameters

<i>tick</i>	The continuing tick.
-------------	----------------------

10.60.2.47 init_clock()

```
void seq64::midibase::init_clock (
    midipulse tick )
```

This function doesn't depend upon the MIDI API in use. Here, e_clock_off and e_clock_disabled have the same effect... none.

Parameters

<i>tick</i>	The starting tick.
-------------	--------------------

10.60.2.48 print()

```
void seq64::midibase::print ( )
```

10.60.2.49 set_input()

```
bool seq64::midibase::set_input (
    bool inputing )
```

If the parameter is true, then [init_in\(\)](#) is called; otherwise, [deinit_in\(\)](#) is called.

Parameters

<i>inputing</i>	The inputing value to set. For input system ports, it is always set to true, no matter how it is configured in the "rc" file.
-----------------	---

10.60.2.50 display_name() [2/2]

```
void seq64::midibase::display_name (
    const std::string & name ) [inline], [protected]
```

10.60.2.51 bus_name() [2/2]

```
void seq64::midibase::bus_name (
    const std::string & name ) [inline], [protected]
```

10.60.2.52 port_name() [2/2]

```
void seq64::midibase::port_name (
    const std::string & name ) [inline], [protected]
```

10.60.2.53 set_port_id()

```
void seq64::midibase::set_port_id (
    int id ) [inline], [protected]
```

10.60.2.54 api_poll_for_midi()

```
virtual int seq64::midibase::api_poll_for_midi ( ) [inline], [protected], [virtual]
```

Also used in the JACK implementation.

Reimplemented in [seq64::midi_in_jack](#), [seq64::rtmidi](#), [seq64::midi_api](#), and [seq64::midibus](#).

10.60.2.55 api_get_midi_event()

```
virtual bool seq64::midibase::api_get_midi_event (
    event * inev ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi_in_jack](#), [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

10.60.2.56 api_init_in_sub()

```
virtual bool seq64::midibase::api_init_in_sub ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

10.60.2.57 api_init_out_sub()

```
virtual bool seq64::midibase::api_init_out_sub ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

10.60.2.58 api_deinit_in()

```
virtual bool seq64::midibase::api_deinit_in ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

10.60.2.59 api_play()

```
virtual void seq64::midibase::api_play (
    event * e24,
    midibyte channel ) [protected], [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

10.60.2.60 api_sysex()

```
virtual void seq64::midibase::api_sysex (
    event * ) [inline], [protected], [virtual]
```

The *e24* parameter, the SysEx event pointer, is unused here.

Reimplemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::rtmidi](#), and [seq64::midi_api](#).

10.60.2.61 `api_flush()`

```
virtual void seq64::midibase::api_flush ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::rtmidi](#), and [seq64::midi_api](#).

10.60.2.62 `api_init_in()`

```
virtual bool seq64::midibase::api_init_in ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

10.60.2.63 `api_init_out()`

```
virtual bool seq64::midibase::api_init_out ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

10.60.2.64 `api_continue_from()`

```
virtual void seq64::midibase::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

10.60.2.65 `api_start()`

```
virtual void seq64::midibase::api_start ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

10.60.2.66 `api_stop()`

```
virtual void seq64::midibase::api_stop ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

10.60.2.67 api_clock()

```
virtual void seq64::midibase::api_clock (
    midipulse tick ) [protected], [pure virtual]
```

Implemented in [seq64::midi_jack](#), [seq64::midi_alsa](#), [seq64::midi_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

10.60.3 Friends And Related Function Documentation

10.60.3.1 mastermidibus

```
friend class mastermidibus [friend]
```

10.60.4 Field Documentation

10.60.4.1 m_clock_mod

```
int seq64::midibase::m_clock_mod [static], [private]
```

Initialize this static member.

10.60.4.2 m_bus_index

```
const int seq64::midibase::m_bus_index [private]
```

Otherwise, it is currently -1.

10.60.4.3 m_bus_id

```
int seq64::midibase::m_bus_id [private]
```

For example, on one system the IDs are 14 (MIDI Through), 128 (TiMidity), and 129 (Yoshimi).

10.60.4.4 m_port_id

```
int seq64::midibase::m_port_id [private]
```

10.60.4.5 m_clock_type

```
clock_e seq64::midibase::m_clock_type [private]
```

The special value `e_clock_disabled` means we will not be using the port, so that a failure in setting up the port is not a "fatal error". (We could have added an "m_outputing" boolean as an alternative.)

10.60.4.6 m_inputing

```
bool seq64::midibase::m_inputing [private]
```

It is turned on if the user selects the port in the Options / MIDI Input tab.

10.60.4.7 m_ppqn

```
int seq64::midibase::m_ppqn [private]
```

Some APIs can control or use this value.

10.60.4.8 m_bpm

```
midibpm seq64::midibase::m_bpm [private]
```

Some APIs can control or use this value.

10.60.4.9 m_queue

```
int seq64::midibase::m_queue [private]
```

For ALSA, it is the ALSA queue number. For PortMidi, this is the old "m_pm_num" value. For RtMidi, it is not currently used.

10.60.4.10 m_display_name

```
std::string seq64::midibase::m_display_name [private]
```

Assembled by the [set_name\(\)](#) function.

10.60.4.11 m_bus_name

```
std::string seq64::midibase::m_bus_name [private]
```

This should be something like a major device name or the name of a subsystem such as Timidity.

10.60.4.12 m_port_name

```
std::string seq64::midibase::m_port_name [private]
```

This should be the name of a specific device or port on a major device.

10.60.4.13 m_lasttick

```
midipulse seq64::midibase::m_lasttick [private]
```

10.60.4.14 m_is_virtual_port

```
bool seq64::midibase::m_is_virtual_port [private]
```

The default is to create a system port (true).

10.60.4.15 m_is_input_port

```
bool seq64::midibase::m_is_input_port [private]
```

It matters when we are creating the name of the port, where we don't want an input virtual port to have the same name as an output virtual port... one of them will fail.

10.60.4.16 m_is_system_port

```
bool seq64::midibase::m_is_system_port [private]
```

Two examples are the ALSA System Timer buss and the ALSA System Announce bus, the latter being necessary for input subscription and notification. For most ports, this value will be false. A restricted setter is provided. Only the rtmidi ALSA implementation sets this flag.

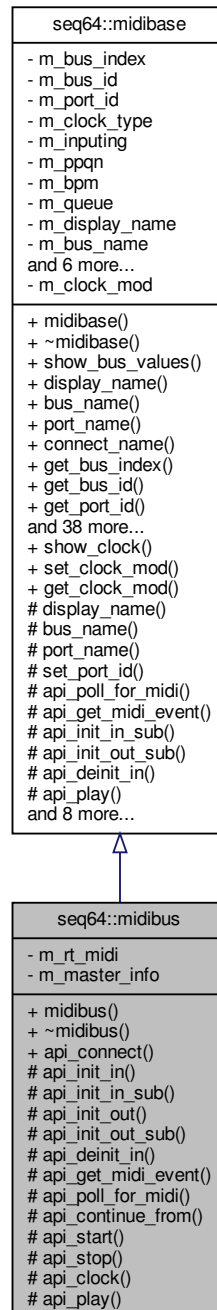
10.60.4.17 m_mutex

```
mutex seq64::midibase::m_mutex [private]
```

10.61 seq64::midibus Class Reference

This class implements with rtmidi version of the midibus object.

Inheritance diagram for seq64::midibus:



Public Member Functions

- [midibus](#) ([rtmidi_info](#) &rt, int index, bool makevirtual=SEQ64_MIDI_NORMAL_PORT, bool isinput=SEQ64_MIDI_OUTPUT_PORT, int busoverride=SEQ64_NO_BUS, bool makesystem=false)

- virtual `~midibus()`
- virtual bool `api_connect()`

Protected Member Functions

- virtual bool `api_init_in()`
- virtual bool `api_init_in_sub()`
- virtual bool `api_init_out()`
- virtual bool `api_init_out_sub()`
- virtual bool `api_deinit_in()`
- virtual bool `api_get_midi_event(event *inev)`
- virtual int `api_poll_for_midi()`
- virtual void `api_continue_from(midipulse tick, midipulse beats)`
- virtual void `api_start()`
- virtual void `api_stop()`
- virtual void `api_clock(midipulse tick)`
- virtual void `api_play(event *e24, midibyte channel)`

Private Attributes

- `rtmidi * m_rt_midi`
The RtMidi API interface object this midibus will be creating and then using.
- `rtmidi_info & m_master_info`
For Sequencer64, the ALSA model used requires that all the midibus objects use the same ASLA sequencer "handle".

Friends

- class `mastermidibus`
The master MIDI bus sets up the buss, so it gets access to private details.

Additional Inherited Members

10.61.1 Constructor & Destructor Documentation

10.61.1.1 midibus()

```
seq64::midibus::midibus (
    rtmidi_info & rt,
    int index,
    bool makevirtual = SEQ64_MIDI_NORMAL_PORT,
    bool isinput = SEQ64_MIDI_OUTPUT_PORT,
    int bussoverride = SEQ64_NO_BUS,
    bool makesystem = false )
```

10.61.1.2 ~midibus()

```
virtual seq64::midibus::~~midibus ( ) [virtual]
```

10.61.2 Member Function Documentation

10.61.2.1 api_connect()

```
virtual bool seq64::midibus::api_connect ( ) [virtual]
```

10.61.2.2 api_init_in()

```
virtual bool seq64::midibus::api_init_in ( ) [protected], [virtual]
```

Implements [seq64::midibase](#).

10.61.2.3 api_init_in_sub()

```
virtual bool seq64::midibus::api_init_in_sub ( ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

10.61.2.4 api_init_out()

```
virtual bool seq64::midibus::api_init_out ( ) [protected], [virtual]
```

Implements [seq64::midibase](#).

10.61.2.5 api_init_out_sub()

```
virtual bool seq64::midibus::api_init_out_sub ( ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

10.61.2.6 api_deinit_in()

```
virtual bool seq64::midibus::api_deinit_in ( ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

10.61.2.7 api_get_midi_event()

```
virtual bool seq64::midibus::api_get_midi_event (
    event * inev ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

10.61.2.8 api_poll_for_midi()

```
virtual int seq64::midibus::api_poll_for_midi ( ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

10.61.2.9 api_continue_from()

```
virtual void seq64::midibus::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [virtual]
```

Implements [seq64::midibase](#).

10.61.2.10 api_start()

```
virtual void seq64::midibus::api_start ( ) [protected], [virtual]
```

Implements [seq64::midibase](#).

10.61.2.11 api_stop()

```
virtual void seq64::midibus::api_stop ( ) [protected], [virtual]
```

Implements [seq64::midibase](#).

10.61.2.12 `api_clock()`

```
virtual void seq64::midibus::api_clock (
    midipulse tick ) [protected], [virtual]
```

Implements [seq64::midibase](#).

10.61.2.13 `api_play()`

```
virtual void seq64::midibus::api_play (
    event * e24,
    midibyte channel ) [protected], [virtual]
```

Implements [seq64::midibase](#).

10.61.3 Friends And Related Function Documentation

10.61.3.1 `mastermidibus`

```
friend class mastermidibus [friend]
```

10.61.4 Field Documentation

10.61.4.1 `m_rt_midi`

```
rtmidi* seq64::midibus::m_rt_midi [private]
```

10.61.4.2 `m_master_info`

```
rtmidi_info& seq64::midibus::m_master_info [private]
```

The [rtmidi_info](#) object used for enumerating the ports is a good place to get this handle. It is an extension of the legacy RtMidi interface.

10.62 `seq64::midifile` Class Reference

This class handles the parsing and writing of MIDI files.

Public Member Functions

- [midifile](#) (const std::string &name, int [ppqn](#)=SEQ64_USE_DEFAULT_PPQN, bool oldformat=false, bool glob-
alogs=true)
Principal constructor.
- virtual [~midifile](#) ()
A rote destructor.
- virtual bool [parse](#) ([perform](#) &p, int screenset=0, bool importing=false)
This function opens a binary MIDI file and parses it into sequences and other application objects.
- virtual bool [write](#) ([perform](#) &p, bool doseqspec=true)
Write the whole MIDI data and Seq24 information out to the file.
- bool [write_song](#) ([perform](#) &p)
- const std::string & [error_message](#) () const
'Getter' function for member m_error_message
- bool [error_is_fatal](#) () const
'Getter' function for member m_error_is_fatal
- int [ppqn](#) () const
*'Getter' function for member m_ppqn Provides a way to get the actual value of PPQN used in processing the se-
quences when [parse\(\)](#) was called.*
- size_t [get_file_pos](#) ()
'Getter' function for member m_pos

Protected Member Functions

- virtual [sequence](#) * [initialize_sequence](#) ([perform](#) &p)
- virtual void [finalize_sequence](#) ([perform](#) &p, [sequence](#) &seq, int seqnum, int screenset)
- void [clear_errors](#) ()
'Setter' function for member m_error_message
- void [ppqn](#) (int p)
'Setter' function for member m_ppqn
- bool [at_end](#) () const
Checks if the data stream pointer has reached the end position.
- bool [grab_input_stream](#) (const std::string &tag)
Creates the stream input, reads it into the "buffer", and then closes the file.
- bool [parse_smf_0](#) ([perform](#) &p, int screenset)
*This function parses an SMF 0 binary MIDI file as if it were an SMF 1 file, then, if more than one MIDI channel was
encountered in the sequence, splits all of the channels in the sequence out into separate sequences.*
- bool [parse_smf_1](#) ([perform](#) &p, int screenset, bool is_smf0=false)
This function parses an SMF 1 binary MIDI file; it is basically the original seq24 [midifile::parse\(\)](#) function.
- [midilong](#) [parse_prop_header](#) (int file_size)
Parse the proprietary header, figuring out if it is the new format, or the legacy format, for sequencer-specific data.
- bool [parse_proprietary_track](#) ([perform](#) &a_perf, int file_size)
*After all of the conventional MIDI tracks are read, we're now at the "proprietary" Seq24 data section, which describes
the various features that Seq24 supports.*
- bool [checklen](#) ([midilong](#) len, [midibyte](#) type)
Internal function to check for and report a bad length value.
- void [add_trigger](#) ([sequence](#) &seq, [midishort](#) ppqn)
Internal function to make the parser easier to read.
- bool [read_seek](#) (size_t pos)
Seeks to a new, absolute, position in the data stream.
- [midilong](#) [read_long](#) ()

- Reads 4 bytes of data using [read_byte\(\)](#).*

 - [midishort read_short](#) ()

Reads 2 bytes of data using [read_byte\(\)](#).

 - [midibyte read_byte](#) ()

Reads 1 byte of data directly from the `m_data` vector, incrementing `m_pos` after doing so.

 - [midilong read_varinum](#) ()

Read a MIDI Variable-Length Value (VLV), which has a variable number of bytes.

 - `bool read_byte_array (midibyte *b, size_t len)`

A helper function to simplify reading [midi_control](#) data from the MIDI file.

 - `bool read_byte_array (midistring &b, size_t len)`

A overload function to simplify reading [midi_control](#) data from the MIDI file.

 - `void read_gap (size_t sz)`

Jumps/skips the given number of bytes in the data stream.

 - `void write_long (midilong value)`

Writes 4 bytes, each extracted from the long value and shifted rightward down to byte size, using the [write_byte\(\)](#) function.

 - `void write_triple (midilong value)`

Writes 3 bytes, each extracted from the long value and shifted rightward down to byte size, using the [write_byte\(\)](#) function.

 - `void write_short (midishort value)`

Writes 2 bytes, each extracted from the long value and shifted rightward down to byte size, using the [write_byte\(\)](#) function.

 - `void write_byte (midibyte c)`

Writes 1 byte.

 - `void write_varinum (midilong)`

Writes a MIDI Variable-Length Value (VLV), which has a variable number of bytes.

 - `void write_track_name (const std::string &trackname)`

Writes out a track name.

 - `std::string read_track_name ()`

Reads the track name.

 - `void write_seq_number (midishort seqnum)`

Writes out a sequence number.

 - `int read_seq_number ()`

Reads the sequence number.

 - `void write_track_end ()`

Writes out the end-of-track marker.

 - `bool write_header (int numtracks)`

We want to write:

 - `void write_prop_header (midilong tag, long len)`

Writes a "proprietary" (SeqSpec) Seq24 footer header in either the new MIDI-compliant format, or the legacy Seq24 format.

 - `bool write_proprietary_track (perform &a_perf)`

Writes out the final proprietary/SeqSpec section, using the new format if the legacy format is not in force.

 - `long varinum_size (long len) const`

Calculates the length of a variable length value.

 - `long prop_item_size (long datalen) const`

Calculates the size of a proprietary item, as written by the [write_prop_header\(\)](#) function, plus whatever is called to write the data.

 - `long track_name_size (const std::string &trackname) const`

Calculates the size of a trackname and the meta event that specifies it.

 - `bool set_error (const std::string &msg)`

A new function that just sets the fatal-error status and the error message.

- bool `set_error_dump` (const std::string &msg)
Helper function to emit more useful error messages.
- bool `set_error_dump` (const std::string &msg, unsigned long p)
Helper function to emit more useful error messages for erroneous long values.
- void `write_track` (const `midi_vector` &lst)
- long `seq_number_size` () const
Returns the size of a sequence-number event, which is always 5 bytes, plus one byte for the delta time that precedes it.
- long `track_end_size` () const
Returns the size of a track-end event, which is always 3 bytes.
- bool `is_sysex_special_id` (midibyte ch)
Check for special SysEx ID byte.

Private Attributes

- `mutex m_mutex`
Provides locking for the sequence.
- `size_t m_file_size`
Holds the size of the MIDI file.
- `std::string m_error_message`
Holds the last error message, useful for trouble-shooting without having Sequencer64 running in a console window.
- bool `m_error_is_fatal`
Indicates if the error should be considered fatal.
- bool `m_disable_reported`
Indicates that file reading has already been disabled (due to serious errors), so don't complain about it anymore.
- `size_t m_pos`
Holds the position in the MIDI file.
- `const std::string m_name`
The unchanging name of the MIDI file.
- `std::vector< midibyte > m_data`
This vector of characters holds our MIDI data.
- `std::list< midibyte > m_char_list`
Provides a list of characters.
- bool `m_new_format`
Use the new format for the proprietary footer section of the Seq24 MIDI file.
- bool `m_global_bgsequence`
Indicates to store the new key, scale, and background sequence in the global, "proprietary" section of the MIDI song.
- int `m_ppqn`
Provides the current value of the PPQN, which used to be constant and is now only the macro SEQ64_DEFAULT_PPQN.
- bool `m_use_default_ppqn`
Indicates that the default PPQN is in force.
- `midi_splitter m_smf0_splitter`
Provides support for SMF 0.

10.62.1 Detailed Description

In addition to the standard MIDI tracks, it also handles some "private" or "proprietary" tracks specific to Seq24. It does not, however, handle SYSEX events.

10.62.2 Constructor & Destructor Documentation

10.62.2.1 midifile()

```
seq64::midifile::midifile (
    const std::string & name,
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    bool oldformat = false,
    bool globalbgs = true )
```

Parameters

<i>name</i>	Provides the name of the MIDI file to be read or written.
<i>ppqn</i>	<p>Provides the initial value of the PPQN setting. It is handled differently for parsing (reading) versus writing the MIDI file.</p> <ul style="list-style-type: none"> • Reading. <ul style="list-style-type: none"> – If set to SEQ64_USE_DEFAULT_PPQN, the legacy application behavior is used. The m_ppqn member is set to the default PPQN, DEFAULT_PPQN. The value read from the MIDI file, ppqn, is then use to scale the running-time of the sequence relative to DEFAULT_PPQN. – Otherwise, m_ppqn is set to the value read from the MIDI file. No scaling is done. Since the value gets written, specify ppqn as 0, an obviously bogus value, to get this behavior. • Writing. This value is written to the MIDI file in the header chunk of the song. Note that the caller must query for the PPQN set during parsing, and pass it to the constructor when preparing to write the file. See how it is done in the mainwnd class.
<i>oldformat</i>	If true, write out the MIDI file using the old Seq24 format, instead of the new MIDI-compliant sequencer-specific format, for the seq24-specific SeqSpec tags defined in the globals module. This option is false by default. Note that this option is only used in writing; reading can handle either format transparently.
<i>globalbgs</i>	If true, write any non-default values of the key, scale, and background sequence to the global "proprietary" section of the MIDI file, instead of to each sequence. Note that this option is only used in writing; reading can handle either format transparently.

10.62.2.2 ~midifile()

```
seq64::midifile::~~midifile ( ) [virtual]
```

10.62.3 Member Function Documentation

10.62.3.1 parse()

```
bool seq64::midifile::parse (
    perform & p,
    int screenset = 0,
    bool importing = false ) [virtual]
```

In addition to the standard MIDI track data in a normal track, Seq24/Sequencer64 adds four sequencer-specific events just before the end of the track:

```
c_triggers_new:    SeqSpec FF 7F 1C 24 24 00 08 00 00 ...
c_midibus:         SeqSpec FF 7F 05 24 24 00 01 00
c_timesig:         SeqSpec FF 7F 06 24 24 00 06 04 04
c_midich:          SeqSpec FF 7F 05 24 24 00 02 06
```

Note that only Sequencer64 adds "FF 7F len" to the SeqSpec data.

Standard MIDI provides for port and channel specification meta events, but they are apparently considered obsolete:

```
Obsolete meta-event:      Replacement:
MIDI port (buss):        FF 21 01 po    Device (port) name: FF 09 len text
MIDI channel:           FF 20 01 ch
```

What do other applications use for specifying port/channel?

Note the is-modified flag: We now assume that the perform object is starting from scratch when parsing. But we let mainwnd tell the perform object when to clear everything with `perform::clear_all()`. The mainwnd does this for a new file, opening a file, but not for a file import, which might be done simply to add more MIDI tracks to the current composition. So, if parsing succeeds, all we want to do is make sure the flag is set. Parsing a file successfully is not always a modification of the setup. For instance, the first read of a MIDI file should start clean, not dirty.

SysEx notes:

Some files (e.g. Dixie04.mid) do not always encode System Exclusive messages properly for a MIDI file. Instead of a varinum length value, they are followed by extended IDs (0x7D, 0x7E, or 0x7F).

We've covered some of those cases by disabling access to `m_data` if the position passes the size of the file, but we want try to bypass these odd cases properly. So we look ahead for one of these special values.

Currently, Sequencer64, like Se24, handles SysEx message only to the extend of passing them via MIDI Thru. We hope to improve on that capability eventually.

Parameters

<i>p</i>	Provides a reference to the perform object into which sequences/tracks are to be added.
<i>screenset</i>	The screen-set offset to be used when loading a sequence (track) from the file. This value ranges from -31 to 0 to +31 (32 is the maximum screen-set available in Seq24). This offset is added to the sequence number read in for the sequence, to place it elsewhere in the imported tune, and locate it in a specific screen-set. If this parameter is non-zero, then we will assume that the perform data is dirty.
<i>importing</i>	Indicates that we are importing a file, and do not want to parse/erase any "proprietary" information from the performance.

Returns

Returns true if the parsing succeeded. Note that the error status is saved in `m_error_is_fatal`, and a message (to display later) is saved in `m_error_message`.

10.62.3.2 write()

```
bool seq64::midifile::write (
    perform & p,
    bool doseqspec = true ) [virtual]
```

Also see the [write_song\(\)](#) function, for exporting to standard MIDI.

Sequencer64 sometimes reverses the order of some events, due to popping from its container. Not an issue, but can make a file slightly different for no reason.

Parameters

<i>p</i>	Provides the object that will contain and manage the entire performance.
<i>doseqspec</i>	If true (the default, then the Sequencer64-specific SeqSpec sections are written to the file. If false, we want to export the tracks as a basic MIDI sequence (which is not the same as exporting a Song, with triggers, as a MIDI sequence).

Returns

Returns true if the write operations succeeded. If false is returned, then `m_error_message` will contain a description of the error.

10.62.3.3 write_song()

```
bool seq64::midifile::write_song (
    perform & p )
```

10.62.3.4 error_message()

```
const std::string& seq64::midifile::error_message ( ) const [inline]
```

10.62.3.5 error_is_fatal()

```
bool seq64::midifile::error_is_fatal ( ) const [inline]
```

10.62.3.6 ppqn() [1/2]

```
int seq64::midifile::ppqn ( ) const [inline]
```

The PPQN will be either the global ppqn (legacy behavior) or the value read from the file, depending on the ppqn parameter passed to the midifile constructor.

10.62.3.7 get_file_pos()

```
size_t seq64::midifile::get_file_pos ( ) [inline]
```

Current position in the data stream.

10.62.3.8 initialize_sequence()

```
sequence * seq64::midifile::initialize_sequence (
    perform & p ) [protected], [virtual]
```

10.62.3.9 finalize_sequence()

```
void seq64::midifile::finalize_sequence (
    perform & p,
    sequence & seq,
    int seqnum,
    int screenset ) [protected], [virtual]
```

10.62.3.10 clear_errors()

```
void seq64::midifile::clear_errors ( ) [inline], [protected]
```

10.62.3.11 ppqn() [2/2]

```
void seq64::midifile::ppqn (
    int p ) [inline], [protected]
```

10.62.3.12 at_end()

```
bool seq64::midifile::at_end ( ) const [inline], [protected]
```

Returns

Returns true if the read pointer is at the end.

10.62.3.13 grab_input_stream()

```
bool seq64::midifile::grab_input_stream (
    const std::string & tag ) [protected]
```

No file buffering needed on these beefy machines! :-) As a side-effect, also sets m_file_size.

Parameters

<i>tag</i>	Basically an informative string to denote what kind of file is being opened, "MIDI" or "WRK".
------------	---

Returns

Returns true if the input stream was successfully opened on a good file. Use it only if the return value is true.

10.62.3.14 parse_smf_0()

```
bool seq64::midifile::parse_smf_0 (
    perform & p,
    int screenset ) [protected]
```

The original sequence remains in place, in sequence slot 16 (the 17th slot). The user is responsible for deleting it if it is not needed.

Parameters

<i>p</i>	Provides a reference to the perform object into which sequences/tracks are to be added.
<i>screenset</i>	The screen-set offset to be used when loading a sequence (track) from the file.

Returns

Returns true if the parsing succeeded.

10.62.3.15 parse_smf_1()

```
bool seq64::midifile::parse_smf_1 (
    perform & p,
    int screenset,
    bool is_smf0 = false ) [protected]
```

It assumes the file-data has already been read into memory. It also assumes that the ID, track-length, and format have already been read.

If the MIDI file contains both proprietary (c_timesig) and MIDI type 0x58 then it came from seq42 or seq32 (Stazed versions). In this case the MIDI type is parsed first (because it is listed first) then it gets overwritten by the proprietary, above.

Note that NumTracks doesn't count the Seq24 "proprietary" footer section, even if it uses the new format, so that section will still be read properly after all normal tracks have been processed.

PPQN:

Current time (RunningTime) is re the ppqn according to the file, we have to adjust it to our own ppqn. PPQN / ppqn gives us the ratio. (This change is not enough; a song with a ppqn of 120 plays too fast in Seq24, which has a constant ppqn of 192. Triggers must also be modified.)

Tempo events:

If valid, set the global tempo to the first encountered tempo; this is legacy behavior. Bad tempos occur and stick around, munging exported songs. We log only the first tempo officially; the rest are stored as events if in the first track. We also adjust the upper draw-tempo range value to about twice this value, to give some headroom... it will not be saved unless the --user-save option is in force.

Time Signature:

Like Tempo, Time signature is now handled more robustly.

Key Signature and other Meta events:

Although we don't support these events, we do want to keep them, so we can output them upon saving. Instead of bypassing unhandled Meta events, we now store them, so that they are not lost when exporting/saving the MIDI data.

Track name:

This event is optional. It's interpretation depends on its context. If it occurs in the first track of a format 0 or 1 MIDI file, then it gives the Sequence Name. Otherwise it gives the Track Name.

End of Track:

"If Delta is 0, then another event happened at the same time as track-end. Class sequence discards the last note. This fixes that. A native Seq24 file will always have a Delta >= 1." Not true! We've fixed the real issue by commenting the code that increments current time. Question: What if BPM is set *after* this event?

Sequences:

If the sequence is shorter than a quarter note, assume it needs to be padded to a measure. This happens anyway if the short pattern is opened in the sequence editor (seqedit).

Add sorting after reading all the events for the sequence. Then add the sequence with it's preferred location as a hint.

Unknown chunks:

Let's say we don't know what kind of chunk it is. It's not a MTrk, we don't know how to deal with it, so we just eat it. If this happened on the first track, it is a fatal error.

Parameters

<i>p</i>	Provides a reference to the perform object into which sequences/tracks are to be added.
<i>screenset</i>	The screen-set offset to be used when loading a sequence (track) from the file.
<i>is_smf0</i>	True if we detected that the MIDI file is in SMF 0 format.

Returns

Returns true if the parsing succeeded.

10.62.3.16 parse_prop_header()

```
midilong seq64::midifile::parse_prop_header (
    int file_size ) [protected]
```

The new format creates a final track chunk, starting with "MTrk". Then comes the delta-time (here, 0), and the event. An event is a MIDI event, a SysEx event, or a Meta event.

A MIDI Sequencer Specific meta message includes either a delta time or absolute time, and the MIDI Sequencer Specific event encoded as follows:

```
0x00 0xFF 0x7F length data
```

For convenience, this function first checks the amount of file data left. If enough, then it reads a long value. If the value starts with 0x00 0xFF 0x7F, then that is a SeqSpec event, which signals usage of the new Sequencer64 "proprietary" format. Otherwise, it is probably the old format, and the long value is a control tag (0x24240nn), which can be returned immediately.

If it is the new format, we back up to the FF, then get the next byte, which should be a 7F. If so, then we read the length (a variable length value) of the data, and then read the long value, which should be the control tag, which, again, is returned by this function.

Note

Most sequencers seem to be tolerant of both the lack of an "MTrk" marker and of the presence of an unwrapped control tag, and so can handle both the old and new formats of the final proprietary track.

Parameters

<i>file_size</i>	The size of the data file. This value is compared against the member <i>m_pos</i> (the position inside <i>m_data[]</i>), to make sure there is enough data left to process.
------------------	--

Returns

Returns the control-tag value found. These are the values, such as *c_midich*, found in the *globals* module, that indicate the type of sequencer-specific data that comes next. If there is not enough data to process, then 0 is returned.

10.62.3.17 parse_proprietary_track()

```
bool seq64::midifile::parse_proprietary_track (
    perform & p,
    int file_size ) [protected]
```

It consists of series of tags:

```
c_midictrl
c_midiclocks
c_notes
c_bpmtag (beats per minute)
c_mutegroups
c_musickey (new, added if usr() global_seq_feature() is true)
c_musicscale (ditto)
c_backsequence (ditto)
```

(There are more tags defined in the globals module, but they are not used in this function. This doesn't quite make sense, as there are also some "triggers" values, and we're pretty sure the application uses them. Oh, it turns out that they are set up by actions performed on each sequence, and are stored as sequencer-specific ("SeqSpec") data with each track's data as held in the MIDI container for the track. See the midi_container module for more information.)

The format is (1) tag ID; (2) length of data; (3) the data.

First, we separate out this function for a little more clarity. Then we added code to handle reading both the legacy Seq24 format and the new, MIDI-compliant format. Note that even the new format is not quite correct, since it doesn't handle a MIDI manufacturer's ID, making it a single byte that is part of the data. But it does have the "MTrk" marker and track name, so that must be processed for the new format.

Now, in our "midicvt" project, we have a test MIDI file, b4uacuse-non-mtrk.midi that is good, except for having a tag "MUnk" instead of "MTrk". We should consider being more permissive, if possible. Otherwise, though, the only penalty is that the "proprietary" chunk is completely skipped.

Extra precision BPM:

Based on a request for two decimals of precision in beats-per-minute, we now save a scaled version of BPM. Our supported range of BPM is SEQ64_MINIMUM_BPM = 1 to SEQ64_MAXIMUM_BPM = 600. If this range is encountered, the value is read as is. If greater than this range (actually, we use 999 as the limit), then we divide the number by 1000 to get the actual BPM, which can thus have more precision than the old integer value allowed. Obviously, when saving, we will multiply by 1000 to encode the BPM.

Parameters

<i>p</i>	The performance object that is being set via the incoming MIDI file.
<i>file_size</i>	The file size as determined in the parse() function.

There are also implicit parameters, with the m_pos and m_new_format member variables.

10.62.3.18 checklen()

```
bool seq64::midifile::checklen (
    midilong len,
    midibyte type ) [protected]
```

A length of zero is now considered legal, but a "warning" message is shown. The largest value allowed within a MIDI file is 0x0FFFFFFF. This limit is set to allow variable-length quantities to be manipulated as 32-bit integers.

Parameters

<i>len</i>	The length value to be checked, and it should be greater than 0. However, we have seen files with zero-length events, such as Lyric events (0x05).
<i>type</i>	The type of meta event. Used for displaying an error.

Returns

Returns true if the length parameter is valid. This now means it is simply less than 0x0FFFFFFF.

10.62.3.19 add_trigger()

```
void seq64::midifile::add_trigger (
    sequence & seq,
    midishort ppqn ) [protected]
```

Handles only `c_triggers_new` values, not the old `c_triggers` value. If `m_ppqn` isn't set to the default value, then we must scale these triggers accordingly, just as is done for the MIDI events.

Parameters

<i>seq</i>	Provides the sequence to which the trigger is to be added.
<i>ppqn</i>	Provides the ppqn value to use to scale the tick values if <code>m_use_default_ppqn</code> is true. If 0, the ppqn value is not used.

10.62.3.20 read_seek()

```
bool seq64::midifile::read_seek (
    size_t pos ) [protected]
```

All this function does is change the value of `m_pos`. All of the file is already in memory.

Parameters

<i>pos</i>	Provides the new position to seek.
------------	------------------------------------

Returns

Returns true if the seek could be accomplished. No error message is logged, but the caller should take evasive action if false is returned. And, in that case, m_pos is unchanged.

10.62.3.21 read_long()

```
midilong seq64::midifile::read_long ( ) [protected]
```

Warning

This code looks endian-dependent and integer-size dependent.

Returns

Returns the four bytes, shifted appropriately and added together, most-significant byte first, to sum to a long value.

10.62.3.22 read_short()

```
midishort seq64::midifile::read_short ( ) [protected]
```

Returns

Returns the two bytes, shifted appropriately and added together, most-significant byte first, to sum to a short value.

10.62.3.23 read_byte()

```
midibyte seq64::midifile::read_byte ( ) [protected]
```

Returns

Returns the byte that was read. Returns 0 if there was an error, though there's no way for the caller to determine if this is an error or a good value.

10.62.3.24 read_varinum()

```
midilong seq64::midifile::read_varinum ( ) [protected]
```

This function reads the bytes while bit 7 is set in each byte. Bit 7 is a continuation bit. See [write_varinum\(\)](#) for more information.

Returns

Returns the accumulated values as a single number.

10.62.3.25 read_byte_array() [1/2]

```
bool seq64::midifile::read_byte_array (
    midibyte * b,
    size_t len ) [protected]
```

Parameters

<i>b</i>	The byte array to receive the data.
<i>len</i>	The number of bytes in the array, and to be read.

Returns

Returns true if any bytes were read. Do not use *b* if false is returned.

10.62.3.26 read_byte_array() [2/2]

```
bool seq64::midifile::read_byte_array (
    midistring & b,
    size_t len ) [protected]
```

It uses a midistring object instead of a buffer.

Parameters

<i>b</i>	The midistring to receive the data.
<i>len</i>	The number of bytes to be read.

Returns

Returns true if any bytes were read. The string *b* will be empty if false is returned.

10.62.3.27 read_gap()

```
void seq64::midifile::read_gap (
    size_t sz ) [protected]
```

If too large, the position is left at the end.

Parameters

<i>sz</i>	Provides the gap size, in bytes.
-----------	----------------------------------

10.62.3.28 write_long()

```
void seq64::midifile::write_long (
    midilong x ) [protected]
```

Warning

This code looks endian-dependent.

Parameters

x	The long value to be written to the MIDI file.
---	--

10.62.3.29 write_triple()

```
void seq64::midifile::write_triple (
    midilong x ) [protected]
```

This function is kind of the reverse of [tempo_us_to_bytes\(\)](#) defined in the calculations.cpp module.

Warning

This code looks endian-dependent.

Parameters

x	The long value to be written to the MIDI file.
---	--

10.62.3.30 write_short()

```
void seq64::midifile::write_short (
    midishort x ) [protected]
```

Warning

This code looks endian-dependent.

Parameters

x	The short value to be written to the MIDI file.
---	---

10.62.3.31 write_byte()

```
void seq64::midifile::write_byte (
    midibyte c ) [inline], [protected]
```

The byte is written to the m_char_list member, using a call to push_back().

Parameters

<code>c</code>	The MIDI byte to be "written".
----------------	--------------------------------

10.62.3.32 `write_varinum()`

```
void seq64::midifile::write_varinum (
    midilong value ) [protected]
```

A MIDI file Variable Length Value is stored in bytes. Each byte has two parts: 7 bits of data and 1 continuation bit. The highest-order bit is set to 1 if there is another byte of the number to follow. The highest-order bit is set to 0 if this byte is the last byte in the VLV.

To recreate a number represented by a VLV, first you remove the continuation bit and then concatenate the leftover bits into a single number.

To generate a VLV from a given number, break the number up into 7 bit units and then apply the correct continuation bit to each byte.

In theory, you could have a very long VLV number which was quite large; however, in the standard MIDI file specification, the maximum length of a VLV value is 5 bytes, and the number it represents can not be larger than 4 bytes.

Here are some common cases:

- Numbers between 0 and 127 are represented by a single byte: 0x00 to 7F.
- 0x80 is represented as 0x81 00.
- The largest 2-byte MIDI value (e.g. a sequence number) is 0xFF 7F, which is $127 * 128 + 127 = 16383 = 0x3FFF$.
- The largest 3-byte MIDI value is 0xFF FF 7F = 0x1FFFFFF.
- The largest number, 4 bytes, is 0xFF FF FF 7F = 0xFFFFFFFF.

Also see the [varinum_size\(\)](#) function.

Parameters

<code>value</code>	The long value to be encoded as a MIDI varinum, and written to the MIDI file.
--------------------	---

10.62.3.33 `write_track_name()`

```
void seq64::midifile::write_track_name (
    const std::string & trackname ) [protected]
```

Note that we have to precede this "event" with a delta time value, set to 0. The format of the output is "0x00 0xFF 0x03 len track-name-bytes".

Parameters

<i>trackname</i>	Provides the name of the track to be written to the MIDI file.
------------------	--

10.62.3.34 read_track_name()

```
std::string seq64::midifile::read_track_name ( ) [protected]
```

Meant only for usage in the proprietary/SeqSpec footer track, in the new file format.

Returns

Returns the track name, or an empty string if there was a problem.

10.62.3.35 write_seq_number()

```
void seq64::midifile::write_seq_number (
    midishort seqnum ) [protected]
```

The format is "00 FF 00 02 ss ss", where "02" is actually the constant length of the data. We have to precede these values with a 0 delta time, of course.

Now, for sequence 0, an alternate format is "FF 00 00". But that format can only occur in the first track, and the rest of the tracks then don't need a sequence number, since it is assumed to increment. Our application doesn't bother with that shortcut.

Parameters

<i>seqnum</i>	The sequence number to write.
---------------	-------------------------------

10.62.3.36 read_seq_number()

```
int seq64::midifile::read_seq_number ( ) [protected]
```

Meant only for usage in the proprietary/SeqSpec footer track, in the new file format.

Returns

Returns the sequence number found, or -1 if it was not found.

10.62.3.37 write_track_end()

```
void seq64::midifile::write_track_end ( ) [protected]
```

10.62.3.38 write_header()

```
bool seq64::midifile::write_header (
    int numtracks ) [protected]
```

- 0x4D54726B. The track tag "MTrk". The MIDI spec requires that software can skip over non-standard chunks. "Prop"? Would require a fix to midicvt.
- 0xaabbccdd. The length of the track. This needs to be calculated somehow.
- 0x00. A zero delta time.
- 0x7f7f. Sequence number, a special value, well out of normal range.
- The name of the track:
 - "Seq24-Spec"
 - "Sequencer64-S"

Then follows the proprietary/SeqSpec data, written in the normal manner. Finally, tack on the track-end meta-event.

Components of final track size:

```
-# Delta time. 1 byte, always 0x00.
-# Sequence number. 5 bytes. OPTIONAL. We won't write it.
-# Track name. 3 + 10 or 3 + 15
-# Series of proprietary/SeqSpec specs:
-# Prop header:
-#   If legacy format, 4 bytes.
-#   Otherwise, 2 bytes + varinum_size(length) + 4 bytes.
-#   Length of the prop data.
-# Track End. 3 bytes.
```

10.62.3.39 write_prop_header()

```
void seq64::midifile::write_prop_header (
    midilong control_tag,
    long data_length ) [protected]
```

This function does not write the data. It replaces calls such as "write_long(c_midich)" in the proprietary section of [write\(\)](#).

The legacy format just writes the control tag (0x242400xx). The new format writes 0x00 0xFF 0x7F len 0x242400xx; the first 0x00 is the delta time.

In the new format, the 0x24 is a kind of "manufacturer ID". At <http://www.midi.org/techspecs/manid.php> we see that most manufacturer IDs start with 0x00, and are thus three bytes long, or start with codes at 0x40 and above. Similary, this site shows that no manufacturer uses 0x24:

<http://sequence15.blogspot.com/2008/12/midi-manufacturer-ids.html>

Warning

Currently, the manufacturer ID is not handled; it is part of the data, which can be misleading in programs that analyze MIDI files.

Parameters

<i>control_tag</i>	Determines the type of sequencer-specific section to be written. It should be one of the value in the globals module, such as <code>c_midibus</code> or <code>c_mutegroups</code> .
<i>data_length</i>	The amount of data that will be written. This parameter does not count the length of the header itself.

10.62.3.40 `write_proprietary_track()`

```
bool seq64::midifile::write_proprietary_track (
    perform & p ) [protected]
```

The first thing to do, for the new format only, is calculate the length of this big section of data. This was quite tricky; we tweaked and adjusted until the midicvt program handled the whole new-format file without emitting any errors.

Here's the basics of what Seq24 did for writing the data in this part of the file:

```
-# Write the c_midictrl value, then write a 0. To us, this looks like
no one wrote any code to write this data. And yet, the parsing
code can handles a non-zero value, which is the number of sequences
as a long value, not a byte. So shouldn't we write 4 bytes, not
one? Yes, indeed, we made a mistake. However, we should be
writing out the full data set as well. But not even Seq24 does
that! Perhaps they decided it was best kept in the "rc"
configuration file.
-# MORE TO COME.
```

We need a way to make the group mute data optional. Why write 4096 bytes of zeroes?

Parameters

<i>p</i>	Provides the object that will contain and manage the entire performance.
----------	--

Returns

Always returns true. No efficient way to check all of the writes that can happen. Might revisit this issue if some bug crops up.

10.62.3.41 `varinum_size()`

```
long seq64::midifile::varinum_size (
    long len ) const [protected]
```

This function is needed when calculating the length of a track. Note that it handles only the following situations:

https://en.wikipedia.org/wiki/Variable-length_quantity

This restriction allows the calculation to be simple and fast.

```
1 byte:  0x00 to 0x7F
2 bytes: 0x80 to 0x3FFF
3 bytes: 0x4000 to 0x001FFFFF
4 bytes: 0x200000 to 0x0FFFFFFF
```

Parameters

<i>len</i>	The long value whose length, when encoded as a MIDI varinum, is to be found.
------------	--

Returns

Returns values as noted above. Anything beyond that range returns 0.

10.62.3.42 prop_item_size()

```
long seq64::midifile::prop_item_size (
    long data_length ) const [protected]
```

If using the new format, the length includes the sum of sequencer-specific tag (0xFF 0x7F) and the size of the variable-length value. Then, for legacy and new format, 4 bytes are added for the Seq24 MIDI control value, and then the data length is added.

Parameters

<i>data_length</i>	Provides the data length value to be encoded.
--------------------	---

Returns

Returns the length of the item size, including the delta time, meta bytes, length bytes, the control tag, and the data-length itself.

10.62.3.43 track_name_size()

```
long seq64::midifile::track_name_size (
    const std::string & trackname ) const [protected]
```

Parameters

<i>trackname</i>	Provides the name of the track to be written to the MIDI file.
------------------	--

Returns

Returns the length of the event, which is of the format "0x00 0xFF 0x03 len track-name-bytes".

10.62.3.44 set_error()

```
bool seq64::midifile::set_error (
    const std::string & msg ) [protected]
```

Parameters

<i>msg</i>	Provides the error message.
------------	-----------------------------

Returns

Returns false, so that the caller can just assign this as the erroneous boolean function result.

10.62.3.45 set_error_dump() [1/2]

```
bool seq64::midifile::set_error_dump (
    const std::string & msg ) [protected]
```

It adds the file offset to the message.

Parameters

<i>msg</i>	The main error message string, without an ending newline character.
------------	---

Returns

Always returns false, to make it easier on the caller. The constructed string is returned as a side-effect, in case we want to pass it along to the externally-accessible error-message buffer.

10.62.3.46 set_error_dump() [2/2]

```
bool seq64::midifile::set_error_dump (
    const std::string & msg,
    unsigned long value ) [protected]
```

It adds the file offset to the message.

Parameters

<i>msg</i>	The main error message string, without an ending newline character.
<i>value</i>	The long value to show as part of the message.

Returns

Always returns false, to make it easier on the caller. The constructed string is returned as a side-effect, in case we want to pass it along to the externally-accessible error-message buffer.

10.62.3.47 write_track()

```
void seq64::midifile::write_track (
    const midi\_vector & lst ) [protected]
```

10.62.3.48 seq_number_size()

```
long seq64::midifile::seq_number_size ( ) const [inline], [protected]
```

10.62.3.49 track_end_size()

```
long seq64::midifile::track_end_size ( ) const [inline], [protected]
```

10.62.3.50 is_sysex_special_id()

```
bool seq64::midifile::is_sysex_special_id (
    midibyte ch ) [inline], [protected]
```

Parameters

<i>ch</i>	Provides the byte to be checked against 0x7D through 0x7F.
-----------	--

Returns

Returns true if the byte is SysEx special ID.

10.62.4 Field Documentation**10.62.4.1 m_mutex**

```
mutex seq64::midifile::m_mutex [mutable], [private]
```

Made mutable for use in certain locked getter functions.

10.62.4.2 m_file_size

```
size_t seq64::midifile::m_file_size [private]
```

This variable was added when loading a file that caused an attempt to load data well beyond the file-size of the midicvt test file Dixie04.mid.

10.62.4.3 m_error_message

```
std::string seq64::midifile::m_error_message [private]
```

If empty, there's no pending error. Currently most useful in the [parse\(\)](#) function.

10.62.4.4 m_error_is_fatal

```
bool seq64::midifile::m_error_is_fatal [private]
```

The caller can query for this value after getting the return value from [parse\(\)](#).

10.62.4.5 m_disable_reported

```
bool seq64::midifile::m_disable_reported [private]
```

Once is enough.

10.62.4.6 m_pos

```
size_t seq64::midifile::m_pos [private]
```

This is at least a 31-bit value in the recent architectures running Linux and Windows, so it will handle up to 2 Gb of data. This member is used as the offset into the `m_data` vector.

10.62.4.7 m_name

```
const std::string seq64::midifile::m_name [private]
```

10.62.4.8 m_data

```
std::vector<midibyte> seq64::midifile::m_data [private]
```

We could also use a string of characters, unsigned. This member is resized to the putative size of the MIDI file, in the [parse\(\)](#) function. Then the whole file is read into it, as if it were an array. This member is an input buffer.

10.62.4.9 m_char_list

```
std::list<midibyte> seq64::midifile::m_char_list [private]
```

The class pushes each MIDI byte into this list using the [write_byte\(\)](#) function. Also note that the [write\(\)](#) function calls `sequence::fill_list()` to fill a temporary `std::list<char>` (!) buffer, then writes that data *backwards* to this member. This member is an output buffer.

10.62.4.10 m_new_format

```
bool seq64::midifile::m_new_format [private]
```

In the new format, each sequencer-specific value (0x242400xx, as defined in the globals module) is preceded by the sequencer-specific prefix, 0xFF 0x7F len id/date). By default, the new format is used, but the user can specify the `-legacy` (-l) option, or make a soft link to the sequence24 binary called "seq24", to write the data in the old format. [We will eventually add the `-legacy` option to the "rc" configuration file.] Note that reading can handle either format transparently.

10.62.4.11 m_global_bgsequence

```
bool seq64::midifile::m_global_bgsequence [private]
```

10.62.4.12 m_ppqn

```
int seq64::midifile::m_ppqn [private]
```

10.62.4.13 m_use_default_ppqn

```
bool seq64::midifile::m_use_default_ppqn [private]
```

10.62.4.14 m_smf0_splitter

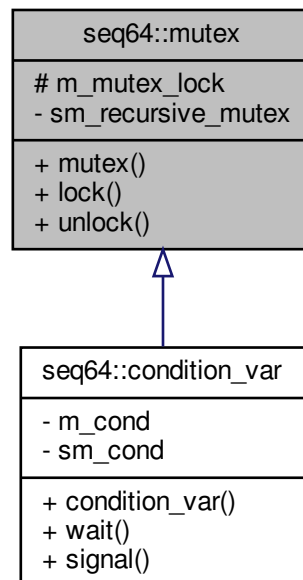
```
midi_splitter seq64::midifile::m_smf0_splitter [private]
```

This object holds all of the information needed to split a multi-channel sequence.

10.63 seq64::mutex Class Reference

The mutex class provides a simple wrapper for the pthread_mutex_t type used as a recursive mutex.

Inheritance diagram for seq64::mutex:



Public Member Functions

- `mutex ()`
The constructor assigns the recursive mutex to the local locking mutex.
- `void lock () const`
Lock the mutex.
- `void unlock () const`
Unlock the mutex.

Protected Attributes

- `pthread_mutex_t m_mutex_lock`
Provides a mutex lock usable by a single module or class.

Static Private Attributes

- `static const pthread_mutex_t sm_recursive_mutex`
Provides a recursive mutex that can be used by the whole application, and is, apparently.

10.63.1 Constructor & Destructor Documentation

10.63.1.1 mutex()

```
seq64::mutex::mutex ( )
```

10.63.2 Member Function Documentation

10.63.2.1 lock()

```
void seq64::mutex::lock ( ) const
```

10.63.2.2 unlock()

```
void seq64::mutex::unlock ( ) const
```

10.63.3 Field Documentation

10.63.3.1 sm_recursive_mutex

```
const pthread_mutex_t seq64::mutex::sm_recursive_mutex [static], [private]
```

Define the static recursive mutex and its condition variable.

10.63.3.2 m_mutex_lock

```
pthread_mutex_t seq64::mutex::m_mutex_lock [mutable], [protected]
```

However, this mutex ends up being a copy of the static sm_recursive_mutex (and, of course, a different "object").

10.64 seq64::editable_event::name_value_t Struct Reference

Provides a type that contains the pair of values needed for the various lookup maps that are needed to manage editable events.

Data Fields

- unsigned short [event_value](#)
Holds a midibyte value (0x00 to 0xFF) or SEQ64_END_OF_MIDIBYTE_TABLE to indicate the end of an array of [name_value_t](#) items.
- std::string [event_name](#)
Holds the human-readable name for an event code or other numeric value in an array of [name_value_t](#) items.

10.64.1 Field Documentation

10.64.1.1 event_value

```
unsigned short seq64::editable_event::name_value_t::event_value
```

This field can be considered a "key" value, as it is often looked up to find the event name.

10.64.1.2 event_name

```
std::string seq64::editable_event::name_value_t::event_name
```

10.65 seq64::options Class Reference

This class supports a full tabbed options dialog.

Inherits Dialog.

Public Member Functions

- [options](#) (Gtk::Window &parent, [perform](#) &p, bool showjack=false)

Private Types

- enum [button](#) {
[e_jack_transport](#),
[e_jack_master](#),
[e_jack_master_cond](#),
[e_jack_midi](#),
[e_jack_start_mode_live](#),
[e_jack_start_mode_song](#),
[e_jack_connect](#),
[e_jack_disconnect](#) }

Defines buttons indices or IDs for some controls related to JACK.

Private Member Functions

- [perform](#) & [perf](#) ()
 - 'Getter' function for member m_mainperf*
- void [clock_callback_off](#) (int bus, Gtk::RadioButton *[button](#))
- void [clock_callback_on](#) (int bus, Gtk::RadioButton *[button](#))
- void [clock_callback_mod](#) (int bus, Gtk::RadioButton *[button](#))
- void [clock_callback_disable](#) (int bus, Gtk::RadioButton *[button](#))
- void [clock_mod_callback](#) (Gtk::Adjustment *adj)
- void [edit_tempo_track_number](#) (Gtk::Entry *text)
- void [log_tempo_track_number](#) ()
- void [input_callback](#) (int bus, Gtk::Button *[button](#))
- void [filter_callback](#) (Gtk::Button *[button](#))
- void [transport_callback](#) ([button](#) type, Gtk::Button *[button](#))
- void [mouse_seq24_callback](#) (Gtk::RadioButton *)
- void [mouse_fruity_callback](#) (Gtk::RadioButton *)
- void [mouse_mod4_callback](#) (Gtk::CheckButton *)
- void [mouse_snap_split_callback](#) (Gtk::CheckButton *)
- void [mouse_click_edit_callback](#) (Gtk::CheckButton *)
- void [lash_support_callback](#) (Gtk::CheckButton *)
- void [add_midi_clock_page](#) ()
- void [add_midi_input_page](#) ()
- void [add_keyboard_page](#) ()
- void [add_extended_keys_page](#) ()
- void [add_mouse_page](#) ()
- void [add_jack_sync_page](#) ()

Private Attributes

- Gtk::Tooltips * [m_tooltips](#)
 - A repository for GTK tooltip support.*
- [perform](#) & [m_mainperf](#)
 - The performance object to which some of these options apply.*
- Gtk::Button * [m_button_ok](#)
 - The famous "OK" button's pointer.*
- Gtk::CheckButton * [m_button_jack_transport](#)
 - Main JACK transport selection.*
- Gtk::CheckButton * [m_button_jack_master](#)
 - Main JACK transport master selection.*
- Gtk::CheckButton * [m_button_jack_master_cond](#)
 - Main JACK transport master-conditional selection.*
- Gtk::Button * [m_button_jack_connect](#)
 - JACK Connect button, which we need to enable/disable for clarity and some additional safety.*
- Gtk::Button * [m_button_jack_disconnect](#)
 - JACK Disconnect button, which we need to enable/disable for clarity and some additional safety.*
- Gtk::Notebook * [m_notebook](#)
 - Not sure yet what this notebook is for.*

10.65.1 Member Enumeration Documentation

10.65.1.1 button

```
enum seq64::options::button [private]
```

These values are handled in `options::transport_callback()`. Some of them set JACK-related values in the `rc_settings` object, while the others set up or tear down the JACK support of sequencer64.

The JACK Transport settings are a little messy. They should be radio buttons, and control each other's settings. Currently, if the user wants to set up for JACK Master, the JACK Transport button must also be checked.

Enumerator

e_jack_transport	Turns on the "with JACK Transport" option, <code>rc_settings :: with_jack_transport()</code> .
e_jack_master	Turns on the "with JACK Master" option, <code>rc_settings :: with_jack_master()</code> . If another application is already JACK Master, this will fail.
e_jack_master_cond	Turns on the "with JACK Master" option <code>rc_settings :: with_jack_master_cond()</code> . This option makes sequencer64 the JACK Master conditionally, that is, if no other application has claimed that role.
e_jack_midi	Turns on the "Native JACK MIDI" option <code>rc_settings :: with_jack_midi()</code> . This is a setting independent of the JACK Transport settings. This is use only in the "rtmidi" implementation os Sequencer64, <code>seq64</code> .
e_jack_start_mode_live	Doesn't directly do anything; the live mode versus song mode is set by the <code>e_jack_start_mode_song</code> value.
e_jack_start_mode_song	Sets the "JACK start mode" value to true, which means that sequencer64 is in song mode. This value is obtained via <code>rc_settings :: song_start_mode()</code> . It will eventually be the start mode that applies to either ALSA or JACK playback.
e_jack_connect	Causes the perform object's JACK initialization function, <code>perform::init_jack()</code> , to be called.
e_jack_disconnect	Causes the perform object's JACK deinitialization function, <code>perform::deinit_jack()</code> , to be called.

10.65.2 Constructor & Destructor Documentation

10.65.2.1 options()

```
seq64::options::options (
    Gtk::Window & parent,
    perform & p,
    bool showjack = false )
```

10.65.3 Member Function Documentation

10.65.3.1 perf()

```
perform& seq64::options::perf ( ) [inline], [private]
```

10.65.3.2 clock_callback_off()

```
void seq64::options::clock_callback_off (
    int bus,
    Gtk::RadioButton * button ) [private]
```

10.65.3.3 clock_callback_on()

```
void seq64::options::clock_callback_on (
    int bus,
    Gtk::RadioButton * button ) [private]
```

10.65.3.4 clock_callback_mod()

```
void seq64::options::clock_callback_mod (
    int bus,
    Gtk::RadioButton * button ) [private]
```

10.65.3.5 clock_callback_disable()

```
void seq64::options::clock_callback_disable (
    int bus,
    Gtk::RadioButton * button ) [private]
```

10.65.3.6 clock_mod_callback()

```
void seq64::options::clock_mod_callback (
    Gtk::Adjustment * adj ) [private]
```

10.65.3.7 edit_tempo_track_number()

```
void seq64::options::edit_tempo_track_number (
    Gtk::Entry * text ) [private]
```

10.65.3.8 log_tempo_track_number()

```
void seq64::options::log_tempo_track_number ( ) [private]
```

10.65.3.9 input_callback()

```
void seq64::options::input_callback (
    int bus,
    Gtk::Button * button ) [private]
```

10.65.3.10 filter_callback()

```
void seq64::options::filter_callback (
    Gtk::Button * button ) [private]
```

10.65.3.11 transport_callback()

```
void seq64::options::transport_callback (
    button type,
    Gtk::Button * button ) [private]
```

10.65.3.12 mouse_seq24_callback()

```
void seq64::options::mouse_seq24_callback (
    Gtk::RadioButton * ) [private]
```

10.65.3.13 mouse_fruity_callback()

```
void seq64::options::mouse_fruity_callback (
    Gtk::RadioButton * ) [private]
```

10.65.3.14 mouse_mod4_callback()

```
void seq64::options::mouse_mod4_callback (
    Gtk::CheckButton * ) [private]
```

10.65.3.15 mouse_snap_split_callback()

```
void seq64::options::mouse_snap_split_callback (
    Gtk::CheckButton * ) [private]
```

10.65.3.16 mouse_click_edit_callback()

```
void seq64::options::mouse_click_edit_callback (
    Gtk::CheckButton * ) [private]
```

10.65.3.17 lash_support_callback()

```
void seq64::options::lash_support_callback (
    Gtk::CheckButton * ) [private]
```

10.65.3.18 add_midi_clock_page()

```
void seq64::options::add_midi_clock_page ( ) [private]
```

10.65.3.19 add_midi_input_page()

```
void seq64::options::add_midi_input_page ( ) [private]
```

10.65.3.20 add_keyboard_page()

```
void seq64::options::add_keyboard_page ( ) [private]
```

10.65.3.21 add_extended_keys_page()

```
void seq64::options::add_extended_keys_page ( ) [private]
```

10.65.3.22 add_mouse_page()

```
void seq64::options::add_mouse_page ( ) [private]
```

10.65.3.23 add_jack_sync_page()

```
void seq64::options::add_jack_sync_page ( ) [private]
```

10.65.4 Field Documentation

10.65.4.1 m_tooltips

```
Gtk::Tooltips* seq64::options::m_tooltips [private]
```

10.65.4.2 m_mainperf

```
perform& seq64::options::m_mainperf [private]
```

10.65.4.3 m_button_ok

```
Gtk::Button* seq64::options::m_button_ok [private]
```

10.65.4.4 m_button_jack_transport

```
Gtk::CheckButton* seq64::options::m_button_jack_transport [private]
```

10.65.4.5 m_button_jack_master

```
Gtk::CheckButton* seq64::options::m_button_jack_master [private]
```


10.65.4.6 m_button_jack_master_cond

Gtk::CheckButton* seq64::options::m_button_jack_master_cond [private]

10.65.4.7 m_button_jack_connect

Gtk::Button* seq64::options::m_button_jack_connect [private]

10.65.4.8 m_button_jack_disconnect

Gtk::Button* seq64::options::m_button_jack_disconnect [private]

10.65.4.9 m_notebook

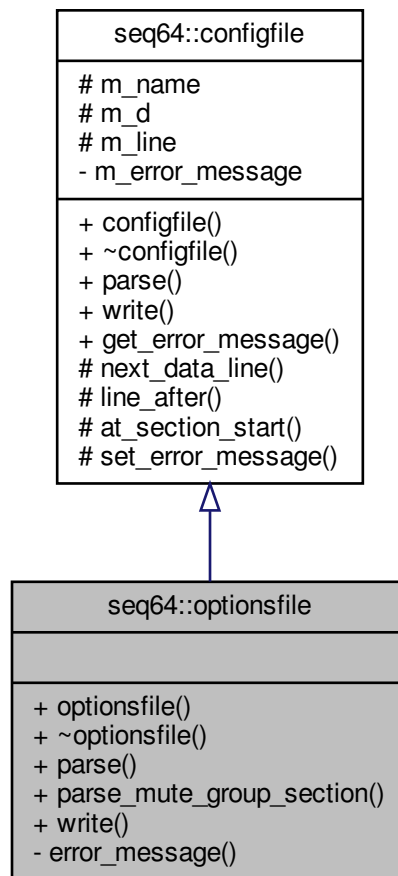
Gtk::Notebook* seq64::options::m_notebook [private]

Must be a GTK thang.

10.66 seq64::optionsfile Class Reference

Provides a file for reading and writing the application' main configuration file.

Inheritance diagram for seq64::optionsfile:



Public Member Functions

- `optionsfile` (const std::string &name)
Principal constructor.
- `~optionsfile` ()
A rote destructor.
- bool `parse` (perform &p)
Parse the `~/seq24rc` or `~/config/sequencer64/sequencer64.rc` file.
- bool `parse_mute_group_section` (perform &p)
Parses the [mute-group] section.
- bool `write` (const perform &p)
This options-writing function is just about as complex as the options-reading function.

Private Member Functions

- bool `error_message` (const std::string §ionname, const std::string &additional="")
Helper function for error-handling.

Additional Inherited Members

10.66.1 Detailed Description

The settings that are passed around are provided or used by the perform class.

10.66.2 Constructor & Destructor Documentation

10.66.2.1 optionsfile()

```
seq64::optionsfile::optionsfile (
    const std::string & name )
```

Parameters

<i>name</i>	Provides the name of the options file; this is usually a full path file-specification.
-------------	--

10.66.2.2 ~optionsfile()

```
seq64::optionsfile::~~optionsfile ( )
```

10.66.3 Member Function Documentation

10.66.3.1 parse()

```
bool seq64::optionsfile::parse (
    perform & p ) [virtual]
```

[midi-control]

Get the number of sequence definitions provided in the [midi-control] section. Ranges from 32 on up. Then read in all of the sequence lines. The first 32 apply to the first screen set. There can also be a comment line "# mute in group" followed by 32 more lines. Then there are additional comments and single lines for BPM up, BPM down, Screen Set Up, Screen Set Down, Mod Replace, Mod Snapshot, Mod Queue, Mod Gmute, Mod Glearn, and Screen Set Play. These are all forms of MIDI automation useful to control the playback while not sitting near the computer.

[mute-group]

The mute-group starts with a line that indicates up to 32 mute-groups are defined. A common value is 1024, which means there are 32 groups times 32 keys. But this value is currently thrown away. This value is followed by 32

lines of data, each contained 4 sets of 8 settings. See the seq24-doc project on GitHub for a much more detailed description of this section.

[midi-clock]

The MIDI-clock section defines the clocking value for up to 16 output busses. The first number, 16, indicates how many busses are specified. Generally, these busses are shown to the user with names such as "[1] seq24 1".

[keyboard-control]

The keyboard control defines the keys that will toggle the stage of each of up to 32 patterns in a pattern/sequence box. These keys are displayed in each box as a reminder. The first number specifies the Key number, and the second number specifies the Sequence number.

[keyboard-group]

The keyboard group specifies more automation for the application. The first number specifies the Key number, and the second number specifies the Group number. This section should be better described in the seq24-doc project on GitHub.

[extended-keys]

Additional keys (not yet represented in the Options dialog) to support additional keys for tempo-tapping, Seq32's new transport and connection functionality, and maybe a little more.

[New-keys]

Conditional support for reading Seq32 "rc" files.

[jack-transport]

This section covers various JACK settings, one setting per line. In order, the following numbers are specified:

- jack_transport - Enable sync with JACK Transport.
- jack_master - Seq24 will attempt to serve as JACK Master.
- jack_master_cond - Seq24 will fail to be Master if there is already a Master set.
- song_start_mode:
 - 0 = Playback will be in Live mode. Use this to allow muting and unmuting of loops.
 - 1 = Playback will use the Song Editor's data.

[midi-input]

This section covers the MIDI input busses, and has a format similar to "[midi-clock]". Generally, these busses are shown to the user with names such as "[1] seq24 1", and currently there is only one input buss. The first field is the port number, and the second number indicates whether it is disabled (0), or enabled (1).

[midi-clock-mod-ticks]

This section covers.... One common value is 64.

[manual-alsa-ports]

Set to 1 if you want seq24 to create its own ALSA ports and not connect to other clients.

[last-used-dir]

This section simply holds the last path-name that was used to read or write a MIDI file. We still need to add a check for a valid path, and currently the path must start with a "/", so it is not suitable for Windows.

[interaction-method]

This section specified the kind of mouse interaction.

- 0 = 'seq24' (original Seq24 method).
- 1 = 'fruity' (similar to a certain fruity sequencer we like).

The second data line is set to "1" if Mod4 can be used to keep seq24 in note-adding mode even after the right-click is released, and "0" otherwise.

Parameters

<i>p</i>	Provides the performance object to which all of these options apply.
----------	--

Returns

Returns true if the file was able to be opened for reading. Currently, there is no indication if the parsing actually succeeded.

One thing about MIDI clock values. If a device (e.g. my Korg nanoKEY2) is present in a system when Sequencer64 is exited, it will be saved in the [midi-clock] list. When unplugged, it will be read here at startup, but won't be shown. The next exit will find it removed from this list.

Also, we want to pre-allocate the number of clock entries needed, and then use the buss number to populate the list of clocks, in the odd event that the user changed the bus-order of the entries.

Implements [seq64::configfile](#).

10.66.3.2 parse_mute_group_section()

```
bool seq64::optionsfile::parse_mute_group_section (
    perform & p )
```

This function is used both in the original reading of the "rc" file, and for reloading the original mute-group data from the "rc".

We used to throw the mute-group count value away, since it was always 1024, but it is useful if no mute groups have been created. So, if it reads 0 (instead of 1024), we will assume there are no mute-group settings. We also have to be sure to go to the next data line even if the strip-empty-mutes option is on.

Parameters

<i>p</i>	Provides a reference to the main perform object. However, we have to cast away the constness, because too many of the perform getter functions are used in non-const contexts.
----------	--

Returns

Returns true if the file was able to be opened for reading, and the desired data successfully extracted.

10.66.3.3 write()

```
bool seq64::optionsfile::write (
    const perform & p ) [virtual]
```

Parameters

<i>p</i>	Provides a const reference to the main perform object. However, we have to cast away the constness, because too many of the perform getter functions are used in non-const contexts.
----------	--

Returns

Returns true if the write operations all succeeded.

New boolean to show sequence numbers; ignored in legacy mode.

Implements [seq64::configfile](#).

10.66.3.4 error_message()

```
bool seq64::optionsfile::error_message (
    const std::string & sectionname,
    const std::string & additional = "" ) [private]
```

It assembles a message and then passes it to [set_error_message\(\)](#).

Parameters

<i>sectionname</i>	Provides the name of the section for reporting the error.
<i>additional</i>	Additional context information to help in finding the error.

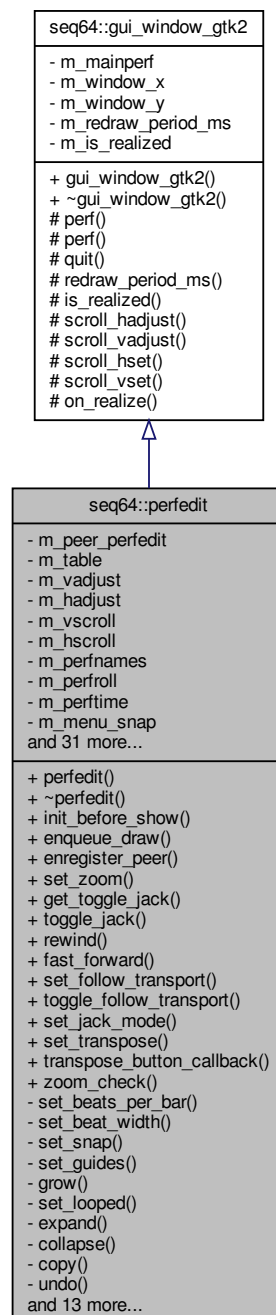
Returns

Always returns false.

10.67 seq64::perfedit Class Reference

This class supports a Performance Editor that is used to arrange the patterns/sequences defined in the patterns panel.

Inheritance diagram for seq64::perfedit:



Public Member Functions

- [perfedit](#) ([perform](#) &p, bool second_perfedit=false, int ppqn=SEQ64_USE_DEFAULT_PPQN)
Principal constructor, has a reference to a perform object.
- virtual [~perfedit](#) ()
This rote destructor does nothing.
- void [init_before_show](#) ()

- This function forwards its call to the perfroll function of the same name.*

 - void [enqueue_draw](#) (bool forward=true)
Helper wrapper for calling perfroll::queue_draw() for one or both perfedits.
 - void [enregister_peer](#) ([perfedit](#) *peer)
Register the peer perfedit object.
 - void [set_zoom](#) (int z)
Implements the horizontal zoom feature.
 - bool [get_toggle_jack](#) ()
Gets the state fo the JACK toggle button in the Song editor, when compiled with seq32 JACK support.
 - void [toggle_jack](#) ()
Sets the state fo the JACK toggle button in the Song editor, when compiled with seq32 JACK support.
 - void [rewind](#) (bool press)
Implements the seq32/stazed rewind operation.
 - void [fast_forward](#) (bool press)
Implements the seq32/stazed fast-forward operation.
 - void [set_follow_transport](#) ()
Sets the transport status when compiled for seq32 JACK support.
 - void [toggle_follow_transport](#) ()
Toggles the transport status when compiled for seq32 JACK support.
 - void [set_jack_mode](#) ()
Sets the JACK transport status, based on the status of the JACK button in the Song editor, when compiled for seq32 JACK support.
 - void [set_transpose](#) (int transpose)
Sets the value of transposition for this window.
 - void [transpose_button_callback](#) (int transpose)
The button callback for transposition for this window.

Static Public Member Functions

- static bool [zoom_check](#) (int z)
Checks zoom values for the z/Z keystrokes used in perfroll and perftime.

Private Member Functions

- void [set_beats_per_bar](#) (int bpm)
Sets the beats-per-measure text and value to the given value, and then calls [set_guides\(\)](#).
- void [set_beat_width](#) (int bw)
Sets the BW (beat width, or the denominator in the time signature) text and values to the given value, and then calls [set_guides\(\)](#).
- void [set_snap](#) (int snap)
Sets the snap text and values to the given value, and then calls [set_guides\(\)](#).
- void [set_guides](#) ()
Sets the guides, which are the L and R user-interface elements.
- void [grow](#) ()
Increments the size of the perfroll and perftime objects.
- void [set_looped](#) ()
Set the looping in the perform object.
- void [expand](#) ()
Implement the expand action.
- void [collapse](#) ()

- Implement the collapse action.*
- void [copy](#) ()
 - Implement the copy (actually, expand-and-copy) action.*
- void [undo](#) ()
 - Implement the undo feature (Ctrl-Z).*
- void [redo](#) ()
 - Implement the redo feature (Ctrl-R).*
- void [popup_menu](#) (Gtk::Menu *menu)
 - Opens the given popup menu.*
- void [draw_sequences](#) ()
 - Forces a redraw of the sequences, though currently just the perfnames part of each sequence in the performance editor.*
- bool [timeout](#) ()
 - Handles a drawing timeout.*
- void [set_image](#) (bool isrunning)
 - Changes the image used for the pause/play button.*
- void [start_playing](#) ()
 - Implement the playing.*
- void [pause_playing](#) ()
 - Pauses the playing of the song, leaving the progress bar where it stopped.*
- void [stop_playing](#) ()
 - Stop the playing.*
- void [toggle_playing](#) ()
 - Reverses the state of playback.*
- void [on_realize](#) ()
 - This callback function calls the base-class [on_realize\(\)](#) function, and then connects the [perfedit::timeout\(\)](#) function to the Glib signal-timeout, with a redraw timeout of [redraw_period_ms\(\)](#).*
- bool [on_key_press_event](#) (GdkEventKey *ev)
 - This function is the callback for a key-press event.*
- bool [on_key_release_event](#) (GdkEventKey *ev)
 - This function is the callback for a key-release event.*
- bool [on_delete_event](#) (GdkEventAny *)
 - All this callback function does is return false.*

Private Attributes

- [perfedit](#) * [m_peer_perfedit](#)
 - The partner instance of perfedit.*
- Gtk::Table * [m_table](#)
 - A whole horde of GUI elements.*
- Gtk::Adjustment * [m_vadjust](#)
 - Vertical adjust for piano roll.*
- Gtk::Adjustment * [m_hadjust](#)
 - Horizontal adjust for piano roll.*
- Gtk::VScrollbar * [m_vscroll](#)
 - Vertical scroll for piano roll.*
- Gtk::HScrollbar * [m_hscroll](#)
 - Horizontal scroll for piano roll.*
- [perfnames](#) * [m_perfnames](#)
 - Pattern names in leftmost column.*

- [perfroll](#) * [m_perfroll](#)
The piano roll in the song editor.
- [perftime](#) * [m_perftime](#)
The time/measures bar above roll.
- Gtk::Menu * [m_menu_snap](#)
The menu for grid-snap selection.
- Gtk::Menu * [m_menu_xpose](#)
The menu for transpose selection.
- Gtk::Button * [m_button_xpose](#)
Button to bring up transpose menu.
- Gtk::Entry * [m_entry_xpose](#)
Text edit for the transpose value.
- Gtk::Image * [m_image_play](#)
The image for the play button.
- Gtk::Button * [m_button_snap](#)
Button to bring up the snap menu.
- Gtk::Entry * [m_entry_snap](#)
Text edit for the grid-snap value.
- Gtk::Button * [m_button_stop](#)
The Stop Play button object.
- Gtk::Button * [m_button_play](#)
Implements the yellow two-bar pause button.
- Gtk::ToggleButton * [m_button_loop](#)
Button for Left-to-Right looping.
- Gtk::Button * [m_button_expand](#)
Button for Left/Right expansion.
- Gtk::Button * [m_button_collapse](#)
Button for Left/Right collapse.
- Gtk::Button * [m_button_copy](#)
Expand and copy between L/R.
- Gtk::Button * [m_button_grow](#)
Expand grid (bottom-right button).
- Gtk::Button * [m_button_undo](#)
Button to undo previous action.
- Gtk::Button * [m_button_redo](#)
Button to redo previous action.
- Gtk::ToggleButton * [m_button_jack](#)
Button to toggle JACK connection.
- Gtk::ToggleButton * [m_button_follow](#)
Button to toggle JACK following.
- Gtk::Button * [m_button_bpm](#)
Beats-per-measure menu button.
- Gtk::Entry * [m_entry_bpm](#)
Text-edit for beats-per-measure.
- Gtk::Button * [m_button_bw](#)
Beat-width menu button.
- Gtk::Entry * [m_entry_bw](#)
Text-edit for beat-width.
- Gtk::HBox * [m_hbox](#)
Horizontal box (which?) in table.
- Gtk::HBox * [m_hlbox](#)

- Horizontal box for buttons at top.*

 - `Gtk::Menu * m_menu_bpm`
Menus for time signature, beats per measure, beat width.
 - `Gtk::Menu * m_menu_bw`
Drop-down menu for beat-width.
 - `int m_snap`
Sets the horizontal grid snap-to in units of "pulses" or "ticks".
 - `int m_bpm`
The current "beats per measure" value.
 - `int m_bw`
The current "beat width" value.
 - `int m_ppqn`
The current "parts per quarter note" value.
 - `bool m_is_running`
Holds the current status of running, for use in display the play versus pause icon.
 - `int m_standard_bpm`
The standard "beats per measure" of Sequencer64, which here matches the beats-per-measure displayed in the perroll (piano roll).

Friends

- `void update_perfedit_sequences ()`
This global function in the seq64 namespace calls perfedit :: draw_sequences(), if the global perfedit objects exist.

Additional Inherited Members

10.67.1 Detailed Description

It has a seqroll and piano roll? No, it has a perform, a perfnames, a perroll, and a perftime.

10.67.2 Constructor & Destructor Documentation

10.67.2.1 perfedit()

```
seq64::perfedit::perfedit (
    perform & p,
    bool second_perfedit = false,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

We've reordered the pointer members and put them in the initializer list to make the constructor a bit cleaner.

Todo Offload most of the work into an initialization function like options does.

Parameters

<i>p</i>	Refers to the main performance object.
<i>second_perfedit</i>	If true, this object is the second perfedit object.
<i>ppqn</i>	The optionally-changed PPQN value to use for the performance editor.

10.67.2.2 ~perfedit()

```
virtual seq64::perfedit::~~perfedit ( ) [inline], [virtual]
```

We're going to have to run the application through valgrind to make sure that nothing is left behind.

10.67.3 Member Function Documentation

10.67.3.1 init_before_show()

```
void seq64::perfedit::init_before_show ( )
```

It does not seem to need to also forward to the perftime function of the same name.

10.67.3.2 enqueue_draw()

```
void seq64::perfedit::enqueue_draw (
    bool forward = true )
```

Note that we call the children's queue_draw() functions, not [enqueue_draw\(\)](#), otherwise we'll get stack overflow.

Parameters

<i>forward</i>	If true (the default), pass the call to the peer. When passing this call to the peer, this parameter is set to false to prevent an infinite loop and the resultant stack overflow.
----------------	--

10.67.3.3 zoom_check()

```
static bool seq64::perfedit::zoom_check (
    int z ) [inline], [static]
```

It has to range from greater than 1 (the highest zoom-in causes an unexplained drawing artifact at this time), and not greater than four times the c_perf_scale_x value, at which point we have zoomed out so far that the measure numbers are almost completely obscured.

Parameters

<i>z</i>	The desired zoom value to validate.
----------	-------------------------------------

10.67.3.4 enregister_peer()

```
void seq64::perfedit::enregister_peer (
    perfedit * peer ) [inline]
```

This function is meant to be called by mainwnd, which creates the perfedits and then makes sure they get along. Only the first call to this function will work; only one peer can be registered.

Parameters

<i>peer</i>	The peer perfedit object to register, if not null.
-------------	--

10.67.3.5 set_zoom()

```
void seq64::perfedit::set_zoom (
    int z )
```

Parameters

<i>z</i>	The zoom value to be set. The child zoom functions called each check that this value is valid.
----------	--

10.67.3.6 get_toggle_jack()

```
bool seq64::perfedit::get_toggle_jack ( )
```

Returns

Returns the JACK button's get_active() status.

10.67.3.7 toggle_jack()

```
void seq64::perfedit::toggle_jack ( )
```

Note that this will trigger the button signal callback.

10.67.3.8 rewind()

```
void seq64::perfedit::rewind (
    bool press ) [inline]
```

The timeout is in milliseconds, and is currently hard-wired to 120.

Note the use of "&perf()" to get the address of the perform object.

Parameters

<i>press</i>	True if the operation is a key press, false if the operation is a key release.
--------------	--

10.67.3.9 fast_forward()

```
void seq64::perfedit::fast_forward (
    bool press ) [inline]
```

Parameters

<i>press</i>	True if the operation is a key press, false if the operation is a key release.
--------------	--

10.67.3.10 set_follow_transport()

```
void seq64::perfedit::set_follow_transport ( )
```

Note that this will trigger the button signal callback.

10.67.3.11 toggle_follow_transport()

```
void seq64::perfedit::toggle_follow_transport ( )
```

Note that this will trigger the button signal callback.

10.67.3.12 set_jack_mode()

```
void seq64::perfedit::set_jack_mode ( )
```

To avoid a lot of pointer dereferencing, much of the code is offloaded to [perform::set_jack_mode\(\)](#), which now returns a boolean.

10.67.3.13 set_transpose()

```
void seq64::perfedit::set_transpose (
    int transpose )
```

Parameters

<i>transpose</i>	The amount to transpose the transposable sequences. We need to add validation at some point, if the widget does not enforce that.
------------------	---

10.67.3.14 transpose_button_callback()

```
void seq64::perfedit::transpose_button_callback (
    int transpose )
```

Parameters

<i>transpose</i>	The amount to transpose the transposable sequences.
------------------	---

10.67.3.15 set_beats_per_bar()

```
void seq64::perfedit::set_beats_per_bar (
    int bpm ) [private]
```

The usage of `is modified` was faulty. Offloaded it to the `perform` object to make it more foolproof. See the [perform↵::modify\(\)](#) function.

Parameters

<i>bpm</i>	Provides the beats/measure or beats/bar value to be set. This value is basically the numerator of the time signature.
------------	---

10.67.3.16 set_beat_width()

```
void seq64::perfedit::set_beat_width (
    int bw ) [private]
```

The usage of `is modified` was faulty. Offloaded it to the `perform` object to make it more foolproof. See the [perform↵::modify\(\)](#) function.

Parameters

<i>bw</i>	Provides the beat width to be set. The beat width is basically the denominator of the time signature.
-----------	---

10.67.3.17 set_snap()

```
void seq64::perfedit::set_snap (
    int snap ) [private]
```

Parameters

<i>snap</i>	Provide the snap value to be set. This value is basically the numerator of the expression "1 / snap".
-------------	---

10.67.3.18 set_guides()

```
void seq64::perfedit::set_guides ( ) [private]
```

See the [set_snap\(\)](#) function.

It's a little confusing; I assigned the label "m_standard_bpm" to the value 4 in "measure_pulse = 192 * 4 * m_bpm / m_bw", but I am not sure I understand this equation... why the extra factor of 4? That 4 appears in "c_ppqn * 4" a lot in the original code.

10.67.3.19 grow()

```
void seq64::perfedit::grow ( ) [private]
```

Make sure that setting the modified flag makes sense for this operation. It doesn't seem to modify members.

10.67.3.20 set_looped()

```
void seq64::perfedit::set_looped ( ) [private]
```

10.67.3.21 expand()

```
void seq64::perfedit::expand ( ) [private]
```

This action opens up a space of events between the L and R (left and right) markers. This action is preceded by pushing an Undo operation in the perform object, moving its triggers, and telling the perffroll to redraw.

10.67.3.22 collapse()

```
void seq64::perfedit::collapse ( ) [private]
```

This action removes all events between the L and R (left and right) markers. This action is preceded by pushing an Undo operation in the perform object, not moving its triggers (they go away), and telling the perffroll to redraw.

10.67.3.23 copy()

```
void seq64::perfedit::copy ( ) [private]
```

This action opens up a space of events between the L and R (left and right) markers, and copies the information from the same amount of events that follow the R marker. This action is preceded by pushing an Undo operation in the perform object, copying its triggers, and telling the perffroll to redraw.

10.67.3.24 undo()

```
void seq64::perfedit::undo ( ) [private]
```

We pop an Undo trigger, and then ask the perffroll to queue up a (re)drawing action.

10.67.3.25 redo()

```
void seq64::perfedit::redo ( ) [private]
```

We pop an Redo trigger, and then ask the perffroll to queue up a (re)drawing action.

10.67.3.26 popup_menu()

```
void seq64::perfedit::popup_menu (
    Gtk::Menu * menu ) [private]
```

10.67.3.27 draw_sequences()

```
void seq64::perfedit::draw_sequences ( ) [private]
```

This is meant to be called when the focus of an open seqedit or eventedit window changes.

10.67.3.28 timeout()

```
bool seq64::perfedit::timeout ( ) [private]
```

It redraws "dirty" sequences in the perffroll and the perffnames objects, and shows draw progress on the perffroll. It also changes the pause/play image if the status of running has changed. This function is called frequently and continuously. It will work for both perfedit windows, if both are up.

10.67.3.29 set_image()

```
void seq64::perfedit::set_image (
    bool isrunning ) [private]
```

Parameters

<i>isrunning</i>	If true, the image should be the pause image. Otherwise, it should be the play image.
------------------	---

10.67.3.30 start_playing()

```
void seq64::perfedit::start_playing ( ) [private]
```

JACK will be used if it is present and, in the application, enabled and working. Note the new flag to let perform know that it is a pause/play request from the perfedit window. In other words, a forced Song mode.

10.67.3.31 pause_playing()

```
void seq64::perfedit::pause_playing ( ) [private]
```

Keeps the stop button enabled as a kind of rewind for ALSA. Stop in place!

10.67.3.32 stop_playing()

```
void seq64::perfedit::stop_playing ( ) [private]
```

We need to make the progress line move back to the beginning right away here.

10.67.3.33 toggle_playing()

```
void seq64::perfedit::toggle_playing ( ) [inline], [private]
```

Meant only to be called when the "Play" button is pressed. Currently, the GUI does not change. This function will ultimately act like a Pause/Play button, but currently the pause functionality on works (partially) for JACK transport. Currently not used.

10.67.3.34 on_realize()

```
void seq64::perfedit::on_realize ( ) [private]
```

10.67.3.35 on_key_press_event()

```
bool seq64::perfedit::on_key_press_event (
    GdkEventKey * ev ) [private]
```

By default, the space-bar starts the playing, and the Escape key stops the playing. The start/end key may be the same key (i.e. space-bar), allow toggling when the same key is mapped to both triggers. Note that we now pass false in the call to [perform::playback_key_event\(\)](#), if SEQ64_PAUSE_SUPPORT is compiled in. Song mode doesn't yield the pause effect we want.

Parameters

<i>ev</i>	Provides the key event to implement.
-----------	--------------------------------------

10.67.3.36 on_key_release_event()

```
bool seq64::perfedit::on_key_release_event (
    GdkEventKey * ev ) [private]
```

It is needed to turn off the fast-forward and rewind keys functionality when released.

Parameters

<i>ev</i>	Provides the key event to implement.
-----------	--------------------------------------

10.67.3.37 on_delete_event()

```
bool seq64::perfedit::on_delete_event (
    GdkEventAny * ) [inline], [private]
```

10.67.4 Friends And Related Function Documentation

10.67.4.1 update_perfedit_sequences

```
void update_perfedit_sequences ( ) [friend]
```

It is used by other objects (seqedit and eventedit) that can modify the currently-edited sequence shown in the perfedit (song window).

10.67.5 Field Documentation

10.67.5.1 m_peer_perfedit

```
perfedit* seq64::perfedit::m_peer_perfedit [private]
```

10.67.5.2 m_table

Gtk::Table* seq64::perfedit::m_table [private]

Layout table for song editor.

10.67.5.3 m_vadjust

Gtk::Adjustment* seq64::perfedit::m_vadjust [private]

10.67.5.4 m_hadjust

Gtk::Adjustment* seq64::perfedit::m_hadjust [private]

10.67.5.5 m_vscroll

Gtk::VScrollbar* seq64::perfedit::m_vscroll [private]

10.67.5.6 m_hscroll

Gtk::HScrollbar* seq64::perfedit::m_hscroll [private]

10.67.5.7 m_perfnames

perfnames* seq64::perfedit::m_perfnames [private]

10.67.5.8 m_perfroll

perfroll* seq64::perfedit::m_perfroll [private]

10.67.5.9 m_perftime

perftime* seq64::perfedit::m_perftime [private]

10.67.5.10 m_menu_snap

Gtk::Menu* seq64::perfedit::m_menu_snap [private]

10.67.5.11 m_menu_xpose

Gtk::Menu* seq64::perfedit::m_menu_xpose [private]

10.67.5.12 m_button_xpose

Gtk::Button* seq64::perfedit::m_button_xpose [private]

10.67.5.13 m_entry_xpose

Gtk::Entry* seq64::perfedit::m_entry_xpose [private]

10.67.5.14 m_image_play

Gtk::Image* seq64::perfedit::m_image_play [private]

10.67.5.15 m_button_snap

Gtk::Button* seq64::perfedit::m_button_snap [private]

10.67.5.16 m_entry_snap

Gtk::Entry* seq64::perfedit::m_entry_snap [private]

10.67.5.17 m_button_stop

Gtk::Button* seq64::perfedit::m_button_stop [private]

10.67.5.18 m_button_play

```
Gtk::Button* seq64::perfedit::m_button_play [private]
```

The Play button object.

10.67.5.19 m_button_loop

```
Gtk::ToggleButton* seq64::perfedit::m_button_loop [private]
```

10.67.5.20 m_button_expand

```
Gtk::Button* seq64::perfedit::m_button_expand [private]
```

10.67.5.21 m_button_collapse

```
Gtk::Button* seq64::perfedit::m_button_collapse [private]
```

10.67.5.22 m_button_copy

```
Gtk::Button* seq64::perfedit::m_button_copy [private]
```

10.67.5.23 m_button_grow

```
Gtk::Button* seq64::perfedit::m_button_grow [private]
```

10.67.5.24 m_button_undo

```
Gtk::Button* seq64::perfedit::m_button_undo [private]
```

10.67.5.25 m_button_redo

```
Gtk::Button* seq64::perfedit::m_button_redo [private]
```

10.67.5.26 m_button_jack

Gtk::ToggleButton* seq64::perfedit::m_button_jack [private]

10.67.5.27 m_button_follow

Gtk::ToggleButton* seq64::perfedit::m_button_follow [private]

10.67.5.28 m_button_bpm

Gtk::Button* seq64::perfedit::m_button_bpm [private]

10.67.5.29 m_entry_bpm

Gtk::Entry* seq64::perfedit::m_entry_bpm [private]

10.67.5.30 m_button_bw

Gtk::Button* seq64::perfedit::m_button_bw [private]

10.67.5.31 m_entry_bw

Gtk::Entry* seq64::perfedit::m_entry_bw [private]

10.67.5.32 m_hbox

Gtk::HBox* seq64::perfedit::m_hbox [private]

10.67.5.33 m_hlbox

Gtk::HBox* seq64::perfedit::m_hlbox [private]

10.67.5.34 m_menu_bpm

```
Gtk::Menu* seq64::perfedit::m_menu_bpm [private]
```

Drop-down menu for beats/minute.

10.67.5.35 m_menu_bw

```
Gtk::Menu* seq64::perfedit::m_menu_bw [private]
```

10.67.5.36 m_snap

```
int seq64::perfedit::m_snap [private]
```

10.67.5.37 m_bpm

```
int seq64::perfedit::m_bpm [private]
```

Do not confuse it with BPM (beats per minute). The numerator of the time signature.

10.67.5.38 m_bw

```
int seq64::perfedit::m_bw [private]
```

The denominator of the time signature.

10.67.5.39 m_ppqn

```
int seq64::perfedit::m_ppqn [private]
```

10.67.5.40 m_is_running

```
bool seq64::perfedit::m_is_running [private]
```

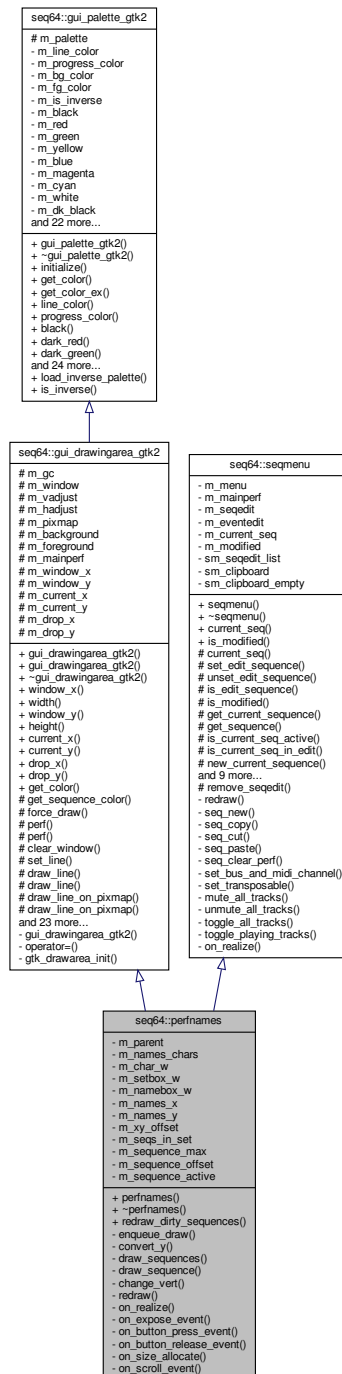
10.67.5.41 m_standard_bpm

```
int seq64::perfedit::m_standard_bpm [private]
```


10.68 seq64::perfnames Class Reference

This class implements the left-side keyboard in the patterns window.

Inheritance diagram for seq64::perfnames:



Public Member Functions

- [perfnames](#) (perform &p, [perfedit](#) &parent, Gtk::Adjustment &vadjust)

Principal constructor for this user-interface object.

- virtual `~perfnames ()`

Let's provide a do-nothing virtual destructor.

- void `redraw_dirty_sequences ()`

Redraws sequences that have been modified.

Private Member Functions

- void `enqueue_draw ()`

Wraps `queue_draw()` and forwards the call to the parent `perfed`, so that it can forward it to any other `perfed` that exists, and to the other sub-elements of the song editor.

- int `convert_y (int y)`

Converts a y-value into a sequence number and returns it.

- void `draw_sequences ()`

New function to encapsulate forced redrawing of all sequence names in the current viewport.

- void `draw_sequence (int sequence)`

Draw the given sequence.

- void `change_vert ()`

Change the vertical offset of a sequence/pattern.

- void `redraw (int sequence)`

Redraw the given sequence.

- void `on_realize ()`

Handles the callback when the window is realized.

- bool `on_expose_event (GdkEventExpose *ev)`

Handles an on-expose event.

- bool `on_button_press_event (GdkEventButton *ev)`

Provides the callback for a button press, and it handles only a left mouse button [the right mouse button is handled in `on_button_release_event()`].

- bool `on_button_release_event (GdkEventButton *ev)`

Handles a button-release for the right button, bringing up a popup menu that is identical to the right-click popup menu for a slot in the patterns panel (`mainwid`), and context sensitive.

- void `on_size_allocate (Gtk::Allocation &)`

Handles a size-allocation event.

- bool `on_scroll_event (GdkEventScroll *ev)`

Handle the vertical scrolling of the window.

Private Attributes

- `perfed` & `m_parent`

Provides a link to the `perfed` that created this object.

- int `m_names_chars`

Provides the number of the characters in the name box.

- int `m_char_w`

Provides the "real" width of a character.

- int `m_setbox_w`

Provides the width of the "set number" box.

- int `m_namebox_w`

Provides the width of the "name" box.

- int `m_names_x`

Provides the width of the names box, which is the width of a character for 24 characters.

- int [m_names_y](#)
Provides the height of the names box, which is hardwired to 24 pixels.
- int [m_xy_offset](#)
Provides the horizontal and vertical offsets of the text relative to the names box.
- const int [m_seqs_in_set](#)
The number of sequences in a set, currently still hardwired to 32.
- const int [m_sequence_max](#)
The maximum number of sequences, current $32 \times 32 = 1024$.
- int [m_sequence_offset](#)
The offset from the 0th sequence, which is determined by the vertical view of the piano roll, controlled by the vertical scroll-bar.
- bool [m_sequence_active](#) [[c_max_sequence](#)]
Indicates if the given sequence is active or not.

Friends

- class [perfedit](#)

Additional Inherited Members

10.68.1 Detailed Description

It inherits from [gui_drawingarea_gtk2](#) to support the font, color, and other GUI functionality, and from [seqmenu](#) to support the right-click Edit/New/Cut right-click menu.

Obsolete Note the usage of virtual base classes. Since these can add some extra overhead, we should determine if we can do without the virtuality (and indeed it doesn't seem to be needed).

10.68.2 Constructor & Destructor Documentation

10.68.2.1 perfnames()

```
seq64::perfnames::perfnames (
    perform & p,
    perfedit & parent,
    Gtk::Adjustment & vadjust )
```

Weird is that the window (x,y) are set to (c_names_x, 100), when c_names_y is 22 (now 24) in globals.h.

Parameters

<i>p</i>	Provides a reference to the main performance object of the application.
<i>parent</i>	Provides a reference to the object that contains this object, so that this object can tell the parent to queue up a drawing operation.
<i>vadjust</i>	Provides the vertical scrollbar object needed so that perfnames can respond to scrollbar cursor/thumb movement.

10.68.2.2 ~perfnames()

```
virtual seq64::perfnames::~~perfnames ( ) [inline], [virtual]
```

10.68.3 Member Function Documentation

10.68.3.1 redraw_dirty_sequences()

```
void seq64::perfnames::redraw_dirty_sequences ( )
```

10.68.3.2 enqueue_draw()

```
void seq64::perfnames::enqueue_draw ( ) [private]
```

The parent perfedit will call perfnames::queue_draw() on behalf of this object, and it will pass a [perfnames↵::enqueue_draw\(\)](#) to the peer perfedit's perfnames, if the peer exists.

10.68.3.3 convert_y()

```
int seq64::perfnames::convert_y (
    int y ) [private]
```

Used in figuring out which sequence to mute/unmute in the performance editor.

Parameters

y	The y value (within the vertical limits of the perfnames column to the left of the performance editor's piano roll.
----------	---

Returns

Returns the sequence number corresponding to the y value.

10.68.3.4 draw_sequences()

```
void seq64::perfnames::draw_sequences ( ) [private]
```

10.68.3.5 draw_sequence()

```
void seq64::perfnames::draw_sequence (
    int seqnum ) [private]
```

This function has to be prepared to handle an almost endless list of sequences, including unused ones, to draw them all with compatible styles. The sequences are grouped by set-number. The set-number occurs every 32 sequences in the leftmost column of the window.

1. Render the set number, or a blank box, in leftmost column. If the y height of the first draw_rectangle is `m_names_y + 1`, then we get a black line for the blank tracks, looks ugly.
2. Make sure that the rectangle drawn with the proper background colors for various combinations of muting and highlighting, otherwise just the name is properly colored.
3. Render the column with the name of the sequence. The channel number ranges from 1 to 16, but SMF 0 is indicated on-screen by a channel number of 0. We get the label format from the perform object, for consistency across windows.

Parameters

<i>seqnum</i>	Index to the sequence information to be drawn.
---------------	--

10.68.3.6 change_vert()

```
void seq64::perfnames::change_vert ( ) [private]
```

10.68.3.7 redraw()

```
void seq64::perfnames::redraw (
    int sequence ) [inline], [private], [virtual]
```

This function is a virtual function of seqmenu that must be overridden in this class.

Parameters

<i>sequence</i>	Provides the number of the sequence to be redrawn.
-----------------	--

Implements [seq64::seqmenu](#).

10.68.3.8 on_realize()

```
void seq64::perfnames::on_realize ( ) [private]
```

It first calls the base-class version of [on_realize\(\)](#). Then it allocates any additional resources needed.

10.68.3.9 on_expose_event()

```
bool seq64::perfnames::on_expose_event (
    GdkEventExpose * ev ) [private]
```

It draws all of the sequences that will be visible.

We could actually optimize this a tiny bit, to save some additions in the for loop.

Parameters

<i>ev</i>	The expose event, not used.
-----------	-----------------------------

Returns

Always returns true.

10.68.3.10 on_button_press_event()

```
bool seq64::perfnames::on_button_press_event (
    GdkEventButton * ev ) [private]
```

Two operations are supported by left-clicking on the sequence/track name:

- Normal. Toggles the mute status of the sequence that is clicked.
- Shift. Toggles the mutes status of all other sequences, making this operation an easy way to preview a single sequence in the performance editor, then bring back the rest of the tracks.

Parameters

<i>ev</i>	The mouse button event.
-----------	-------------------------

Returns

Always returns true.

10.68.3.11 on_button_release_event()

```
bool seq64::perfnames::on_button_release_event (
    GdkEventButton * p0 ) [private]
```

Parameters

<i>p0</i>	The button event.
-----------	-------------------

Returns

Always returns false.

10.68.3.12 on_size_allocate()

```
void seq64::perfnames::on_size_allocate (
    Gtk::Allocation & a ) [private]
```

It first calls the base-class version of this function.

Parameters

a	The allocation event. It is passed to the base-class on_size_allocate() function, and then m_window_x and m_window_y are set to the width and height, respectively, of the allocation.
----------	--

10.68.3.13 on_scroll_event()

```
bool seq64::perfnames::on_scroll_event (
    GdkEventScroll * ev ) [private]
```

The vertical value is incremented or decremented by the amount of the step increment, and the page is clamped to the new value.

Parameters

ev	The scrolling event.
-----------	----------------------

Returns

Always returns true.

10.68.4 Friends And Related Function Documentation**10.68.4.1 perfedit**

```
friend class perfedit [friend]
```

10.68.5 Field Documentation

10.68.5.1 m_parent

```
perfedit& seq64::perfnames::m_parent [private]
```

We want to support two perfedit windows, but the children of perfedit will have to communicate changes requiring a redraw through the parent.

10.68.5.2 m_names_chars

```
int seq64::perfnames::m_names_chars [private]
```

Pretty much hardwired to 24 at present.

10.68.5.3 m_char_w

```
int seq64::perfnames::m_char_w [private]
```

This value is obtained from a font-renderer accessor function.

10.68.5.4 m_setbox_w

```
int seq64::perfnames::m_setbox_w [private]
```

This used to be hardwired to $6 * 2$ (character-width times two).

10.68.5.5 m_namebox_w

```
int seq64::perfnames::m_namebox_w [private]
```

This used to be a weird calculation based on character width.

10.68.5.6 m_names_x

```
int seq64::perfnames::m_names_x [private]
```

10.68.5.7 m_names_y

```
int seq64::perfnames::m_names_y [private]
```

This value was once 22 pixels, but we need a little extra room for our new font. This extra room is compatible enough with the old font, as well.

10.68.5.8 m_xy_offset

```
int seq64::perfnames::m_xy_offset [private]
```

Currently hardwired.

10.68.5.9 m_seqs_in_set

```
const int seq64::perfnames::m_seqs_in_set [private]
```

Belay that, we now get it from [user_settings](#).

10.68.5.10 m_sequence_max

```
const int seq64::perfnames::m_sequence_max [private]
```

10.68.5.11 m_sequence_offset

```
int seq64::perfnames::m_sequence_offset [private]
```

10.68.5.12 m_sequence_active

```
bool seq64::perfnames::m_sequence_active[c_max_sequence] [private]
```

If this really is the true meaning of this value, we ought to get it directly from the sequence if we can.

10.69 seq64::perform Class Reference

This class supports the performance mode.

Public Types

- enum `action_t` {
`ACTION_NONE`,
`ACTION_SEQ_TOGGLE`,
`ACTION_GROUP_MUTE`,
`ACTION_BPM`,
`ACTION_SCREENSET`,
`ACTION_GROUP_LEARN`,
`ACTION_C_STATUS` }

In many cases, when we check a key action that perform wil do, it is sufficient to return a boolean.

- enum `record_tempo_op_t` {
`RECORD_TEMPO_LOG_EVENT`,
`RECORD_TEMPO_ON`,
`RECORD_TEMPO_OFF` }

Provides settings for tempo recording.

- enum `mute_op_t` {
`MUTE_TOGGLE`,
`MUTE_OFF`,
`MUTE_ON` }

Provides settings for muting.

- enum `ff_rw_button_t` {
`FF_RW_REWIND`,
`FF_RW_NONE`,
`FF_RW_FORWARD` }

Provides setting for the fast-forward and rewind functionality.

Public Member Functions

- `perform` (`gui_assistant` &`mygui`, int `ppqn`=SEQ64_USE_DEFAULT_PPQN)

This construction initializes a vast number of member variables, some of them public (but we're working on that)!

- `~perform` ()

The destructor sets some running flags to false, signals this condition, then joins the input and output threads if the were launched.

- bool `is_modified` () const

'Getter' function for member `m_is_modified`

- void `modify` ()

'Setter' function for member `m_is_modified` This setter only sets the modified-flag to true.

- int `ppqn` () const

'Getter' function for member `m_ppqn`

- `midibpm bpm` () const

'Getter' function for member `m_bpm`

- int `sequence_count` () const

'Getter' function for member `m_sequence_count` It is better to call this getter before bothering to even try to use a sequence.

- int `sequence_high` () const

'Getter' function for member `m_sequence_high`

- int `sequence_max` () const

'Getter' function for member `m_sequence_max`

- int `group_max` () const

'Getter' function for member `m_max_groups`

- bool `is_control_status` () const

'Getter' function for member `m_control_status`

- bool [midi_mute_group_present](#) () const
'Getter' function for member m_midi_mute_group_present
- void [set_edit_sequence](#) (int seqnum)
'Setter' function for member m_edit_sequence
- void [unset_edit_sequence](#) (int seqnum)
'Setter' function for member m_edit_sequence
- bool [is_edit_sequence](#) (int seqnum) const
'Getter' function for member m_edit_sequence
- int [get_beats_per_bar](#) () const
'Getter' function for member m_beats_per_bar
- void [set_beats_per_bar](#) (int bpm)
'Setter' function for member m_beats_per_bar
- int [get_beat_width](#) () const
'Getter' function for member m_beat_width
- void [set_beat_width](#) (int bw)
'Setter' function for member m_beat_width
- int [get_tempo_track_number](#) () const
'Getter' function for member m_tempo_track_number
- void [set_tempo_track_number](#) (int tempotrack)
'Setter' function for member m_tempo_track_number
- void [clocks_per_metronome](#) (int cpm)
'Setter' function for member m_clocks_per_metronome
- int [clocks_per_metronome](#) () const
'Getter' function for member m_clocks_per_metronome
- void [set_32nds_per_quarter](#) (int tpq)
'Setter' function for member m_32nds_per_quarter
- int [get_32nds_per_quarter](#) () const
'Getter' function for member m_32nds_per_quarter
- void [us_per_quarter_note](#) (long upqn)
'Setter' function for member m_us_per_quarter_note
- long [us_per_quarter_note](#) () const
'Getter' function for member m_us_per_quarter_note
- const [gui_assistant](#) & [gui](#) () const
'Getter' function for member m_gui_support The const getter.
- [gui_assistant](#) & [gui](#) ()
'Getter' function for member m_gui_support The un-const getter.
- const [keys_perform](#) & [keys](#) () const
'Getter' function for member m_gui_support.keys() The const getter.
- [keys_perform](#) & [keys](#) ()
'Getter' function for member m_gui_support.keys() The un-const getter.
- [mastermidibus](#) & [master_bus](#) ()
'Getter' function for member m_master_bus Obviously, this is a dangerous function, but we've got ya covered.
- void [filter_by_channel](#) (bool flag)
'Setter' function for member m_master_bus.filter_by_channel()
- bool [is_running](#) () const
'Getter' function for member m_is_running Could also be called "is_playing()".
- bool [is_pattern_playing](#) () const
'Setter' function for member m_is_pattern_playing
- void [is_pattern_playing](#) (bool flag)
'Setter' function for member m_is_pattern_playing
- bool [toggle_song_start_mode](#) ()

- 'Setter' function for member `m_song_start_mode`
- void `song_start_mode` (bool flag)
- 'Setter' function for member `m_song_start_mode`
- bool `song_start_mode` () const
- 'Getter' function for member `m_song_start_mode`
- bool `is_jack_running` () const
- 'Getter' function for member `m_jack_asst.is_running()` This function is useful for announcing the status of JACK in user-interface items that only have access to the perform object.
- bool `is_jack_master` () const
- 'Getter' function for member `m_jack_asst.is_master()` Also now includes `is_jack_running()`, since one cannot be JACK Master if JACK is not running.
- void `enregister` (performcallback *pfcfb)
- Adds a pointer to an object to be notified by this perform object.
- void `toggle_jack_mode` ()
- bool `set_jack_mode` (bool mode)
- Encapsulates behavior needed by perfedit.
- bool `get_toggle_jack` () const
- 'Getter' function for member `m_jack_asst.get_jack_mode()`
- void `set_jack_stop_tick` (midipulse tick)
- 'Setter' function for member `m_jack_asst.set_jack_stop_tick()`
- unsigned short `combine_bytes` (midibyte b0, midibyte b1)
- Combines bytes into an unsigned-short value.
- void `FF_rewind` ()
- Implements the fast-forward or rewind functionality imported from seq32.
- bool `FF_RW_timeout` ()
- Convenience function.
- void `start_from_perfeddit` (bool flag)
- 'Setter' function for member `m_start_from_perfeddit`
- bool `start_from_perfeddit` () const
- 'Getter' function for member `m_start_from_perfeddit`
- void `set_follow_transport` (bool flag)
- 'Getter' function for member `m_jack_asst.set_follow_transport()`
- bool `get_follow_transport` () const
- 'Getter' function for member `m_jack_asst.get_follow_transport()`
- void `toggle_follow_transport` ()
- 'Setter' function for member `m_jack_asst.toggle_follow_transport()`
- bool `follow_progress` () const
- Convenience function for following progress in seqedit.
- void `set_reposition` (bool postype=true)
- 'Getter' function for member `m_reposition`
- `ff_rw_button_t ff_rw_type` ()
- 'Getter' function for member `m_FF_RW_button_type`
- void `ff_rw_type` (`ff_rw_button_t` button_type)
- 'Getter' function for member `m_FF_RW_button_type`
- void `rewind` (bool press)
- Sets the rewind status.
- void `fast_forward` (bool press)
- Sets the fast-forward status.
- void `reposition` (midipulse tick)
- Encapsulates some repositioning code needed to move the position to the mouse pointer location in perroll.
- void `set_sequence_input` (bool active, `sequence` *s)

- 'Getter' function for member `m_master_bus`->`set_sequence_input()`
- void `set_recording` (bool `rec_active`, bool `thru_active`, `sequence` *s)
 - Encapsulates code used by `seqedit::record_change_callback()`.
- void `set_recording` (bool `rec_active`, int `seq`, bool `toggle`=false)
 - Encapsulates code used by `seqedit::record_change_callback()`.
- void `set_quantized_recording` (bool `rec_active`, `sequence` *s)
 - Sets quantized recording in the way used by `seqedit`.
- void `set_quantized_recording` (bool `rec_active`, int `seq`, bool `toggle`=false)
 - Sets quantized recording.
- void `set_overwrite_recording` (bool `overwrite_active`, int `seq`, bool `toggle`=false)
 - Set recording for overwrite.
- void `set_thru` (bool `rec_active`, bool `thru_active`, `sequence` *s)
 - Encapsulates code used by `seqedit::thru_change_callback()`.
- void `set_thru` (bool `thru_active`, int `seq`, bool `toggle`=false)
 - Encapsulates code used by `seqedit::thru_change_callback()`.
- bool `selected_trigger` (int `seqnum`, `midipulse` droptick, `midipulse` &tick0, `midipulse` &tick1)
 - Encapsulates getting the trigger limits without putting the burden on the caller.
- bool `clear_all` ()
 - Clears all of the patterns/sequences.
- void `launch` (int `ppqn`)
 - Calls the MIDI buss and JACK initialization functions and the input/output thread-launching functions.
- void `finish` ()
 - The rough opposite of `launch()`; it doesn't stop the threads.
- void `new_sequence` (int `seq`)
 - Creates a new pattern/sequence for the given slot, and sets the new pattern's master MIDI bus address.
- void `add_sequence` (`sequence` *seq, int `perf`)
 - Adds a pattern/sequence pointer to the list of patterns.
- void `delete_sequence` (int `seq`)
 - Deletes a pattern/sequence by number.
- bool `is_sequence_in_edit` (int `seq`)
 - Check if the pattern/sequence, given by number, has an edit in progress.
- void `print_busses` () const
 - Shows all the triggers of all the sequences.
- `midipulse` `get_tick` () const
 - 'Getter' function for member `m_tick`
- void `set_tick` (`midipulse` tick)
 - This version for song-recording not only logs `m_tick`, it also does JACK positioning (if applicable), calls the master bus's `continue_from()` function, and sets `m_current_tick` as well.
- `midipulse` `get_jack_tick` () const
 - 'Getter' function for member `m_jack_tick`
- void `set_jack_tick` (`midipulse` tick)
 - 'Setter' function for member `m_jack_tick`
- void `set_left_tick` (`midipulse` tick, bool `setstart`=true)
 - Set the left marker at the given tick.
- `midipulse` `get_left_tick` () const
 - 'Getter' function for member `m_left_tick`
- void `set_start_tick` (`midipulse` tick)
 - 'Setter' function for member `m_starting_tick`
- `midipulse` `get_start_tick` () const
 - 'Setter' function for member `m_starting_tick`
- void `set_right_tick` (`midipulse` tick, bool `setstart`=true)

- Set the right marker at the given tick.*

 - `midipulse get_right_tick ()` const
 - 'Getter' function for member `m_right_tick`*
- `double left_right_size ()` const

Convenience function for JACK support when loop in song mode.
- `bool is_active (int seq)` const

Checks the pattern/sequence for activity.
- `void apply_song_transpose ()`

Calls the `apply_song_transpose()` function for all active sequences.
- `void set_transpose (int t)`

'Setter' function for member `m_transpose` For sanity's sake, the values are restricted to +-64.
- `int get_transpose ()` const

'Getter' function for member `m_transpose`
- `midibpm get_beats_per_minute ()`

'Getter' function for member `m_master_bus.get_beats_per_minute` Retrieves the BPM setting of the master MIDI buss.
- `bool reload_mute_groups (std::string &errmsg)`

Reloads the mute groups from the "rc" file.
- `bool clear_mute_groups ()`

Clears all the group-mute items, whether they came from the "rc" file or from the most recently-loaded Sequencer64 MIDI file.
- `void set_sequence_control_status (int status)`

If the given status is present in the `c_status_snapshot`, the playing state is saved.
- `void unset_sequence_control_status (int status)`

If the given status is present in the `c_status_snapshot`, the playing state is restored.
- `void unset_queued_replace (bool clearbits=true)`

Helper function that clears the queued-replace feature.
- `void sequence_playing_toggle (int seq)`

If the given sequence is active, then it is toggled as per the current value of `m_control_status`.
- `void sequence_playing_change (int seq, bool on)`

Turn the playing of a sequence on or off.
- `void set_keep_queue (bool activate)`

Sets or unsets the keep-queue functionality, to be used by the new "Q" button in the main window.
- `bool is_keep_queue ()` const

Returns true if the `c_status_queue` bit is set.
- `void sequence_playing_on (int seq)`

Calls `sequence_playing_change()` with a value of true.
- `void sequence_playing_off (int seq)`

Calls `sequence_playing_change()` with a value of false.
- `void mute_all_tracks (bool flag=true)`

Mutes/unmutes all tracks in the current set of active patterns/sequences.
- `void toggle_all_tracks ()`

Toggles the mutes status of all tracks in the current set of active patterns/sequences.
- `bool armed_saved ()` const

'Getter' function for member `m_armed_saved`
- `void toggle_playing_tracks ()`

Toggles the mutes status of all playing (currently unmuted) tracks in the current set of active patterns/sequences on all screen-sets.
- `void mute_screenset (int ss, bool flag=true)`

Mutes/unmutes all tracks in the desired screen-set.
- `void output_func ()`

- Performance output function.*

 - void [input_func](#) ()

This function is called by [input_thread_func\(\)](#).
- void [set_group_mute_state](#) (int gtrack, bool muted)

This function sets the mute state of an element in the `m_mute_group` array.
- bool [get_group_mute_state](#) (int gtrack)

The opposite of [set_group_mute_state\(\)](#), it gets the value of the desired track.
- int [mute_group_offset](#) (int track)

A helper function to calculate the index into the mute-group array, based on the desired track.
- int [screenset_offset](#) () const

'Getter' function for member `m_screenset_offset`
- int [slot_number](#) (int s)

Translates a pattern number to a slot number re the current screenset offset.
- void [save_playing_state](#) ()

For all active patterns/sequences, this function gets the playing status and saves it in `m_sequence_state[i]`.
- void [restore_playing_state](#) ()

For all active patterns/sequences, this function gets the playing status from `m_sequence_state[i]` and sets it for the sequence.
- void [save_current_screenset](#) (int repseq)

For all active patterns/sequences in the current (playing) screen-set, this function gets the playing status and saves it in `m_sequence_state[i]`.
- void [clear_current_screenset](#) ()

Clears the `m_screenset_state[]` array.
- std::string [key_name](#) (unsigned k) const

Here follows a few forwarding functions for the keys_perform-derived classes.
- [keys_perform::SlotMap](#) & [get_key_events](#) ()

Forwarding function for key events.
- int [get_key_count](#) (unsigned k) const

Returns the number of times the given key appears in the SlotMap, either 0 or 1.
- [keys_perform::SlotMap](#) & [get_key_groups](#) ()

Forwarding function for key groups.
- [keys_perform::RevSlotMap](#) & [get_key_events_rev](#) ()

Forwarding function for reverse key events.
- [keys_perform::RevSlotMap](#) & [get_key_groups_rev](#) ()

Forwarding function for reverse key groups.
- bool [show_ui_sequence_key](#) () const

'Getter' function for member `m_show_ui_sequency_key` Provides access to [keys\(\).show_ui_sequence_key\(\)](#).
- void [show_ui_sequence_key](#) (bool flag)

'Setter' function for member `m_show_ui_sequency_key`
- bool [show_ui_sequence_number](#) () const

'Getter' function for member `m_show_ui_sequency_number` Provides access to [keys\(\).show_ui_sequence_number\(\)](#).
- void [show_ui_sequence_number](#) (bool flag)

'Getter' function for member `m_show_ui_sequency_number`
- unsigned [lookup_keyevent_key](#) (int seqnum)

Gets the event key for the given sequence.
- unsigned [lookup_slot_key](#) (int slotnum)

Like [lookup_keyevent_key\(\)](#), but assumes the slot number has already been correctly calculated.
- int [lookup_keyevent_seq](#) (unsigned keycode)

Gets the sequence number for the given event key.
- unsigned [lookup_keygroup_key](#) (int groupnum)

Gets the group key for the given sequence.

- int `lookup_keygroup_group` (unsigned keycode)
Gets the group number for the given group key.
- void `start_playing` (bool songmode=false)
Encapsulates a series of calls used in mainwnd.
- void `pause_playing` (bool songmode=false)
Pause playback, so that progress bars stay where they are, and playback always resumes where it left off, at least in ALSA mode, which doesn't have to worry about being a "slave".
- void `stop_playing` ()
Encapsulates a series of calls used in mainwnd.
- void `start_key` (bool songmode=false)
Invoke the start key functionality.
- void `pause_key` (bool songmode=false)
Invoke the pause key functionality.
- void `stop_key` ()
Invoke the stop key functionality.
- void `learn_toggle` ()
Encapsulates some calls used in mainwnd.
- `midibpm decrement_beats_per_minute` ()
Encapsulates some calls used in mainwnd.
- `midibpm increment_beats_per_minute` ()
Encapsulates some calls used in mainwnd.
- `midibpm page_decrement_beats_per_minute` ()
Provides additional coarse control over the BPM value, which comes into force when the Page-Up/Page-Down keys are pressed.
- `midibpm page_increment_beats_per_minute` ()
Provides additional coarse control over the BPM value, which comes into force when the Page-Up/Page-Down keys are pressed.
- int `decrement_screenset` (int amount=1)
Encapsulates some calls used in mainwnd.
- int `increment_screenset` (int amount=1)
Encapsulates some calls used in mainwnd.
- bool `highlight` (const `sequence` &seq) const
True if a sequence is empty and should be highlighted.
- bool `is_smf_0` (const `sequence` &seq) const
True if the sequence is an SMF 0 sequence.
- const `sequence` * `get_sequence` (int seq) const
Retrieves the actual sequence, based on the pattern/sequence number.
- `sequence` * `get_sequence` (int seq)
Retrieves the actual sequence, based on the pattern/sequence number.
- void `sequence_key` (int seq)
Handle a sequence key to toggle the playing of an active pattern in the selected screen-set.
- std::string `sequence_label` (const `sequence` &seq)
Provides a way to format the sequence parameters string for display in the mainwid or perfnames modules.
- std::string `sequence_label` (int seqnumb)
A pass-through to the other `sequence_label()` function.
- std::string `sequence_title` (const `sequence` &seq)
Creates the sequence title, adjusting it for scaling down.
- std::string `main_window_title` ()
Creates the main window title.
- std::string `sequence_window_title` (const `sequence` &seq)
Creates a sequence ("seqedit") window title, a longer version of `sequence_title()`.

- void [set_input_bus](#) (bussbyte bus, bool input_active)

Sets the input bus, and handles the special "key labels on sequence" and "sequence numbers on sequence" functionality.
- void [set_clock_bus](#) (bussbyte bus, [clock_e](#) clocktype)

Sets the clock value, as specified in the Options / MIDI Clocks tab.
- bool [mainwnd_key_event](#) (const [keystroke](#) &k)

Provided for mainwnd :: on_key_press_event() and mainwnd :: on_key_release_event() to call.
- bool [keyboard_control_press](#) (unsigned key)

Still need to work on this one.
- bool [keyboard_group_c_status_press](#) (unsigned key)

Categories of keyboard actions:
- bool [keyboard_group_c_status_release](#) (unsigned key)
- bool [keyboard_group_press](#) (unsigned key)
- bool [keyboard_group_release](#) (unsigned key)
- [action_t](#) [keyboard_group_action](#) (unsigned key)
- bool [perroll_key_event](#) (const [keystroke](#) &k, int drop_sequence)

Provided for perroll :: on_key_press_event() and perroll :: on_key_release_event() to call.
- bool [playback_key_event](#) (const [keystroke](#) &k, bool songmode=false)

New function provided to unify the stop/start (space/escape) behavior of the various windows where playback can be started, paused, or stopped.
- void [clear_sequence_triggers](#) (int seq)

Clears the patterns/sequence for the given sequence, if it is active.
- void [print_triggers](#) () const

Shows all the triggers of all the sequences.
- void [move_triggers](#) (bool direction)

If the left tick is less than the right tick, then, for each sequence that is active, its triggers are moved by the difference between the right and left in the specified direction.
- void [copy_triggers](#) ()

If the left tick is less than the right tick, then, for each sequence that is active, its triggers are copied, offset by the difference between the right and left.
- void [push_trigger_undo](#) (int track=SEQ64_ALL_TRACKS)

For every active sequence, call that sequence's [push_trigger_undo\(\)](#) function.
- void [pop_trigger_undo](#) ()

For every active sequence, call that sequence's [pop_trigger_undo\(\)](#) function.
- void [pop_trigger_redo](#) ()

For every active sequence, call that sequence's [pop_trigger_redo\(\)](#) function.
- bool [get_trigger_state](#) (int seqnum, [midipulse](#) tick) const

Pass-along to [sequence::get_trigger_state\(\)](#).
- void [add_trigger](#) (int seqnum, [midipulse](#) tick)

Adds a trigger on behalf of a sequence.
- void [delete_trigger](#) (int seqnum, [midipulse](#) tick)

Delete the existing specified trigger.
- void [add_or_delete_trigger](#) (int seqnum, [midipulse](#) tick)

Add a new trigger if nothing is selected, otherwise delete the existing trigger.
- void [split_trigger](#) (int seqnum, [midipulse](#) tick)

Convenience function for perroll's split-trigger functionality.
- void [paste_trigger](#) (int seqnum, [midipulse](#) tick)

Convenience function for perroll's paste-trigger functionality.
- void [paste_or_split_trigger](#) (int seqnum, [midipulse](#) tick)

Convenience function for perroll's paste-or-split-trigger functionality.
- bool [intersect_triggers](#) (int seqnum, [midipulse](#) tick)

Finds the trigger intersection.

- [midipulse get_max_trigger](#) () const
Locates the largest trigger value among the active sequences.
- bool [is_dirty_main](#) (int seq)
Checks the pattern/sequence for main-dirtiness.
- bool [is_dirty_edit](#) (int seq)
Checks the pattern/sequence for edit-dirtiness.
- bool [is_dirty_perf](#) (int seq)
Checks the pattern/sequence for perf-dirtiness.
- bool [is_dirty_names](#) (int seq)
Checks the pattern/sequence for names-dirtiness.
- bool [is_exportable](#) (int seq) const
Indicates that the desired sequence is active, unmuted, and has a non-zero trigger count.
- int [set_screenset](#) (int ss)
Sets the `m_screenset` value (the index or ID of the current screen-set).
- int [screenset](#) () const
'Getter' function for member `m_screenset`
- int [get_playing_screenset](#) () const
'Getter' function for member `m_playscreen`
- bool [toggle_other_seqs](#) (int seqnum, bool isshiftkey)
This code handles the use of the Shift key to toggle the mute state of all other sequences.
- bool [toggle_other_names](#) (int seqnum, bool isshiftkey)
This code handles the use of the Shift key to toggle the mute state of all other sequences.
- bool [are_any_armed](#) ()
Indicates if any sequences are armed (playing).
- void [max_sets](#) (int sets)
'Setter' function for member `m_max_sets` This setter is needed to modify the value after reading the "user" file.
- void [seqs_in_set](#) (int seqs)
'Setter' function for member `m_seqs_in_set` This setter modifies the current value based on the current values of the settings found in the [user_settings](#) module.
- bool [song_recording](#) () const
- bool [song_record_snap](#) () const
- bool [resume_note_ons](#) () const
- void [resume_note_ons](#) (bool f)
- bool [select_trigger](#) (int dropseq, [midipulse](#) droptick)
Selects a trigger for the given sequence.
- void [unselect_all_triggers](#) ()
Calls `sequence::unselect_triggers()` for all active sequences.
- const std::string & [get_bank_name](#) (int bank) const
A better name for `get_screen_set_notepad()`, adapted from Kepler34.
- void [set_looping](#) (bool looping)
'Setter' function for member `m_looping`
- int [get_sequence_color](#) (int seqnum) const
Deals with the colors used to represent specific sequences.
- void [set_sequence_color](#) (int seqnum, int c)
- bool [have_undo](#) () const
'Getter' function for member `m_have_undo`
- void [set_have_undo](#) (bool undo)
'Setter' function for member `m_have_undo` Note that, if the undo parameter is true, then we mark the performance as modified.
- bool [have_redo](#) () const
'Getter' function for member `m_have_redo`

- void [set_have_redo](#) (bool redo)
'Setter' function for member m_have_redo
- [edit_mode_t seq_edit_mode](#) (int seq) const
- void [seq_edit_mode](#) (int seq, [edit_mode_t](#) ed)
A pass-along function to set the edit-mode of the given sequence.
- const std::string & [current_screenset_notepad](#) () const
Returns the notepad text for the current screen-set.
- void [set_screenset_notepad](#) (int [screenset](#), const std::string ¬e, bool is_load_modification=false)
Copies the given string into m_screenset_notepad[].
- void [set_screenset_notepad](#) (const std::string ¬e)
Sets the notepad text for the current screen-set.
- void [start](#) (bool state)
If JACK is not running, call [inner_start\(\)](#) with the given state.
- void [stop](#) ()
If JACK is not running, call [inner_stop\(\)](#).
- void [start_jack](#) ()
If JACK is supported, starts the JACK transport.
- void [stop_jack](#) ()
If JACK is supported, stops the JACK transport.
- void [song_recording_stop](#) ()
Calls sequence::song_recording_stop(m_current_tick) for all sequences.
- void [song_recording](#) (bool f)
- void [song_record_snap](#) (bool f)
- bool [playback_mode](#) ()
'Getter' function for member m_playback_mode
- void [playback_mode](#) (bool playbackmode)
'Setter' function for member m_playback_mode
- bool [is_group_learning](#) ()
'Getter' function for member m_mode_group_learn
- void [set_beats_per_minute](#) (midibpm bpm)
Sets the value of the BPM into the master MIDI buss, after making sure it is squelched to be between 20 and 500.
- void [panic](#) ()
Similar to [all_notes_off\(\)](#), but also sends Note Off events directly to the active busses.

Private Member Functions

- void [ppqn](#) (int p)
'Setter' function for member m_ppqn
- void [collapse](#) ()
Convenience function for perfedit's collapse functionality.
- void [copy](#) ()
Convenience function for perfedit's copy functionality.
- void [expand](#) ()
Convenience function for perfedit's expand functionality.
- [midi_control](#) & [midi_control_toggle](#) (int ctl)
Retrieves a reference to a value from m_midi_cc_toggle[].
- [midi_control](#) & [midi_control_on](#) (int ctl)
Retrieves a reference to a value from m_midi_cc_on[].
- [midi_control](#) & [midi_control_off](#) (int ctl)
Retrieves a reference to a value from m_midi_cc_off[].

- bool [midi_control_event](#) (const [event](#) &ev)
This function encapsulates code in [input_func\(\)](#) to make it easier to read and understand.
- bool [midi_control_record](#) (const [event](#) &ev)
Checks the event to see if it is a [c_midi_control_record](#) event, and performs the requested action (toggle, on, off) if so.
- bool [handle_midi_control](#) (int control, bool state)
Handle the MIDI Control values that provide some automation for the application.
- bool [handle_midi_control_ex](#) (int control, [midi_control::action](#) a, int v)
Provides operation of the new MIDI controls.
- bool [handle_midi_control_event](#) (const [event](#) &ev, int ctrl, int offset=0)
Code extracted from [midi_control_event\(\)](#) to be re-used for handling shorter lists of events.
- const std::string & [get_screenset_notepad](#) (int [screenset](#)) const
Retrieves the given string from [m_screenset_notepad\[\]](#).
- bool [any_group_unmutes](#) () const
'Getter' function for member [m_mute_group\[\]](#)
- void [print_group_unmutes](#) () const
This function is a way to dump the mute-group settings in a way independent of the code in the optionsfile module, for debugging.
- void [mute_group_tracks](#) ()
If [m_mode_group](#) is true, then this function operates.
- void [select_and_mute_group](#) (int g_group)
Select a mute group and then mutes the track in the group.
- void [set_song_mute](#) ([mute_op_t](#) op)
Provides for various settings of the song-mute status of all sequences in the song.
- void [set_playing_screenset](#) ()
Sets the screen-set that is active, based on the value of [m_screenset](#).
- void [set_mode_group_mute](#) ()
'Setter' function for member [m_mode_group](#)
- void [unset_mode_group_mute](#) ()
'Setter' function for member [m_mode_group](#) Unsets this member.
- void [select_group_mute](#) (int gmute)
If we're in group-learn mode, then this function gets the playing statuses of all of the sequences in the current play-screen, and copies them into the desired mute-group.
- void [set_mode_group_learn](#) ()
Sets the group-mute mode, then the group-learn mode, then notifies all of the notification subscribers.
- void [unset_mode_group_learn](#) ()
Notifies all of the notification subscribers that group-learn is being turned off.
- bool [load_mute_group](#) (int gmute, int gm [[c_max_groups](#)])
Combines [select_group_mute\(\)](#) and [set_group_mute_state\(\)](#) so that the optionsfile class can load the groups without altering the [m_mute_group_selected](#) item; doing that is a bit misleading.
- bool [save_mute_group](#) (int gmute, int gm [[c_max_groups](#)]) const
The converse of [load_mute_group\(\)](#).
- void [set_and_copy_mute_group](#) (int group)
When in group-learn mode, for active sequences, the mute-group settings are set based on the playing status of each sequence.
- bool [activate](#) ()
Performs a controlled activation of the [jack_assistant](#) and other JACK modules.
- void [position_jack](#) (bool state, [midipulse](#) tick=0)
If JACK is supported and running, sets the position of the transport.
- void [off_sequences](#) ()
For all active patterns/sequences, set the playing state to false.
- void [unqueue_sequences](#) (int current_seq)

- Does a toggle-queueing for all of the sequences in the current screenset, for all sequences that are on, and for the currently hot-keyed sequence.*
- void `all_notes_off` ()
For all active patterns/sequences, turn off its playing notes.
 - void `set_active` (int seq, bool active)
Sets or unsets the active state of the given pattern/sequence number.
 - void `set_was_active` (int seq)
Sets was-active flags: main, edit, perf, and names.
 - void `reset_sequences` (bool pause=false)
For all active patterns/sequences, get its playing state, turn off the playing notes, set playing to false, zero the markers, and, if not in playback mode, restore the playing state.
 - void `play` (midipulse tick)
Plays all notes to the current tick.
 - void `set_orig_ticks` (midipulse tick)
For every pattern/sequence that is active, sets the "original tick" value for the pattern.
 - int `max_active_set` () const
Checks the whole universe of sequences to determine the current last-active set, that is, the highest set that has any active sequences in it.
 - void `launch_input_thread` ()
Creates the input thread using `input_thread_func()`.
 - void `launch_output_thread` ()
Creates the output thread using `output_thread_func()`.
 - bool `init_jack_transport` ()
Initializes JACK support, if SEQ64_JACK_SUPPORT is defined.
 - bool `deinit_jack_transport` ()
Tears down the JACK infrastructure.
 - bool `seq_in_playing_screen` (int seq)
A helper function for determining if the mode group is in force, the playing screenset is the same as the current screenset, and the sequence is in the range of the playing screenset.
 - void `is_modified` (bool flag)
'Setter' function for member `m_is_modified`
 - bool `valid_midi_control_seq` (int seq) const
Checks the parameter against `c_midi_controls_extended`.
 - int `max_sets` () const
'Getter' function for member `m_max_sets`
 - bool `is_screenset_valid` (int screenset) const
Checks the screenset against `m_max_sets`.
 - void `is_running` (bool running)
'Setter' function for member `m_is_running`
 - int `screenset_offset` (int ss)
Calculates the screen-set offset index.
 - bool `is_seq_valid` (int seq) const
Provides common code to check for the bounds of a sequence number.
 - bool `is_mseq_valid` (int seq) const
Validates the sequence number, which is important since they're currently used as array indices.
 - bool `install_sequence` (sequence *seq, int seqnum)
A private helper function for `add_sequence()` and `new_sequence()`.
 - void `inner_start` (bool state)
Locks on `m_condition_var`.
 - void `inner_stop` (bool midiclock=false)
Unconditionally, and without locking, clears the running status and resets the sequences.

- int `clamp_track` (int track) const
Provides common code to keep the track value valid.
- int `clamp_group` (int group) const
Provides common code to keep the group value valid even in variset mode.
- void `set_key_event` (unsigned keycode, int sequence_slot)
At construction time, this function sets up one keycode and one event slot.
- void `set_key_group` (unsigned keycode, int group_slot)
At construction time, this function sets up one keycode and one group slot.
- bool `log_current_tempo` ()
Used by callers to insert tempo events.
- bool `create_master_bus` ()
Creates the mastermidibus.
- void `preallocate_clocks` (int busscount)
Pre-allocates the desired number of clocks.
- void `add_clock` (clock_e clocktype)
Saves the clock settings read from the "rc" file so that they can be passed to the mastermidibus after it is created.
- void `set_clock` (bussbyte bus, clock_e clocktype)
Sets a single clock item, if in the currently existing range.
- clock_e `get_clock` (bussbyte bus) const
Gets a single clock item, if in the currently existing range.
- void `add_input` (bool flag)
Saves the input settings read from the "rc" file so that they can be passed to the mastermidibus after it is created.
- void `set_input` (bussbyte bus, bool inputing)
Sets a single input item, if in the currently existing range.
- bool `get_input` (bussbyte bus) const
'Getter' function for member m_master_inputs[bus]
- bool `is_input_system_port` (bussbyte bus)
'Getter' function for member m_master_bus->is_input_system_port(bus)
- void `midi_mute_group_present` (bool flag)
'Setter' function for member m_midi_mute_group_present

Private Attributes

- bool `m_song_start_mode`
If true, playback is done in Song mode, not Live mode.
- bool `m_start_from_perfedit`
Indicates that, no matter what the current Song/Live setting, the playback was started from the perfedit window.
- bool `m_reposition`
It seems that this member, if true, forces a repositioning to the left (L) tick marker.
- float `m_excell_FF_RW`
Provides an "acceleration" factor for the fast-forward and rewind functionality.
- ff_rw_button_t `m_FF_RW_button_type`
Indicates whether the fast-forward or rewind key is in effect in the perfedit window.
- bool `m_mute_group` [c_max_sequence]
Mute group support.
- bool `m_mute_group_rc` [c_max_sequence]
Preserves the mute groups from the "rc" file, so that they won't necessarily be overwritten by the mute groups contained in a Sequencer64 MIDI file.
- bool `m_armed_saved`
Indicates if the m_saved_armed_statuses[] values are the saved state of the sequences, and can be restored.

- bool [m_armed_statuses](#) [c_max_sequence]
Holds the "global" saved status of the playing tracks, for restoration after saving.
- int [m_seqs_in_set](#)
We have replaced c_seqs_in_set with this member, which defaults to the value of c_seqs_in_set, but is grabbed from user_settings now.
- int [m_max_groups](#)
Since we can increase the number of sequences in a set, we need to be able to decrease the number of sets or groups we can store.
- std::vector< bool > [m_tracks_mute_state](#)
Holds the current mute states of each track.
- bool [m_mode_group](#)
If true, indicates that a mode group is selected, and playing statuses will be "memorized".
- bool [m_mode_group_learn](#)
If true, indicates that a group learn is selected, which also "memorizes" a mode group, and notifies subscribers of a group-learn change.
- int [m_mute_group_selected](#)
Selects a group to mute.
- bool [m_midi_mute_group_present](#)
If true, indicates that non-zero mute-groups were present in this MIDI file.
- [sequence](#) * [m_seqs](#) [c_max_sequence]
Provides a "vector" of patterns/sequences.
- bool [m_seqs_active](#) [c_max_sequence]
Each boolean value in this array is set to true if a sequence is active, meaning that it will be used to hold some kind of MIDI data, even if only Meta events.
- bool [m_was_active_main](#) [c_max_sequence]
Each boolean value in this array is set to true if a sequence was active, meaning that it was found to be active at the time we were setting it to inactive.
- bool [m_was_active_edit](#) [c_max_sequence]
Each boolean value in this array is set to true if a sequence was active, meaning that it was found to be active at the time we were setting it to inactive.
- bool [m_was_active_perf](#) [c_max_sequence]
Each boolean value in this array is set to true if a sequence was active, meaning that it was found to be active at the time we were setting it to inactive.
- bool [m_was_active_names](#) [c_max_sequence]
Each boolean value in this array is set to true if a sequence was active, meaning that it was found to be active at the time we were setting it to inactive.
- bool [m_sequence_state](#) [c_max_sequence]
Saves the current playing state of each pattern.
- std::vector< bool > [m_screenset_state](#)
Saves the current playing state only for the current set.
- int [m_queued_replace_slot](#)
A value not equal to -1 (it ranges from 0 to 32) ndicates we're now using the saved screen-set state to control the queue-replace (queue-solo) status of sequence toggling.
- int [m_transpose](#)
Holds the global MIDI transposition value.
- pthread_t [m_out_thread](#)
Provides information for managing pthreads.
- pthread_t [m_in_thread](#)
Provides a "handle" to the input thread.
- bool [m_out_thread_launched](#)
Indicates that the output thread has been started.
- bool [m_in_thread_launched](#)

- Indicates that the input thread has been started.*

 - bool [m_is_running](#)
- Indicates that playback is running.*

 - bool [m_is_pattern_playing](#)
- Indicates that a pattern is playing.*

 - bool [m_inputting](#)
- Indicates that events are being written to the MIDI input busses in the input thread.*

 - bool [m_outputting](#)
- Indicates that events are being written to the MIDI output busses in the output thread.*

 - bool [m_looping](#)
- Indicates that status of the "loop" button in the performance editor.*

 - bool [m_song_recording](#)
- Indicates to record live sequence-trigger changes into the Song data.*

 - bool [m_song_record_snap](#)
- Snap recorded playback changes to the sequence length.*

 - bool [m_resume_note_ons](#)
- Indicates to resume notes if the sequence is toggled after a Note On.*

 - double [m_current_tick](#)
- The global current tick, moved out from the output function so that position can be set.*

 - bool [m_playback_mode](#)
- Specifies the playback mode.*

 - int [m_ppqn](#)
- Holds the current PPQN for usage in various actions.*

 - [midibpm m_bpm](#)
- Holds the current BPM (beats per minute) for later usage.*

 - int [m_beats_per_bar](#)
- Holds the beats/bar value as obtained from the MIDI file.*

 - int [m_beat_width](#)
- Holds the beat width value as obtained from the MIDI file.*

 - int [m_tempo_track_number](#)
- Holds the number of the official tempo track for this performance.*

 - int [m_clocks_per_metronome](#)
- Augments the beats/bar and beat-width with the additional values included in a Time Signature meta event.*

 - int [m_32nds_per_quarter](#)
- Augments the beats/bar and beat-width with the additional values included in a Time Signature meta event.*

 - long [m_us_per_quarter_note](#)
- Augments the beats/bar and beat-width with the additional values included in a Tempo meta event.*

 - [mastermidibus * m_master_bus](#)
- Provides our MIDI buss.*

 - bool [m_filter_by_channel](#)
- Provides storage for this "rc" configuration option so that the perform object can set it in the master buss once that has been created.*

 - `std::vector< clock_e >` [m_master_clocks](#)
- Saves the clock settings obtained from the "rc" (options) file so that they can be loaded into the mastermidibus once it is created.*

 - `std::vector< bool >` [m_master_inputs](#)
- Saves the input settings obtained from the "rc" (options) file so that they can be loaded into the mastermidibus once it is created.*

 - [midipulse m_one_measure](#)
- Holds the "one measure's worth" of pulses (ticks), which is normally $m_ppqn * 4$.*

 - [midipulse m_left_tick](#)

- Holds the position of the left (L) marker, and it is first defined as 0.*

 - [midipulse m_right_tick](#)
- Holds the position of the right (R) marker, and it is first defined as the end of the fourth measure.*

 - [midipulse m_starting_tick](#)
- Holds the starting tick for playing.*

 - [midipulse m_tick](#)
- MIDI Clock support.*

 - [midipulse m_jack_tick](#)
- Let's try to save the last JACK pad structure tick for re-use with resume after pausing.*

 - [bool m_usemidiclock](#)
- More MIDI clock support.*

 - [bool m_midiclockrunning](#)
- More MIDI clock support.*

 - [int m_midiclocktick](#)
- More MIDI clock support.*

 - [int m_midiclockpos](#)
- More MIDI clock support.*

 - [bool m_dont_reset_ticks](#)
- Support for pause, which does not reset the "last tick" when playback stops/starts.*

 - [std::string m_screenset_notepad \[c_max_sets\]](#)
- Used in the mainwnd class to set the notepad text for the given set.*

 - [midi_control m_midi_cc_toggle \[c_midi_controls_extended\]](#)
- Provides the settings of MIDI Toggle, as read from the "rc" file.*

 - [midi_control m_midi_cc_on \[c_midi_controls_extended\]](#)
- Provides the settings of MIDI On, as read from the "rc" file.*

 - [midi_control m_midi_cc_off \[c_midi_controls_extended\]](#)
- Provides the settings of MIDI Off, as read from the "rc" file.*

 - [int m_control_status](#)
- Holds the OR'ed control status values.*

 - [int m_screenset](#)
- Indicates the number of the currently-selected screen-set.*

 - [int m_screenset_offset](#)
- Holds the current sequence-number offset for the current screen-set.*

 - [int m_playscreen](#)
- Playing screen support.*

 - [int m_playscreen_offset](#)
- Playing screen sequence number offset.*

 - [int m_max_sets](#)
- A replacement for the c_max_sets constant.*

 - [int m_sequence_count](#)
- Keeps track of created sequences, whether or not they are active.*

 - [int m_sequence_max](#)
- A replacement for the c_max_sequence constant.*

 - [int m_sequence_high](#)
- Indicates the highest-number sequence.*

 - [int m_edit_sequence](#)
- Hold the number of the currently-in-edit sequence.*

 - [bool m_is_modified](#)
- It may be a good idea to eventually centralize all of the dirtiness of a performance here.*

 - [condition_var m_condition_var](#)
- A condition variable to protect playback.*

- [jack_assistant m_jack_asst](#)
A wrapper object for the JACK support of this application.
- bool [m_have_undo](#)
- `std::vector< int >` [m_undo_vect](#)
Holds the "track" numbers or the "all tracks" values for undo operations.
- bool [m_have_redo](#)
Used for redo track modification support.
- `std::vector< int >` [m_redo_vect](#)
Holds the "track" numbers or the "all tracks" values for redo operations.
- `std::vector< performcallback * >` [m_notify](#)
- [gui_assistant](#) & [m_gui_support](#)
Support for a wide range of GUI-related operations.

Static Private Attributes

- static [midi_control](#) [sm_mc_dummy](#)
Provides a dummy, inactive [midi_control](#) object to handle out-of-range [midi_control](#) indices.

Friends

- class [jack_assistant](#)
- class [keybindentry](#)
- class [mainwnd](#)
- class [midifile](#)
- class [wrkfile](#)
- class [optionsfile](#)
- class [options](#)
- class [perfedit](#)
- class [perfroll](#)
- class [sequence](#)
- void * [input_thread_func](#) (void *myperf)
Set up the performance, and set the process to realtime privileges.
- void * [output_thread_func](#) (void *myperf)
Global functions defined in perform.cpp.
- int [jack_sync_callback](#) (jack_transport_state_t state, jack_position_t *pos, void *arg)
- int [jack_transport_callback](#) (jack_nframes_t nframes, void *arg)
Implemented second patch for JACK Transport from freddix/seq24 GitHub project.
- void [jack_shutdown](#) (void *arg)
- void [jack_timebase_callback](#) (jack_transport_state_t state, jack_nframes_t nframes, jack_position_t *pos, int new_pos, void *arg)
The JACK timebase function defined here sets the JACK position structure.
- long [get_current_jack_position](#) (void *arg)
This function gets the current JACK position.

10.69.1 Detailed Description

It has way too many data members. Might be ripe for refactoring. That has its own dangers, of course.

One thing to do soon is remove the need to having GUI classes as friends. Will make some necessary setters public.

10.69.2 Member Enumeration Documentation

10.69.2.1 action_t

```
enum seq64::perform::action_t
```

But, in some cases, we need to indicate what was changed (e.g. via a keystroke). This enumeration provides return values that a (GUI) caller can use to decide which values to get and then change the user-interface to indicate the new value.

See the [keyboard_group_action\(\)](#) function and the "[keyboard-control]" and "[keyboard-group]" configuration sections of the "rc" file.

Enumerator

ACTION_NONE	The keystroke was not handled by perform.
ACTION_SEQ_TOGGLE	For perform::sequence_playing_toggle() .
ACTION_GROUP_MUTE	See mainwnd::on_key_press_event() . ???
ACTION_BPM	Applies to any BPM change, including tap.
ACTION_SCREENSET	The keystroke altered the active set.
ACTION_GROUP_LEARN	See mainwnd::on_key_press_event() . ???
ACTION_C_STATUS	For replace, queue, snapshot, oneshot.

10.69.2.2 record_tempo_op_t

```
enum seq64::perform::record_tempo_op_t
```

Currently not used, though the functionality of logging and recording tempo is in place.

Enumerator

RECORD_TEMPO_LOG_EVENT	
RECORD_TEMPO_ON	
RECORD_TEMPO_OFF	

10.69.2.3 mute_op_t

```
enum seq64::perform::mute_op_t
```

Enumerator

MUTE_TOGGLE	
MUTE_OFF	
MUTE_ON	

10.69.2.4 ff_rw_button_t

```
enum seq64::perform::ff_rw_button_t
```

Enumerator

FF_RW_REWIND	
FF_RW_NONE	
FF_RW_FORWARD	

10.69.3 Constructor & Destructor Documentation

10.69.3.1 perform()

```
seq64::perform::perform (
    gui_assistant & mygui,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

Also note that we have a little issue with the fact that various sequences (patterns) can potentially have different beats/measure and beat-width values.

Currently, when reading the MIDI file, the beats/minute value is obtained from the MIDI file, if present, and this value is passed to [perform::set_beats_per_minute\(\)](#), which forwards it to the master MIDI buss and JACK assistant objects. This Tempo setting comes from both the Tempo meta event in track 0, and from the Seq24's c_bpm SeqSpec section! This setting is now also made for the two Time Signature values.

But note that Sequencer64 now scales the c_bpm value so that two extra digits of precision can be saved with the MIDI file. We went throughout the code, changing BPM from an integer to a double.

Parameters

<i>mygui</i>	Provides access to the GUI assistant that holds many things, including the containers of keys and the "events" they provide. This is a base-class reference; for a real class, see the gui_assistant_gtk2 class in the seq_gtkmm2 GUI-specific library. Note that we access the m_gui_support member using the gui() accessor function.
<i>ppqn</i>	The default, choosable, or actual PPQN value.

10.69.3.2 ~perform()

```
seq64::perform::~~perform ( )
```

Finally, any active or inactive (but allocated) patterns/sequences are deleted, and their pointers nullified.

Note that we could use `m_sequence_high` to replace `m_sequence_max` in the for-loop, but who cares, we are exiting!

10.69.4 Member Function Documentation

10.69.4.1 `is_modified()` [1/2]

```
bool seq64::perform::is_modified ( ) const [inline]
```

10.69.4.2 `modify()`

```
void seq64::perform::modify ( ) [inline]
```

The setter that will, [is_modified\(\)](#), is private. No one but perform and its friends should falsify this flag.

10.69.4.3 `ppqn()` [1/2]

```
int seq64::perform::ppqn ( ) const [inline]
```

10.69.4.4 `bpm()`

```
midibpm seq64::perform::bpm ( ) const [inline]
```

10.69.4.5 `sequence_count()`

```
int seq64::perform::sequence_count ( ) const [inline]
```

In many cases at startup, or when loading a file, there are no sequences yet, and still the code calls functions that try to access them.

10.69.4.6 `sequence_high()`

```
int seq64::perform::sequence_high ( ) const [inline]
```

10.69.4.7 sequence_max()

```
int seq64::perform::sequence_max ( ) const [inline]
```

10.69.4.8 group_max()

```
int seq64::perform::group_max ( ) const [inline]
```

10.69.4.9 is_control_status()

```
bool seq64::perform::is_control_status ( ) const [inline]
```

Returns

Returns true if the `m_control_status` value is non-zero, which means that there is a queue, replace, or snapshot functionality in progress.

10.69.4.10 midi_mute_group_present() [1/2]

```
bool seq64::perform::midi_mute_group_present ( ) const [inline]
```

10.69.4.11 set_edit_sequence()

```
void seq64::perform::set_edit_sequence (
    int seqnum ) [inline]
```

Parameters

<i>seqnum</i>	Pass in -1 to disable the edit-sequence number unconditionally. Use unset_edit_sequence() to disable it if it matches the current edit-sequence number.
---------------	---

10.69.4.12 unset_edit_sequence()

```
void seq64::perform::unset_edit_sequence (
    int seqnum ) [inline]
```

Disables the edit-sequence number if it matches the parameter.

Parameters

<i>seqnum</i>	The sequence number of the sequence to unset.
---------------	---

10.69.4.13 is_edit_sequence()

```
bool seq64::perform::is_edit_sequence (
    int seqnum ) const [inline]
```

Parameters

<i>seqnum</i>	Tests the parameter against m_edit_sequence. Returns true if that member is not -1, and the parameter matches it.
---------------	---

10.69.4.14 get_beats_per_bar()

```
int seq64::perform::get_beats_per_bar ( ) const [inline]
```

10.69.4.15 set_beats_per_bar()

```
void seq64::perform::set_beats_per_bar (
    int bpm ) [inline]
```

Parameters

<i>bpm</i>	Provides the value for beats/measure. Also used to set the beats/measure in the JACK assistant object.
------------	--

10.69.4.16 get_beat_width()

```
int seq64::perform::get_beat_width ( ) const [inline]
```

10.69.4.17 set_beat_width()

```
void seq64::perform::set_beat_width (
    int bw ) [inline]
```

Parameters

<i>bw</i>	Provides the value for beat-width. Also used to set the beat-width in the JACK assistant object.
-----------	--

10.69.4.18 `get_tempo_track_number()`

```
int seq64::perform::get_tempo_track_number ( ) const [inline]
```

10.69.4.19 `set_tempo_track_number()`

```
void seq64::perform::set_tempo_track_number (
    int tempotrack ) [inline]
```

Parameters

<i>tempotrack</i>	Provides the value for beat-width. Also used to set the beat-width in the JACK assistant object.
-------------------	--

10.69.4.20 `clocks_per_metronome()` [1/2]

```
void seq64::perform::clocks_per_metronome (
    int cpm ) [inline]
```

10.69.4.21 `clocks_per_metronome()` [2/2]

```
int seq64::perform::clocks_per_metronome ( ) const [inline]
```

10.69.4.22 `set_32nds_per_quarter()`

```
void seq64::perform::set_32nds_per_quarter (
    int tpq ) [inline]
```


10.69.4.23 get_32nds_per_quarter()

```
int seq64::perform::get_32nds_per_quarter ( ) const [inline]
```

10.69.4.24 us_per_quarter_note() [1/2]

```
void seq64::perform::us_per_quarter_note (
    long upqn ) [inline]
```

10.69.4.25 us_per_quarter_note() [2/2]

```
long seq64::perform::us_per_quarter_note ( ) const [inline]
```

10.69.4.26 gui() [1/2]

```
const gui_assistant& seq64::perform::gui ( ) const [inline]
```

10.69.4.27 gui() [2/2]

```
gui_assistant& seq64::perform::gui ( ) [inline]
```

10.69.4.28 keys() [1/2]

```
const keys_perform& seq64::perform::keys ( ) const [inline]
```

10.69.4.29 keys() [2/2]

```
keys_perform& seq64::perform::keys ( ) [inline]
```

10.69.4.30 master_bus()

```
mastermidibus& seq64::perform::master_bus ( ) [inline]
```

10.69.4.31 filter_by_channel()

```
void seq64::perform::filter_by_channel (
    bool flag ) [inline]
```

10.69.4.32 is_running() [1/2]

```
bool seq64::perform::is_running ( ) const [inline]
```

10.69.4.33 is_pattern_playing() [1/2]

```
bool seq64::perform::is_pattern_playing ( ) const [inline]
```

10.69.4.34 is_pattern_playing() [2/2]

```
void seq64::perform::is_pattern_playing (
    bool flag ) [inline]
```

10.69.4.35 toggle_song_start_mode()

```
bool seq64::perform::toggle_song_start_mode ( ) [inline]
```

10.69.4.36 song_start_mode() [1/2]

```
void seq64::perform::song_start_mode (
    bool flag ) [inline]
```

10.69.4.37 song_start_mode() [2/2]

```
bool seq64::perform::song_start_mode ( ) const [inline]
```

10.69.4.38 is_jack_running()

```
bool seq64::perform::is_jack_running ( ) const [inline]
```

10.69.4.39 is_jack_master()

```
bool seq64::perform::is_jack_master ( ) const [inline]
```

10.69.4.40 enregister()

```
void seq64::perform::enregister (
    performcallback * pfcbl ) [inline]
```

Parameters

<i>pfcbl</i>	Provides the pointer to the performance callback.
--------------	---

10.69.4.41 toggle_jack_mode()

```
void seq64::perform::toggle_jack_mode ( ) [inline]
```

10.69.4.42 set_jack_mode()

```
bool seq64::perform::set_jack_mode (
    bool jack_button_active )
```

Note that we moved some of the code from [perfedit::set_jack_mode\(\)](#) [the seq32 version] to this function.

Parameters

<i>jack_button_active</i>	Indicates if the perfedit JACK button shows it is active.
---------------------------	---

Returns

Returns true if JACK is running currently, and false otherwise.

10.69.4.43 get_toggle_jack()

```
bool seq64::perform::get_toggle_jack ( ) const [inline]
```

10.69.4.44 set_jack_stop_tick()

```
void seq64::perform::set_jack_stop_tick (
    midipulse tick ) [inline]
```

10.69.4.45 combine_bytes()

```
unsigned short seq64::perform::combine_bytes (
    midibyte b0,
    midibyte b1 )
```

<http://www.blitter.com/~russtopia/MIDI/~jglatt/tech/midispec/wheel.htm>

Two data bytes follow the status. The two bytes should be combined together to form a 14-bit value. The first data byte's bits 0 to 6 are bits 0 to 6 of the 14-bit value. The second data byte's bits 0 to 6 are really bits 7 to 13 of the 14-bit value. In other words, assuming that a C program has the first byte in the variable First and the second data byte in the variable Second, here's how to combine them into a 14-bit value (actually 16-bit since most computer CPUs deal with 16-bit, not 14-bit, integers).

I think Kepler64 got the bytes backward.

Parameters

<i>b0</i>	The first byte to be combined.
<i>b1</i>	The second byte to be combined.

Returns

Returns the bytes basically OR'd together.

10.69.4.46 FF_rewind()

```
void seq64::perform::FF_rewind ( )
```

It changes m_tick by a quarter of the number of ticks in a standard measure, with m_excell_FF_RW (defaults to one) to factor the difference.

10.69.4.47 FF_RW_timeout()

```
bool seq64::perform::FF_RW_timeout ( )
```

This function is used in the free function version of [FF_RW_timeout\(\)](#) as a callback to the `gtk_timeout()` function. It multiplies `m_excell_FF_RW` by 1.1 as long as one of the fast-forward or rewind keys is held, and is less than 60.

Returns

Returns true if one of the fast-forward or rewind keys was held, leaving `m_excell_FF_RW` at the last value it had. Otherwise, it resets the value to 1, and returns false.

10.69.4.48 start_from_perfedit() [1/2]

```
void seq64::perform::start_from_perfedit (
    bool flag ) [inline]
```

10.69.4.49 start_from_perfedit() [2/2]

```
bool seq64::perform::start_from_perfedit ( ) const [inline]
```

10.69.4.50 set_follow_transport()

```
void seq64::perform::set_follow_transport (
    bool flag ) [inline]
```

10.69.4.51 get_follow_transport()

```
bool seq64::perform::get_follow_transport ( ) const [inline]
```

10.69.4.52 toggle_follow_transport()

```
void seq64::perform::toggle_follow_transport ( ) [inline]
```

10.69.4.53 follow_progress()

```
bool seq64::perform::follow_progress ( ) const [inline]
```

10.69.4.54 set_reposition()

```
void seq64::perform::set_reposition (
    bool postype = true ) [inline]
```

10.69.4.55 ff_rw_type() [1/2]

```
ff_rw_button_t seq64::perform::ff_rw_type ( ) [inline]
```

10.69.4.56 ff_rw_type() [2/2]

```
void seq64::perform::ff_rw_type (
    ff_rw_button_t button_type ) [inline]
```

10.69.4.57 rewind()

```
void seq64::perform::rewind (
    bool press ) [inline]
```

Parameters

<i>press</i>	If true, the status is set to FF_RW_REWIND, otherwise it is set to FF_RW_NONE.
--------------	--

10.69.4.58 fast_forward()

```
void seq64::perform::fast_forward (
    bool press ) [inline]
```

Parameters

<i>press</i>	If true, the status is set to FF_RW_FORWARD, otherwise it is set to FF_RW_NONE.
--------------	---

10.69.4.59 reposition()

```
void seq64::perform::reposition (
    midipulse tick )
```

Used only in `perroll :: on_key_press_event()` to implement the Seq32 pointer-position feature.

Parameters

<i>tick</i>	Provides the position value to be set.
-------------	--

10.69.4.60 set_sequence_input()

```
void seq64::perform::set_sequence_input (
    bool active,
    sequence * s ) [inline]
```

10.69.4.61 set_recording() [1/2]

```
void seq64::perform::set_recording (
    bool record_active,
    bool thru_active,
    sequence * s )
```

Parameters

<i>record_active</i>	Provides the current status of the Record button.
<i>thru_active</i>	Provides the current status of the Thru button.
<i>s</i>	The sequence that the seqedit window represents. This pointer is checked.

10.69.4.62 set_recording() [2/2]

```
void seq64::perform::set_recording (
    bool record_active,
    int seq,
    bool toggle = false )
```

However, this function depends on the sequence, not the seqedit, for obtaining the thru status.

Parameters

<i>record_active</i>	Provides the current status of the Record button.
<i>seq</i>	The sequence number; the resulting pointer is checked.
<i>toggle</i>	If true, ignore the first flag and let the sequence toggle its setting. Passed along to sequence::set_input_recording() .

10.69.4.63 `set_quantized_recording()` [1/2]

```
void seq64::perform::set_quantized_recording (
    bool record_active,
    sequence * s )
```

Parameters

<i>record_active</i>	The setting desired for the quantized-recording flag.
<i>s</i>	Provides the pointer to the sequence to operate upon. Checked for validity.

10.69.4.64 `set_quantized_recording()` [2/2]

```
void seq64::perform::set_quantized_recording (
    bool record_active,
    int seq,
    bool toggle = false )
```

This isn't quite consistent with setting regular recording, which uses [sequence::set_input_recording\(\)](#).

Parameters

<i>record_active</i>	Provides the current status of the Record button.
<i>seq</i>	The sequence number; the resulting pointer is checked.
<i>toggle</i>	If true, ignore the first flag and let the sequence toggle its setting. Passed along to sequence::set_input_recording() .

10.69.4.65 `set_overwrite_recording()`

```
void seq64::perform::set_overwrite_recording (
    bool overwrite_active,
    int seq,
    bool toggle = false )
```

Todo Might probably as well create(`bool rec_active, bool thru_active, sequence * s`).

Pull request #150:

Ask for a reset explicitly upon toggle-on, since we don't have the GUI to control for progress.

Parameters

<i>overwrite_active</i>	Provides the current status of the overwrite mode.
<i>seq</i>	The sequence number; the resulting pointer is checked.
<i>toggle</i>	If true, ignore the first flag and let the sequence toggle its setting. Passed along to sequence::set_overwrite_rec() .

10.69.4.66 set_thru() [1/2]

```
void seq64::perform::set_thru (
    bool record_active,
    bool thru_active,
    sequence * s )
```

Parameters

<i>record_active</i>	Provides the current status of the Record button.
<i>thru_active</i>	Provides the current status of the Thru button.
<i>s</i>	The sequence that the seqedit window represents. This pointer is checked.

10.69.4.67 set_thru() [2/2]

```
void seq64::perform::set_thru (
    bool thru_active,
    int seq,
    bool toggle = false )
```

However, this function depends on the sequence, not the seqedit, for obtaining the recording status.

Parameters

<i>thru_active</i>	Provides the current status of the Thru button.
<i>seq</i>	The sequence number; the resulting pointer is checked.
<i>toggle</i>	If true, ignore the first flag and let the sequence toggle its setting. Passed along to sequence::set_input_thru() .

10.69.4.68 selected_trigger()

```
bool seq64::perform::selected_trigger (
    int seqnum,
    midipulse droptick,
    midipulse & tick0,
    midipulse & tick1 )
```

The more code moved out of the user-interface, the better.

Parameters

	<i>seqnum</i>	The number of the sequence of interest.
	<i>droptick</i>	The tick location, basically where the mouse was clicked.
out	<i>tick0</i>	The output location for the start of the trigger.
out	<i>tick1</i>	The output location for the end of the trigger.

Returns

Returns true if the sequence is valid and we can select the trigger.

10.69.4.69 clear_all()

```
bool seq64::perform::clear_all ( )
```

The mainwnd module calls this function. Note that perform now handles the "is modified" flag on behalf of all external objects, to centralize and simplify the dirtying of a MIDI tune.

Anything else to clear? What about all the other sequence flags? We can beef up [delete_sequence\(\)](#) for them, at some point.

Added stazed code from 1.0.5 to abort clearing if any of the sequences are in editing.

Returns

Returns true if the clear-all operation could be performed. If false, then at least one active sequence was in editing mode.

10.69.4.70 launch()

```
void seq64::perform::launch (
    int ppqn )
```

This function is called in main(). We collected all the calls here as a simplification, and renamed it because it is more than just initialization. This function must be called after the perform constructor and after the configuration file and command-line configuration overrides. The original implementation, where the master buss was an object, was too inflexible to handle a JACK implementation.

Parameters

<i>ppqn</i>	Provides the PPQN value, which is either the default value (192) or is read from the "user" configuration file.
-------------	---

Todo We probably need a bpm parameter for consistency at some point.

10.69.4.71 finish()

```
void seq64::perform::finish ( )
```

A minor simplification for the main() routine, hides the JACK support macro. We might need to add code to stop any ongoing outputting.

Also gets the settings made/changed while the application was running from the mastermidibase class to here. This action is the converse of calling the set_port_statuses() function defined in the mastermidibase module.

10.69.4.72 new_sequence()

```
void seq64::perform::new_sequence (
    int seq )
```

Then it activates the pattern [this is done in the [install_sequence\(\)](#) function]. It doesn't deal with thrown exceptions.

This function is called by the seqmenu and mainwid objects to create a new sequence. We now pass this sequence to [install_sequence\(\)](#) to better handle potential memory leakage, and to make sure the sequence gets counted. Also, adding a new sequence from the user-interface is a significant modification, so the "is modified" flag gets set.

Change Note ca 2016-05-15 If enabled, wire in the MIDI buss override.

Parameters

<i>seq</i>	The prospective sequence number of the new sequence.
------------	--

10.69.4.73 add_sequence()

```
void seq64::perform::add_sequence (
    sequence * seq,
    int prefnum )
```

No check is made for a null pointer, but the [install_sequence\(\)](#) call will make sure such a pointer is officially logged.

This function checks for the preferred sequence number. This is the number that was specified by the Sequence Number meta-event for the current track. If the preferred sequence number is in the valid range (0 to m_sequence↵_max) and it is not active, add it and activate it. Otherwise, iterate through all patterns from prefnum to m_↵sequence_max and add and activate the first one that is not active, and then finish.

Finally, note that this function is used only by midifile, when reading in a MIDI song. Therefore, the "is modified" flag is *not* set by this function; loading a sequence from a file is not a modification that should lead to a prompt for saving the file later.

Todo Shouldn't we wrap around the sequence list if we can't find an empty sequence slot after *prefnum*?

Todo This function needs some deeper analysis against the original, in my opinion.

Warning

The logic of the if-statement in this function was such that *prefnum* could be out-of-bounds in the else-clause. We reworked the logic to be airtight. This bug was caught by gcc 4.8.3 on CentOS, but not on gcc 4.9.3 on Debian Sid!

Parameters

<i>seq</i>	The pointer to the pattern/sequence to add.
<i>prefnum</i>	The preferred sequence number of the pattern, as explained above. If this value is out-of-range, then it is basically ignored.

10.69.4.74 delete_sequence()

```
void seq64::perform::delete_sequence (
    int seq )
```

We now also solidify the deletion by setting the pointer to null after deletion, so it will blow up if accidentally accessed. The final act is to raise the "is modified" flag, since deleting an existing sequence is always a significant modification.

Now, this function obviously sets the "active" flag for the sequence to false. But there are a few other flags that are not modified; shouldn't we also falsify them here?

Parameters

<i>seq</i>	The sequence number of the sequence to be deleted. It is validated.
------------	---

10.69.4.75 is_sequence_in_edit()

```
bool seq64::perform::is_sequence_in_edit (
    int seq )
```

Parameters

<i>seq</i>	Provides the sequence number to be checked.
------------	---

Returns

Returns `true` if the sequence's `get_editing()` call returns `true`. Otherwise, `false` is returned, which can also indicate an illegal sequence number.

10.69.4.76 print_busses()

```
void seq64::perform::print_busses ( ) const
```

10.69.4.77 get_tick()

```
midipulse seq64::perform::get_tick ( ) const [inline]
```

10.69.4.78 set_tick()

```
void seq64::perform::set_tick (
    midipulse tick )
```

Todo Do we really need `m_current_tick`???

10.69.4.79 get_jack_tick()

```
midipulse seq64::perform::get_jack_tick ( ) const [inline]
```

10.69.4.80 set_jack_tick()

```
void seq64::perform::set_jack_tick (
    midipulse tick ) [inline]
```

Parameters

<i>tick</i>	Provides the current JACK tick (pulse) value to set.
-------------	--

10.69.4.81 set_left_tick()

```
void seq64::perform::set_left_tick (
    midipulse tick,
    bool setstart = true )
```

We let the caller determine if this setting is a modification. If the left tick is later than the right tick, the right tick is move to one measure past the left tick.

Todo The `perform::m_one_measure` member is currently hardwired to `m_ppqn*4`.

Parameters

<i>tick</i>	The tick (MIDI pulse) at which to place the left tick. If the left tick is greater than or equal to the right tick, then the right ticked is moved forward by one "measure's length" (<code>m_ppqn * 4</code>) past the left tick.
<i>setstart</i>	If true (the default, and long-standing implicit setting), then the starting tick is also set to the left tick.

10.69.4.82 get_left_tick()

```
midipulse seq64::perform::get_left_tick ( ) const [inline]
```

10.69.4.83 set_start_tick()

```
void seq64::perform::set_start_tick (
    midipulse tick ) [inline]
```

Parameters

<i>tick</i>	Provides the starting JACK tick (pulse) value to set.
-------------	---

10.69.4.84 get_start_tick()

```
midipulse seq64::perform::get_start_tick ( ) const [inline]
```

10.69.4.85 set_right_tick()

```
void seq64::perform::set_right_tick (
    midipulse tick,
    bool setstart = true )
```

This setting is made only if the tick parameter is at or beyond the first measure. We let the caller determine if this setting is a modification.

Parameters

<i>tick</i>	The tick (MIDI pulse) at which to place the right tick. If less than or equal to the left tick setting, then the left tick is backed up by one "measure's worth" ($m_ppqn * 4$) worth of ticks from the new right tick.
<i>setstart</i>	If true (the default, and long-standing implicit setting), then the starting tick is also set to the left tick, if that got changed.

10.69.4.86 `get_right_tick()`

```
midipulse seq64::perform::get_right_tick ( ) const [inline]
```

10.69.4.87 `left_right_size()`

```
double seq64::perform::left_right_size ( ) const [inline]
```

Returns

Returns the difference between the right and left tick, cast to double.

10.69.4.88 `is_active()`

```
bool seq64::perform::is_active (
    int seq ) const [inline]
```

Parameters

<i>seq</i>	The pattern number. It is checked for invalidity. This can lead to "too many" (i.e. redundant) checks, but we're trying to centralize such checks in this function.
------------	---

Returns

Returns the value of the active-flag, or false if the sequence was invalid or null.

10.69.4.89 `apply_song_transpose()`

```
void seq64::perform::apply_song_transpose ( )
```


10.69.4.90 set_transpose()

```
void seq64::perform::set_transpose (
    int t ) [inline]
```

10.69.4.91 get_transpose()

```
int seq64::perform::get_transpose ( ) const [inline]
```

10.69.4.92 get_beats_per_minute()

```
midibpm seq64::perform::get_beats_per_minute ( ) [inline]
```

This result should be the same as the value of the `m_bpm` member. This function returns that value in a roundabout way.

Returns

Returns the value of beats/minute from the master buss.

10.69.4.93 reload_mute_groups()

```
bool seq64::perform::reload_mute_groups (
    std::string & errmsg )
```

Parameters

<i>errmsg</i>	A pass-back parameter for any error message the file-processing might cause.
---------------	--

Returns

Returns true if the reload succeeded.

10.69.4.94 clear_mute_groups()

```
bool seq64::perform::clear_mute_groups ( )
```

Side-effect(s) If true is returned, the modify flag is set, so that the user has the option to save a MIDI file that contained mute-groups that are no longer wanted.

Returns

Returns true if any of the statuses changed from true to false.

10.69.4.95 set_sequence_control_status()

```
void seq64::perform::set_sequence_control_status (
    int status )
```

Then the given status is OR'd into the m_control_status.

Parameters

<i>status</i>	The status to be used.
---------------	------------------------

10.69.4.96 unset_sequence_control_status()

```
void seq64::perform::unset_sequence_control_status (
    int status )
```

Then the given status is reversed in m_control_status.

If the given status includes c_status_queue, this is a signal to stop queuing (which is already in place elsewhere). It also unsets the new queue-replace feature.

Parameters

<i>status</i>	The status to be used.
---------------	------------------------

10.69.4.97 unset_queued_replace()

```
void seq64::perform::unset_queued_replace (
    bool clearbits = true )
```

This also clears the queue mode; we shall see if this disrupts any user's workflow.

Parameters

<i>clearbits</i>	If true (the default), then clear the queue and replace status bits. If the user is simply replacing the current replace pattern with another pattern, we pass false for this parameter.
------------------	--

10.69.4.98 sequence_playing_toggle()

```
void seq64::perform::sequence_playing_toggle (
    int seq )
```

If `m_control_status` is `c_status_queue`, then the sequence's `toggle_queued()` function is called. This is the "mod queue" implementation.

Otherwise, if it is `c_status_replace`, then the status is unset, and all sequences are turned off. Then the sequence's `toggle_playing()` function is called, which should turn it back on. This is the "mod replace" implementation; it is like a Solo. But can it be undone?

This function is called in [sequence_key\(\)](#) to implement a toggling of the sequence of the pattern slot in the current screen-set that is represented by the keystroke.

This function is also called in [midi_control_event\(\)](#) if the control number represents a sequence number in a screen-set, that is, it ranges from 0 to

1. This value should be offset by the current screen-set number, `m_screenset_offset`, before passing it to this function.

This function now also supports the new queued-replace (queued-solo) feature.

Parameters

<i>seq</i>	The sequence number of the sequence to be potentially toggled. This value must be a valid and active sequence number. If in queued-replace mode, and if this pattern number is different from the currently-stored number (<code>m_queued_replace_slot</code>), then we clear the currently stored set of patterns and set new stored patterns.
------------	---

10.69.4.99 sequence_playing_change()

```
void seq64::perform::sequence_playing_change (
    int seq,
    bool on )
```

Used for the implementation of [sequence_playing_on\(\)](#) and [sequence_playing_off\(\)](#).

Kepler34's version seems slightly different, may need more study.

Parameters

<i>seq</i>	The number of the sequence to be turned off.
<i>on</i>	True if the sequence is to be turned on, false if it is to be turned off.

10.69.4.100 set_keep_queue()

```
void seq64::perform::set_keep_queue (
    bool activate )
```

10.69.4.101 is_keep_queue()

```
bool seq64::perform::is_keep_queue ( ) const
```

10.69.4.102 sequence_playing_on()

```
void seq64::perform::sequence_playing_on (
    int seq ) [inline]
```

Parameters

<i>seq</i>	The sequence number of the sequence to turn on.
------------	---

10.69.4.103 sequence_playing_off()

```
void seq64::perform::sequence_playing_off (
    int seq ) [inline]
```

Parameters

<i>seq</i>	The sequence number of the sequence to turn off.
------------	--

10.69.4.104 mute_all_tracks()

```
void seq64::perform::mute_all_tracks (
    bool flag = true )
```

Covers tracks from 0 to `m_sequence_max`.

We have to also set the sequence's playing status, in opposition to the mute status, in order to see the sequence status change on the user-interface. HMMMMMMM.

Parameters

<i>flag</i>	If true (the default), the song-mute of the sequence is turned on. Otherwise, it is turned off.
-------------	---

10.69.4.105 toggle_all_tracks()

```
void seq64::perform::toggle_all_tracks ( )
```

Covers tracks from 0 to m_sequence_max.

Note that toggle_playing() now has two default parameters used by the new song-recording feature, which are currently not used here.

10.69.4.106 armed_saved()

```
bool seq64::perform::armed_saved ( ) const [inline]
```

10.69.4.107 toggle_playing_tracks()

```
void seq64::perform::toggle_playing_tracks ( )
```

Covers tracks from 0 to m_sequence_max. The statuses are preserved for restoration.

Note that this function operates only in Live mode; it is too confusing to use in Song mode. Also note that toggle_playing() now has two default parameters used by the new song-recording feature, which are currently not used here.

10.69.4.108 mute_screenset()

```
void seq64::perform::mute_screenset (
    int ss,
    bool flag = true )
```

Parameters

<i>ss</i>	The screen-set to be operated upon.
<i>flag</i>	If true (the default), the song-mute of the sequence is turned on. Otherwise, it is turned off.

10.69.4.109 output_func()

```
void seq64::perform::output_func ( )
```

This function is called by the free function `output_thread_func()`. Here's how it works:

- It runs while `m_outputting` is true.
- MORE TO COME. Yeah, a lot more to come. It is a complex function.

Change Note ca 2016-01-26 Hurray, seq24 is coming back to life! We see that there is a fix for clock tick drift here, which relies on using long and long long values. See the Changelog for seq24 0.9.3.

Warning

Valgrind shows that `output_func()` is being called before the JACK client pointer is being initialized!!!

1. Get delta time (current - last).
2. Get delta ticks from time.
3. Add to `current_ticks`.
4. Compute prebuffer ticks.
5. Play from current tick to prebuffer.

Figure out how much time we need to sleep, and do it.

Now we want to trigger every `c_thread_trigger_width_us`, and it took us `delta_us` to `play()`. Also known as the "sleeping_us".

Check MIDI clock adjustment. Note that we replaced "`60000000.0f / m_ppqn / bpm`" with a call to a function. We also removed the "f" specification from the constants.

10.69.4.110 `input_func()`

```
void seq64::perform::input_func ( )
```

It handles certain MIDI input events.

Stazed:

<http://www.blitter.com/~russtopia/MIDI/~jglatt/tech/midispec/ssp.htm>

Example: If a Song Position value of 8 is received, then a sequencer (or drum box) should cue playback to the third quarter note of the song. (8 MIDI beats * 6 MIDI clocks per MIDI beat = 48 MIDI Clocks. Since there are 24 MIDI Clocks in a quarter note, the first quarter occurs on a time of 0 MIDI Clocks, the second quarter note occurs upon the 24th MIDI Clock, and the third quarter note occurs on the 48th MIDI Clock).

8 MIDI beats * 6 MIDI clocks per MIDI beat = 48 MIDI Clocks.

EVENT_MIDI_START:

Starts the MIDI Time Clock. Kepler34 does "stop(); set_playback_mode(false); start();" in its version of this event. This sets the playback mode to Live mode. This behavior seems reasonable, though function names Sequencer64 uses are different.

EVENT_MIDI_CONTINUE:

MIDI continue: start from current position. This is sent immediately after EVENT_MIDI_SONG_POS, and is used for starting from other than beginning of the song, or for starting from previous location at EVENT_MIDI_STOP. Again, converted to Kepler34 mode of setting the playback mode to Live mode.

EVENT_MIDI_STOP:

Do nothing, just let the system pause. Since we're not getting ticks after the stop, the song won't advance when start is received, we'll reset the position. Or, when continue is received, we won't reset the position. We do an inner_stop(); the m_midiclockpos member holds the stop position in case the next event is "continue". This feature is not in Kepler34.

EVENT_MIDI_CLOCK:

MIDI beat clock (MIDI timing clock or simply MIDI clock) is a clock signal broadcast via MIDI to ensure that MIDI-enabled devices stay in synchronization. It is not MIDI timecode. Unlike MIDI timecode, MIDI beat clock is tempo-dependent. Clock events are sent at a rate of 24 ppqn (pulses per quarter note). Those pulses maintain a synchronized tempo for synthesizers that have BPM-dependent voices and for arpeggiator synchronization. Location information can be specified using MIDI Song Position Pointer. Many simple MIDI devices ignore this message.

EVENT_MIDI_SONG_POS:

MIDI song position pointer message tells a MIDI device to cue to a point in the MIDI sequence to be ready to play. This message consists of three bytes of data. The first byte, the status byte, is 0xF2 to flag a song position pointer message. Two bytes follow the status byte. These two bytes are combined in a 14-bit value to show the position in the song to cue to. The top bit of each of the two bytes is not used. Thus, the value of the position to cue to is between 0x0000 and 0x3FFF. The position represents the MIDI beat, where a sequence always starts on beat zero and each beat is a 16th note. Thus, the device will cue to a specific 16th note. Also see the combine_bytes() function.

EVENT_MIDI_SYSEX:

These messages are system-wide messages. We filter system-wide messages. If the master MIDI buss is dumping, set the timestamp of the event and stream it on the sequence. Otherwise, use the event data to control the sequencer, if it is valid for that action.

"Dumping" is set when a seqedit window is open and the user has clicked the "record MIDI" or "thru MIDI" button. In this case, if the seq32 support is in force, dump to it, else stream the event, with possibly multiple sequences set. Otherwise, handle an incoming MIDI control event.

Also available (but macroed out) is Stazed's parse_sysex() function. It seems specific to certain Yamaha devices, but might prove useful later.

For events less than or equal to SysEx, we call [midi_control_event\(\)](#) to handle the MIDI controls that Sequencer64 supports. (These are configurable in the "rc" configuration file.)

10.69.4.111 set_group_mute_state()

```
void seq64::perform::set_group_mute_state (
    int gtrack,
    bool muted )
```

The index value is the track number offset by the number of the selected mute group (which is equivalent to a set number) times the number of sequences in a set. This function is used in midfile and optionsfile when parsing the file to get the initial mute-groups.

Bug We were not using the group track value if it was zero, but that is a legitimate value.

Parameters

<i>gtrack</i>	The number of the track to be muted/unmuted.
<i>muted</i>	This boolean indicates the state to which the track should be set.

10.69.4.112 get_group_mute_state()

```
bool seq64::perform::get_group_mute_state (
    int gtrack )
```

Uses the [mute_group_offset\(\)](#) function. This function is used in midfile and optionsfile when writing the file to get the initial mute-groups.

Bug We were not using the group track value if it was zero, but that is a legitimate value.

Parameters

<i>gtrack</i>	The number of the track for which the state is to be obtained. Like set_group_mute_state() , this value is offset by adding <code>m_mute_group_selected * m_seqs_in_set</code> .
---------------	--

Returns

Returns the desired `m_mute_group[]` value.

10.69.4.113 mute_group_offset()

```
int seq64::perform::mute_group_offset (
    int trackoffset )
```

Remember that the mute-group array, `m_mute_group[c_max_sequence]`, determines which tracks are muted/unmuted. Also remember that `m_mute_group_selected` now determines which "seqs-in-set" set is selected.

Old definition:


```
return clamp_track(track) + m_mute_group_selected * m_seqs_in_set;
```

Parameters

<i>trackoffset</i>	The number of the desired track. This is basically one of the hot-key related values. Traditionally, this value ranges from 0 to 31 (c_seqs_in_set-1), but now the variset mode is supported.
--------------------	---

Returns

Returns a track value from 0 to 1023 if the group is valid for the current seqs-in-set count and a mute-group has been selected. Otherwise, a SEQ64_NO_MUTE_GROUP_SELECTED (-1) is returned. The caller must check this value before using it.

10.69.4.114 screenset_offset() [1/2]

```
int seq64::perform::screenset_offset ( ) const [inline]
```

10.69.4.115 slot_number()

```
int seq64::perform::slot_number (
    int s ) [inline]
```

Parameters

<i>s</i>	Provides the sequence number of interest. This value should range from 0 to 1023 (c_max_seqs).
----------	--

Returns

Returns the "normalized" value. Do not use it if less than zero.

10.69.4.116 save_playing_state()

```
void seq64::perform::save_playing_state ( )
```

Inactive patterns get the value set to false. Used in unsetting the snapshot status (c_status_snapshot).

10.69.4.117 restore_playing_state()

```
void seq64::perform::restore_playing_state ( )
```

Used in unsetting the snapshot status (c_status_snapshot).

10.69.4.118 save_current_screenset()

```
void seq64::perform::save_current_screenset (
    int repseq )
```

Inactive patterns get the value set to false. Used in saving the screen-set state during the queued-replace (queued-sol) operation, which occurs when the c_status_replace is performed while c_status_queue is active.

Parameters

<i>repseq</i>	Provides the number of the pattern for which the replace functionality is invoked. This pattern will set to "playing" whether it is on or off, so that it can stay active while toggling between "solo" and "playing with the rest of the patterns".
---------------	--

10.69.4.119 `clear_current_screenset()`

```
void seq64::perform::clear_current_screenset ( )
```

Needed when disabling the queue mode.

10.69.4.120 `key_name()`

```
std::string seq64::perform::key_name (
    unsigned k ) const [inline]
```

Parameters

<i>k</i>	The key number for which to return the string name of the key.
----------	--

10.69.4.121 `get_key_events()`

```
keys_perform::SlotMap& seq64::perform::get_key_events ( ) [inline]
```

10.69.4.122 `get_key_count()`

```
int seq64::perform::get_key_count (
    unsigned k ) const [inline]
```

Parameters

<i>k</i>	The key value to be checked.
----------	------------------------------

10.69.4.123 `get_key_groups()`

```
keys_perform::SlotMap& seq64::perform::get_key_groups ( ) [inline]
```

10.69.4.124 get_key_events_rev()

```
keys_perform::RevSlotMap& seq64::perform::get_key_events_rev ( ) [inline]
```

10.69.4.125 get_key_groups_rev()

```
keys_perform::RevSlotMap& seq64::perform::get_key_groups_rev ( ) [inline]
```

10.69.4.126 show_ui_sequence_key() [1/2]

```
bool seq64::perform::show_ui_sequence_key ( ) const [inline]
```

Used in mainwid, options, optionsfile, userfile, and perform.

10.69.4.127 show_ui_sequence_key() [2/2]

```
void seq64::perform::show_ui_sequence_key (
    bool flag ) [inline]
```

Parameters

<i>flag</i>	Provides the flag to set into keys().show_ui_sequence_key() .
-------------	---

10.69.4.128 show_ui_sequence_number() [1/2]

```
bool seq64::perform::show_ui_sequence_number ( ) const [inline]
```

Used in mainwid, optionsfile, and perform.

10.69.4.129 show_ui_sequence_number() [2/2]

```
void seq64::perform::show_ui_sequence_number (
    bool flag ) [inline]
```

Parameters

<i>flag</i>	Provides the value to set into keys().show_ui_sequence_number() .
-------------	---

10.69.4.130 lookup_keyevent_key()

```
unsigned seq64::perform::lookup_keyevent_key (
    int seqnum )
```

If we're not in legacy mode, then we adjust for the screenset, so that screensets greater than 0 can also show the correct key name, instead of a question mark (or blank).

Legacy seq24 already responds to the toggling of the mute state via the shortcut keys even if screenset > 0, but it shows the question mark.

Todo In the context of pattern keys, we should replace `c_seqs_in_set` with a better-named value; if sets are actually larger than that, due to the "sets" option, then we simply repeat the pattern here using a modifier key ["shifted", see [lookup_slot_key\(\)](#)]; in other words, we're stuck on using 32 pattern hot-keys.

Parameters

<i>seqnum</i>	The number of the sequence for which to return the event key.
---------------	---

Returns

Returns the desired key. If there is no such value, then the space (' ') character is returned. It used to be the question mark.

10.69.4.131 lookup_slot_key()

```
unsigned seq64::perform::lookup_slot_key (
    int slot )
```

Parameters

<i>slot</i>	The number of the pattern/sequence for which to return the event key. This value can range from 0 to <code>c_seqs_in_set - 1</code> up to <code>(3 * c_seqs_in_set) - 1</code> , since we can support 32 hotkeys, plus these hot-keys "shifted" once and twice. This value is relative to <code>m_screenset_offset</code> , and then is modded re <code>c_seqs_in_set</code> , so that it always ranges from 0 to 31.
-------------	---

Returns

Returns the desired key. This will always work, due to the mod operation.

10.69.4.132 lookup_keyevent_seq()

```
int seq64::perform::lookup_keyevent_seq (
    unsigned keycode ) [inline]
```

The inverse of [lookup_keyevent_key\(\)](#).

Parameters

<i>keycode</i>	The number of the event key for which to return the configured sequence number.
----------------	---

Returns

Returns the desired sequence. If there is no such value, then a sequence number of 0 is returned.

10.69.4.133 lookup_keygroup_key()

```
unsigned seq64::perform::lookup_keygroup_key (  
    int groupnum ) [inline]
```

Parameters

<i>groupnum</i>	The number of the group for which to return the group key.
-----------------	--

Returns

Returns the desired key. If there is no such value, then the default character is returned.

10.69.4.134 lookup_keygroup_group()

```
int seq64::perform::lookup_keygroup_group (  
    unsigned keycode ) [inline]
```

The inverse of [lookup_keygroup_key\(\)](#).

Parameters

<i>keycode</i>	The number of the group key for which to return the configured sequence number.
----------------	---

Returns

Returns the desired group number. If there is no such value, then a group number of 0 is returned.

10.69.4.135 start_playing()

```
void seq64::perform::start_playing (  
    bool songmode = false )
```

We've reversed the `start()` and `start_jack()` calls so that JACK is started first, to match all of the other use-cases for playing that we've found in the code. Note that the complementary function, `stop_playing()`, is an inline function defined in the header file.

The `perform::start()` function passes its boolean flag to `perform::inner_start()`, which sets the playback mode to that flag; if that flag is false, that turns off "song" mode. So that explains why mute/unmute is disabled.

Playback use cases:

These use cases are meant to apply to either a Seq32 or a regular build of Sequencer64, eventually. Currently, the regular build does not have a concept of a "global" perform song-mode flag.

```
-# mainwnd.
-# Play. If the perform song-mode is "Song", then use that mode.
    Otherwise, use "Live" mode.
-# Stop. This action is modeless here. In ALSA, it will cause
    a rewind (but currently seqroll doesn't rewind until Play is
    clicked, a minor bug).
-# Pause. Same processing as Play or Stop, depending on current
    status. When stopping, the progress bars in seqroll and
    perfroll remain at their current point.
-# perfedit.
-# Play. Override the current perform song-mode to use "Song".
-# Stop. Revert the perfedit setting, in case play is restarted
    or resumed via mainwnd.
-# Pause. Same processing as Play or Stop, depending on current
    status.
-# ALSA versus JACK. One issue here is that, if JACK isn't "running"
    at all (i.e. we are in ALSA mode), then we cannot be JACK Master.
```

Helgrind shows a read/write race condition in `m_start_from_perfedit` between `jack_transport_callback()` and `start_playing()` here. Is inline function access of a boolean atomic?

Parameters

<i>songmode</i>	Indicates if the caller wants to start the playback in Song mode (sometimes erroneously referred to as "JACK mode"). In the seq32 code at GitHub, this flag was identical to the "global_jack_start_mode" flag, which is true for Song mode, and false for Live mode. False disables Song mode, and is the default, which matches seq24. Generally, we pass true in this parameter if we're starting playback from the perfedit window. It alters the <code>m_start_from_perfedit</code> member, not the <code>m_song_start_mode</code> member (which replaces the global flag now).
-----------------	--

10.69.4.136 `pause_playing()`

```
void seq64::perform::pause_playing (
    bool songmode = false )
```

Currently almost the same as `stop_playing()`, but expanded as noted in the comments so that we ultimately have more granular control over what happens. We're researching the whole sequence of stopping and starting, and it can be tricky to make correct changes.

We still need to make restarting pick up at the same place in ALSA mode; in JACK mode, JACK transport takes care of that feature.

Change Note ca 2016-10-11 User layk noted this call, and it makes sense to not do this here, since it is unknown at this point what the actual status is. Note that we STILL need to FOLLOW UP on calls to `pause_playing()` and `stop_playing()` in perfedit, mainwnd, etc.

is_pattern_playing(false);

But what about [is_running\(\)](#)?

Parameters

<i>songmode</i>	Indicates that, if resuming play, it should play in Song mode (true) or Live mode (false). See the comments for the start_playing() function.
-----------------	---

10.69.4.137 stop_playing()

```
void seq64::perform::stop_playing ( )
```

Stops playback, turns off the (new) m_dont_reset_ticks flag, and set the "is-pattern-playing" flag to false. With stop, reset the start-tick to either the left-tick or the 0th tick (to be determined, currently resets to 0).

10.69.4.138 start_key()

```
void seq64::perform::start_key (
    bool songmode = false )
```

Meant to be used by GUIs to unify the treatment of keys versus buttons. Also handy in the extended MIDI controls that people have requested.

Parameters

<i>songmode</i>	The live/play mode parameter to be passed along to the key processor. Defaults to false (live mode).
-----------------	--

10.69.4.139 pause_key()

```
void seq64::perform::pause_key (
    bool songmode = false )
```

Meant to be used by GUIs to unify the treatment of keys versus buttons. Also handy in the extended MIDI controls that people have requested.

Parameters

<i>songmode</i>	The live/play mode parameter to be passed along to the key processor, when starting playback. Defaults to false (live mode).
-----------------	--

10.69.4.140 stop_key()

```
void seq64::perform::stop_key ( )
```

Meant to be used by GUIs to unify the treatment of keys versus buttons. Also handy in the extended MIDI controls that people have requested.

10.69.4.141 learn_toggle()

```
void seq64::perform::learn_toggle ( ) [inline]
```

10.69.4.142 decrement_beats_per_minute()

```
midibpm seq64::perform::decrement_beats_per_minute ( )
```

Actually does a lot of work in those function calls.

Returns

Returns the resultant BPM, as a convenience.

10.69.4.143 increment_beats_per_minute()

```
midibpm seq64::perform::increment_beats_per_minute ( )
```

Actually does a lot of work in those function calls.

Returns

Returns the resultant BPM, as a convenience.

10.69.4.144 page_decrement_beats_per_minute()

```
midibpm seq64::perform::page_decrement_beats_per_minute ( )
```

Encapsulates some calls used in mainwnd. Actually does a lot of work in those function calls.

Returns

Returns the resultant BPM, as a convenience.

10.69.4.145 page_increment_beats_per_minute()

```
midibpm seq64::perform::page_increment_beats_per_minute ( )
```

Encapsulates some calls used in mainwnd. Actually does a lot of work in those function calls.

Returns

Returns the resultant BPM, as a convenience.

10.69.4.146 decrement_screenset()

```
int seq64::perform::decrement_screenset (
    int amount = 1 )
```

The value set here will represent the "active" screen-set in multi-window mode.

Parameters

<i>amount</i>	Indicates the amount the screenset is to be decremented. The default value is 1.
---------------	--

Returns

Returns the decremented screen-set value.

10.69.4.147 increment_screenset()

```
int seq64::perform::increment_screenset (
    int amount = 1 )
```

The value set here will represent the "active" screen-set in multi-window mode.

Parameters

<i>amount</i>	Indicates the amount the screenset is to be incremented. The default value is 1.
---------------	--

Returns

Returns the incremented screen-set value.

10.69.4.148 highlight()

```
bool seq64::perform::highlight (
    const sequence & seq ) const [inline]
```

This setting is currently a build-time option, but could be made a run-time option later.

Parameters

seq	Provides a reference to the desired sequence.
---------------------	---

10.69.4.149 is_smf_0()

```
bool seq64::perform::is_smf_0 (
    const sequence & seq ) const [inline]
```

Parameters

seq	Provides a reference to the desired sequence.
---------------------	---

10.69.4.150 get_sequence() [1/2]

```
const sequence* seq64::perform::get_sequence (
    int seq ) const [inline]
```

This is the const version. Note that it is more efficient to call this function and check the result than to call [is_active\(\)](#) and then call this function.

Parameters

seq	The prospective sequence number.
---------------------	----------------------------------

Returns

Returns the value of `m_seqs[seq]` if `seq` is valid. Otherwise, a null pointer is returned.

10.69.4.151 get_sequence() [2/2]

```
sequence* seq64::perform::get_sequence (
    int seq ) [inline]
```

This is the non-const version. Note that it is more efficient to call this function and check the result than to call [is_active\(\)](#) and then call this function.

Parameters

<code>seq</code>	The prospective sequence number.
------------------	----------------------------------

Returns

Returns the value of `m_seqs[seq]` if `seq` is valid. Otherwise, a null pointer is returned.

10.69.4.152 `sequence_key()`

```
void seq64::perform::sequence_key (
    int seq )
```

This function is use in `mainwnd` when toggling the mute/unmute setting using keyboard keys.

Parameters

<code>seq</code>	The sequence's control-key number, which is relative to the current screen-set.
------------------	---

10.69.4.153 `sequence_label()` [1/2]

```
std::string seq64::perform::sequence_label (
    const sequence & seq )
```

This string goes on the bottom-left of those user-interface elements.

The format of this string is something like the following example, depending on the "show sequence numbers" option. The values shown are, in this order, sequence number (if allowed), buss number, channel number, beats per bar, and beat width.

```
No sequence number:    31-16 4/4
Sequence number:      9  31-16 4/4
```

The sequence number and buss number are re 0, while the channel number is displayed re 1, unless it is an SMF 0 null channel (0xFF), in which case it is 0.

```
"%-3d%-3d-%d %d/%d" (old)
```

Note

Later, we could add the sequence hot-key to this string, though showing that is not much use in pernames. Also, this function is a stilted mix of direct access and access through sequence number.

Parameters

<i>seq</i>	Provides the reference to the sequence, use for getting the sequence parameters to be written to the label string.
------------	--

Returns

Returns the filled in label if the sequence is active. Otherwise, an empty string is returned.

10.69.4.154 sequence_label() [2/2]

```
std::string seq64::perform::sequence_label (  
    int seqnum )
```

Parameters

<i>seq</i>	Provides the reference to the sequence, use for getting the sequence parameters to be written to the label string.
------------	--

Returns

Returns the filled in label if the sequence is active. Otherwise, an empty string is returned.

10.69.4.155 sequence_title()

```
std::string seq64::perform::sequence_title (  
    const sequence & seq )
```

This title is used in the slots to show the (possibly shortened) pattern title. Note that the sequence title will also show the sequence length, in measures, if the [show_ui_sequence_key\(\)](#) option is active.

Parameters

<i>seq</i>	Provides the reference to the sequence, use for getting the sequence parameters to be written to the label string.
------------	--

Returns

Returns the filled in label if the sequence is active. Otherwise, an empty string is returned.

10.69.4.156 `main_window_title()`

```
std::string seq64::perform::main_window_title ( )
```

Unlike the disabled code in [mainwnd::update_window_title\(\)](#), this code does not (yet) handle conversions to UTF-8.

Returns

Returns the filled-in main window title.

10.69.4.157 `sequence_window_title()`

```
std::string seq64::perform::sequence_window_title (
    const sequence & seq )
```

Parameters

<i>seq</i>	Provides the reference to the sequence, use for getting the sequence parameters to be written to the string.
------------	--

Returns

Returns the filled in label if the sequence is active. Otherwise, an incomplete string is returned.

10.69.4.158 `set_input_bus()`

```
void seq64::perform::set_input_bus (
    bussbyte bus,
    bool active )
```

This function is called by [options::input_callback\(\)](#).

Note that the [mastermidibus::set_input\(\)](#) function passes the setting along to the input busarray.

Tricky Code See the bus parameter. We should provide two separate functions for this feature, but it is already combined into one input-callback function with a lot of other functionality in the options module.

Parameters

<i>bus</i>	If this value is greater than SEQ64_DEFAULT_BUSS_MAX (32), then it is treated as a user-interface flag (PERFORM_KEY_LABELS_ON_SEQUENCE or PERFORM_NUM_LABELS_ON_SEQUENCE) that causes all the sequences to be dirtied, and thus get redrawn with the new user-interface setting.
<i>active</i>	Indicates whether the buss or the user-interface feature is active or inactive.

10.69.4.159 set_clock_bus()

```
void seq64::perform::set_clock_bus (
    bussbyte bus,
    clock_e clocktype )
```

Note that the call to [mastermidibus::set_clock\(\)](#) also sets the clock in the output busarray.

Parameters

<i>bus</i>	The bus index to be set.
<i>clocktype</i>	Indicates whether the buss or the user-interface feature is e_clock_off, e_clock_pos, e_clock_mod, or (new) e_clock_disabled.

10.69.4.160 mainwnd_key_event()

```
bool seq64::perform::mainwnd_key_event (
    const keystroke & k )
```

This function handles the keys for the functions of replace, queue, keep-queue, snapshots, toggling mute groups, group learn, and playing screenset. For further keystroke processing, see `mainwnd::on_key_press_event()`.

Keys not handled here are handled in `mainwnd`, where GUI elements exist to manage these items:

- bpm up & down
- screenset up & down
- mute group key
- mute group learn

seq24 handles the following keys in two "on_key" events:

Release:

- Replace unset
- Queue unset
- Snapshot 1 and snapshot 2 unset
- Group learn unset

Press:

- BPM dn and BMP up *
- Replace set
- Queue and keep-queue set
- Snapshot 1 and snapshot 2 set
- Screen-set dn and screen-set up *
- Set playing screen-set
- Group on and group off
- Group learn set
- Select and mute the group *
- Start and stop keys *
- Pattern mute/unmute keys *

Note that the asterisk indicates we handle it elsewhere. Screen-set down and up are handled in `mainwnd` by calling [decrement_screenset\(\)](#) and [increment_screenset\(\)](#), and [mainwid::set_screenset\(\)](#). But that latter call is not made when MIDI control is in force, which might be an ISSUE.

Parameters

<i>k</i>	The keystroke object to be handled.
----------	-------------------------------------

Returns

Returns true if the key was handled.

10.69.4.161 keyboard_control_press()

```
bool seq64::perform::keyboard_control_press (
    unsigned key )
```

10.69.4.162 keyboard_group_c_status_press()

```
bool seq64::perform::keyboard_group_c_status_press (
    unsigned key )
```

- [xxxxxxxx]
- perform::c_status "events".
 - * c_status_replace.
 - * c_status_queue.
 - * c_status_snapshot.
 - * c_status_oneshot.
 - * Used by:
 - [mainwnd::on_key_press_event\(\)](#) [[perform::mainwnd_key_event\(\)](#)]
- perform groups
 - * On.
 - * Off.
 - * Learn.
 - * Used by:
 - [mainwnd::on_key_press_event\(\)](#) [[perform::mainwnd_key_event\(\)](#)]
- [perform::playback_key_event\(\)](#).
- [perform::set_playing_screensset\(\)](#).
 - * Start.
 - * Stop.
 - * Pause.
- GUI framework specific

10.69.4.163 keyboard_group_c_status_release()

```
bool seq64::perform::keyboard_group_c_status_release (
    unsigned key )
```

10.69.4.164 keyboard_group_press()

```
bool seq64::perform::keyboard_group_press (
    unsigned key )
```

10.69.4.165 keyboard_group_release()

```
bool seq64::perform::keyboard_group_release (
    unsigned key )
```

10.69.4.166 keyboard_group_action()

```
perform::action_t seq64::perform::keyboard_group_action (
    unsigned key )
```

10.69.4.167 perfroll_key_event()

```
bool seq64::perform::perfroll_key_event (
    const keystroke & k,
    int drop_sequence )
```

It handles the Ctrl keys for cut, copy, paste, and undo.

The "is modified" flag is raised if something is deleted, but we cannot yet handle the case where we undo all the changes. So, for now, we play it safe with the user, even if the user gets annoyed because he knows that he undid all the changes.

Parameters

<i>k</i>	The keystroke object to be handled.
<i>drop_sequence</i>	Provides the index of the sequence whose selected trigger is to be cut, copied, or pasted. Undo and redo are now supported.

Returns

Returns true if the key was handled.

10.69.4.168 playback_key_event()

```
bool seq64::perform::playback_key_event (
    const keystroke & k,
    bool songmode = false )
```

To be used in mainwnd, perfedit, and seqroll.

The start/end key may be the same key (e.g. Space) to allow toggling when the same key is mapped to both triggers.

Checking [is_running\(\)](#) may not work completely in JACK.

Change Note layk 2016-10-11 Issue #42 to prevent inadvertent step-edit in sequence :: stream_event(). We did it slightly different to save a little code; also found a spot that was missed.

Parameters

<i>k</i>	Provides the encapsulated keystroke to check.
<i>songmode</i>	Provides the "jack flag" needed by the mainwnd, seqroll, and perfedit windows. Defaults to false, which disables Song mode, and enables Live mode. But using Song mode seems to make the pause key not work in the performance editor.

Returns

Returns true if the keystroke matched the start, stop, or (new) pause keystrokes. Generally, no further keystroke processing is needed in this case.

10.69.4.169 clear_sequence_triggers()

```
void seq64::perform::clear_sequence_triggers (
    int seq )
```

Parameters

<i>seq</i>	Provides the desired sequence. The is_active() function validates this value.
------------	---

10.69.4.170 print_triggers()

```
void seq64::perform::print_triggers ( ) const
```

10.69.4.171 move_triggers()

```
void seq64::perform::move_triggers (
    bool direction )
```

Parameters

<i>direction</i>	Specifies the desired direction; false = left, true = right.
------------------	--

10.69.4.172 copy_triggers()

```
void seq64::perform::copy_triggers ( )
```

This copies the triggers between the L marker and R marker to the R marker.

10.69.4.173 push_trigger_undo()

```
void seq64::perform::push_trigger_undo (
    int track = SEQ64_ALL_TRACKS )
```

Too bad we cannot yet keep track of all the undoes for the sake of properly handling the "is modified" flag.

This function now has a new parameter. Not added to this function is the seemingly redundant undo-push the seq32 code does; is this actually a seq42 thing?

Also, there is still an issue with our undo-handling for a single track. See [pop_trigger_undo\(\)](#).

Parameters

<i>track</i>	A new parameter (found in the stazed seq32 code) that allows this function to operate on a single track. A parameter value of SEQ64_ALL_TRACKS (-1, the default) implements the original behavior.
--------------	--

10.69.4.174 pop_trigger_undo()

```
void seq64::perform::pop_trigger_undo ( )
```

Todo Look at seq32/src/perform.cpp and the perform :: push_trigger_undo(track) function, which has a track parameter that has a -1 values the supports all tracks. It requires two new vectors (one for undo, one for redo), two new flags (likewise). We've put this code in place, no longer macroed out, now permanent.

10.69.4.175 pop_trigger_redo()

```
void seq64::perform::pop_trigger_redo ( )
```

10.69.4.176 get_trigger_state()

```
bool seq64::perform::get_trigger_state (
    int seqnum,
    midipulse tick ) const
```

Parameters

<i>seqnum</i>	The index to the desired sequence.
<i>tick</i>	The time-location at which to get the trigger state.

Returns

Returns true if the sequence indicated by *seqnum* exists and it's trigger state is true.

10.69.4.177 add_trigger()

```
void seq64::perform::add_trigger (
    int seqnum,
    midipulse tick )
```

Parameters

<i>seqnum</i>	Indicates the sequence that needs to have its trigger handled.
<i>tick</i>	The MIDI pulse number at which the trigger should be handled.

10.69.4.178 delete_trigger()

```
void seq64::perform::delete_trigger (
    int seqnum,
    midipulse tick )
```

Parameters

<i>seqnum</i>	Indicates the sequence that needs to have its trigger handled.
<i>tick</i>	The MIDI pulse number at which the trigger should be handled.

10.69.4.179 add_or_delete_trigger()

```
void seq64::perform::add_or_delete_trigger (
    int seqnum,
    midipulse tick )
```

Parameters

<i>seqnum</i>	Indicates the sequence that needs to have its trigger handled.
<i>tick</i>	The MIDI pulse number at which the trigger should be handled.

10.69.4.180 split_trigger()

```
void seq64::perform::split_trigger (
    int seqnum,
    midipulse tick )
```

Parameters

<i>seqnum</i>	Indicates the sequence that needs to have its trigger split.
<i>tick</i>	The MIDI pulse number at which the trigger should be split.

10.69.4.181 paste_trigger()

```
void seq64::perform::paste_trigger (
    int seqnum,
    midipulse tick )
```

Parameters

<i>seqnum</i>	Indicates the sequence that needs to have its trigger pasted.
<i>tick</i>	The MIDI pulse number at which the trigger should be pasted.

10.69.4.182 paste_or_split_trigger()

```
void seq64::perform::paste_or_split_trigger (
    int seqnum,
    midipulse tick )
```

Parameters

<i>seqnum</i>	Indicates the sequence that needs to have its trigger handled.
<i>tick</i>	The MIDI pulse number at which the trigger should be handled.

10.69.4.183 intersect_triggers()

```
bool seq64::perform::intersect_triggers (
    int seqnum,
    midipulse tick )
```

Parameters

<i>seqnum</i>	The number of the sequence in question.
<i>tick</i>	The time-location desired.

Returns

Returns true if the sequence exists and the [sequence::intersect_triggers\(\)](#) function returned true.

10.69.4.184 get_max_trigger()

```
midipulse seq64::perform::get_max_trigger ( ) const
```

Returns

Returns the highest trigger value, or zero. It is not clear why this function doesn't return a "no trigger found" value. Is there always at least one trigger, at 0?

10.69.4.185 is_dirty_main()

```
bool seq64::perform::is_dirty_main (
    int seq )
```

See the [sequence::is_dirty_main\(\)](#) function.

Parameters

<i>seq</i>	The pattern number. It is checked for validity.
------------	---

Returns

Returns the was-active-main flag value, before setting it to false. Returns false if the pattern was invalid.

10.69.4.186 is_dirty_edit()

```
bool seq64::perform::is_dirty_edit (
    int seq )
```

Parameters

<i>seq</i>	The pattern number. It is checked for validity.
------------	---

Returns

Returns the was-active-edit flag value, before setting it to false. Returns false if the pattern was invalid.

10.69.4.187 is_dirty_perf()

```
bool seq64::perform::is_dirty_perf (
    int seq )
```

Parameters

<i>seq</i>	The pattern number. It is checked for validity.
------------	---

Returns

Returns the was-active-perf flag value, before setting it to false. Returns false if the pattern/sequence number was invalid.

10.69.4.188 is_dirty_names()

```
bool seq64::perform::is_dirty_names (
    int seq )
```

Parameters

<i>seq</i>	The pattern number. It is checked for validity.
------------	---

Returns

Returns the was-active-names flag value, before setting it to false. Returns false if the pattern/sequence number was invalid.

10.69.4.189 is_exportable()

```
bool seq64::perform::is_exportable (
    int seq ) const
```

Parameters

<i>seq</i>	The index of the desired sequence.
------------	------------------------------------

Returns

Returns true if the sequence has the three properties noted above.

10.69.4.190 set_screenset()

```
int seq64::perform::set_screenset (
    int ss )
```

It's not clear that we need to set the "is modified" flag just because we changed the screen-set, so we don't.

This function is called when incrementing and decrementing the screenset. Its counterpart, [set_playing_screenset\(\)](#), is called when the hot-key or the MIDI control for setting the screenset is called.

As a new feature, we would like to queue-mute the previous screenset, and queue-unmute the newly-selected screenset. Still working on getting it right. Aborted.

Parameters

<i>ss</i>	The index of the desired new screen-set. It is forced to range from 0 to m_max_sets - 1. The clamping seems weird, but hews to seq24. What it does is let the user wrap around the screen-sets in the user interface. The value set here will represent the "active" screen-set in multi-window mode.
-----------	---

Returns

Returns the actual final value of the screen-set that was set, i.e. the m_screenset member value.

10.69.4.191 screenset()

```
int seq64::perform::screenset ( ) const [inline]
```

10.69.4.192 get_playing_screenset()

```
int seq64::perform::get_playing_screenset ( ) const [inline]
```

10.69.4.193 toggle_other_seqs()

```
bool seq64::perform::toggle_other_seqs (
    int seqnum,
    bool isshiftkey )
```

See [mainwid::on_button_release_event\(\)](#). If the Shift key is pressed, toggle the mute state of all other sequences. Inactive sequences are skipped.

Parameters

<i>seqnum</i>	The sequence that is being clicked on. It must be active in order to allow toggling.
<i>ishiftkey</i>	Indicates if the shift-key functionality for toggling all of the other sequences is active.

Returns

Returns true if the full toggling was able to be performed.

10.69.4.194 toggle_other_names()

```
bool seq64::perform::toggle_other_names (
    int seqnum,
    bool isshiftkey )
```

See [perfname::on_button_press_event\(\)](#). If the Shift key is pressed, toggle the mute state of all other sequences. Inactive sequences are skipped.

Parameters

<i>seqnum</i>	The sequence that is being clicked on. It must be active in order to allow toggling.
<i>ishiftkey</i>	Indicates if the shift-key functionality for toggling all of the other sequences is active.

Returns

Returns true if the toggling was able to be performed.

10.69.4.195 are_any_armed()

```
bool seq64::perform::are_any_armed ( )
```

Returns

Returns true if even one sequence is armed.

10.69.4.196 max_sets() [1/2]

```
void seq64::perform::max_sets (
    int sets ) [inline]
```

Other than that, it should not be used. We may find a way to enforce that, later.

10.69.4.197 seqs_in_set()

```
void seq64::perform::seqs_in_set (
    int seqs ) [inline]
```

10.69.4.198 song_recording() [1/2]

```
bool seq64::perform::song_recording ( ) const [inline]
```

10.69.4.199 song_record_snap() [1/2]

```
bool seq64::perform::song_record_snap ( ) const [inline]
```

10.69.4.200 resume_note_ons() [1/2]

```
bool seq64::perform::resume_note_ons ( ) const [inline]
```

10.69.4.201 resume_note_ons() [2/2]

```
void seq64::perform::resume_note_ons (
    bool f ) [inline]
```

10.69.4.202 select_trigger()

```
bool seq64::perform::select_trigger (
    int dropseq,
    midipulse droptick )
```

Parameters

<i>dropseq</i>	The sequence number that was in play for the location of the mouse in the (for example) perfedit roll.
<i>droptick</i>	The location of the mouse horizontally in the perfroll.

Returns

Returns true if a trigger was there to select, and was selected.

10.69.4.203 unselect_all_triggers()

```
void seq64::perform::unselect_all_triggers ( )
```

10.69.4.204 get_bank_name()

```
const std::string& seq64::perform::get_bank_name (
    int bank ) const [inline]
```

However, we will still refer to them as "sets".

10.69.4.205 set_looping()

```
void seq64::perform::set_looping (
    bool looping ) [inline]
```

Parameters

<i>looping</i>	The boolean value to set for looping, used in the performance editor.
----------------	---

10.69.4.206 get_sequence_color()

```
int seq64::perform::get_sequence_color (
    int seqnum ) const [inline]
```

We don't want perform knowing the details of the palette color, just treat it as an integer.

10.69.4.207 set_sequence_color()

```
void seq64::perform::set_sequence_color (
    int seqnum,
    int c ) [inline]
```

10.69.4.208 have_undo()

```
bool seq64::perform::have_undo ( ) const [inline]
```

10.69.4.209 set_have_undo()

```
void seq64::perform::set_have_undo (
    bool undo ) [inline]
```

Once it is set, it remains set, unless cleared by saving the file.

10.69.4.210 have_redo()

```
bool seq64::perform::have_redo ( ) const [inline]
```

10.69.4.211 set_have_redo()

```
void seq64::perform::set_have_redo (
    bool redo ) [inline]
```

10.69.4.212 seq_edit_mode() [1/2]

```
edit_mode_t seq64::perform::seq_edit_mode (
    int seq ) const [inline]
```

10.69.4.213 seq_edit_mode() [2/2]

```
void seq64::perform::seq_edit_mode (
    int seq,
    edit_mode_t ed ) [inline]
```

Was private, but a class can have too many friends.

Parameters

<i>seq</i>	Provides the sequence number. If the sequence is not active (available), then nothing is done.
<i>ed</i>	Provides the edit mode, which is "note" or "drum", and which determines if the duration of events matters (note) or not (drum).

10.69.4.214 `current_screenset_notepad()`

```
const std::string& seq64::perform::current_screenset_notepad ( ) const [inline]
```

10.69.4.215 `set_screenset_notepad()` [1/2]

```
void seq64::perform::set_screenset_notepad (
    int screenset,
    const std::string & notepad,
    bool is_load_modification = false )
```

Parameters

<i>screenset</i>	The ID number of the screen-set, an index into the <code>m_screenset_notepad[]</code> array.
<i>notepad</i>	Provides the string date to copy into the notepad. Not sure why a pointer is used, instead of nice "const std::string &" parameter. And this pointer isn't checked. Fixed.
<i>is_load_modification</i>	If true (the default is false), we do not want to set the modify flag, otherwise the user is prompted to save even if no changes have occurred.

10.69.4.216 `set_screenset_notepad()` [2/2]

```
void seq64::perform::set_screenset_notepad (
    const std::string & note ) [inline]
```

Parameters

<i>note</i>	The string value to set into the notepad text.
-------------	--

10.69.4.217 `start()`

```
void seq64::perform::start (
    bool songmode )
```

Question Should we also call `song_start_mode(songmode)` here?

Parameters

<i>songmode</i>	If true, playback is to be in Song mode. Otherwise, it is to be in Live mode.
-----------------	---

10.69.4.218 stop()

```
void seq64::perform::stop ( )
```

The logic seems backward here, in that we call [inner_stop\(\)](#) if JACK is not running. Or perhaps we misunderstand the meaning of `m_jack_running`?

Stazed:

This function's sole purpose was to prevent `inner_stop()` from being called internally when JACK was running... potentially twice. `inner_stop()` was called by `output_func()` when JACK sent a `JackTransportStopped` message. If `seq42` initiated the stop, then `stop_jack()` was called which then triggered the `JackTransportStopped` message to `output_func()` which then triggered the bool `stop_jack` to call `inner_stop()`. The `output_func()` call to `inner_stop()` is only necessary when some other JACK client sends a `jack_transport_stop` message to JACK, not when it is initiated by `seq42`. The method of relying on JACK to call `inner_stop()` when internally initiated caused a (very) obscure apparent freeze if you press and hold the start/stop key if set to toggle. This occurs because of the delay between `JackTransportStarting` and `JackTransportStopped` if both triggered in rapid succession by holding the toggle key down. The variable `global_is_running` gets set false by a delayed `inner_stop()` from JACK after the start (true) is already sent. This means the global is set to true when JACK is actually off (false). Any subsequent presses to the toggle key send a stop message because the global is set to true. Because JACK is not running, `output_func()` is not running to send the `inner_stop()` call which resets the global to false. Thus an apparent freeze as the toggle key endlessly sends a stop, but `inner_stop()` never gets called to reset. Whoo! So, to fix this we just need to call `inner_stop()` directly rather than wait for JACK to send a delayed stop, only when running. This makes the whole purpose of this `stop()` function unneeded. The check for `m_jack_running` is commented out and this function could be removed. It is being left for future generations to ponder!!!

10.69.4.219 start_jack()

```
void seq64::perform::start_jack ( ) [inline]
```

10.69.4.220 stop_jack()

```
void seq64::perform::stop_jack ( ) [inline]
```

10.69.4.221 song_recording_stop()

```
void seq64::perform::song_recording_stop ( )
```

Should be called only when not recording the performance data. This is a Kepler34 feature.

10.69.4.222 song_recording() [2/2]

```
void seq64::perform::song_recording (
    bool f ) [inline]
```

10.69.4.223 song_record_snap() [2/2]

```
void seq64::perform::song_record_snap (
    bool f ) [inline]
```

10.69.4.224 playback_mode() [1/2]

```
bool seq64::perform::playback_mode ( ) [inline]
```

10.69.4.225 playback_mode() [2/2]

```
void seq64::perform::playback_mode (
    bool playbackmode ) [inline]
```

Parameters

<i>playbackmode</i>	The value of the playback mode flag to be set.
---------------------	--

10.69.4.226 is_group_learning()

```
bool seq64::perform::is_group_learning ( ) [inline]
```

10.69.4.227 set_beats_per_minute()

```
void seq64::perform::set_beats_per_minute (
    midibpm bpm )
```

Replaces perform::set_bpm() from seq24.

The value is set only if neither JACK nor this performance object are running.

It's not clear that we need to set the "is modified" flag just because we changed the beats per minute. This setting does get saved to the MIDI file, with the `c_bpmtag`.

Parameters

<i>bpm</i>	Provides the beats/minute value to be set. It is clamped, if necessary, between the values SEQ64_MINIMUM_BPM to SEQ64_MAXIMUM_BPM. They provide a wide range of speeds, well beyond what normal music needs.
------------	--

10.69.4.228 `panic()`

```
void seq64::perform::panic ( )
```

Adapted from Oli Kester's Kepler34 project.

10.69.4.229 `ppqn()` [2/2]

```
void seq64::perform::ppqn (
    int p ) [inline], [private]
```

10.69.4.230 `collapse()`

```
void seq64::perform::collapse ( ) [inline], [private]
```

10.69.4.231 `copy()`

```
void seq64::perform::copy ( ) [inline], [private]
```

10.69.4.232 `expand()`

```
void seq64::perform::expand ( ) [inline], [private]
```

10.69.4.233 `midi_control_toggle()`

```
midi_control & seq64::perform::midi_control_toggle (
    int ctl ) [private]
```

Recall that the [midi_control](#) object specifies if a control is active, inversely-active, what status byte initiates it, what data byte initiates it, and the min/max values. Note that the status byte determines what category of event it is (e.g. note on/off versus a continuous controller), and the data byte indicates the note value or the type of continuous controller.

Parameters

<i>ctl</i>	Provides the index to pass to valid_midi_control_seq() to obtain a control value (such as <code>c_midi_control_bpm_up</code>) to use to retrieve the desired midi_control object.
------------	--

Returns

Returns the "toggle" value if the control value is valid, or a reference to `sm_mc_dummy` otherwise.

10.69.4.234 `midi_control_on()`

```
midi_control & seq64::perform::midi_control_on (
    int ctl ) [private]
```

Parameters

<i>ctl</i>	Provides the index to pass to valid_midi_control_seq() to obtain a control value (such as <code>c_midi_control_bpm_up</code>) to use to retrieve the desired midi_control object.
------------	--

Returns

Returns the "on" value if the control value is valid, and a reference to `sm_mc_dummy` otherwise.

10.69.4.235 `midi_control_off()`

```
midi_control & seq64::perform::midi_control_off (
    int ctl ) [private]
```

Parameters

<i>ctl</i>	Provides a control value (such as <code>c_midi_control_bpm_up</code>) to use to retrieve the desired midi_control object.
------------	--

Returns

Returns the "off" value if the control value is valid, and a reference to `sm_mc_dummy` otherwise.

10.69.4.236 `midi_control_event()`

```
bool seq64::perform::midi_control_event (
    const event & ev ) [private]
```

Here is the processing involved in this function TODO.

Incorporates pull request #24, arnaud-jacquemin, issue #23 "MIDI controller toggles wrong pattern".

Note that the `event::get_status()` function returns a value with the channel nybble stripped off.

Parameters

<code>ev</code>	Provides the MIDI event to potentially trigger a control action.
-----------------	--

Returns

Returns true if the event was actually handled. TODO.

10.69.4.237 midi_control_record()

```
bool seq64::perform::midi_control_record (
    const event & ev ) [private]
```

This function is used for a quick check while recording, so we don't have scan 84 items before adding a musical MIDI event, but still can detect a recording-status change command.

We handle record, but also need to handle quan_record and thru. `midi_control_event()` iterates through all values. We need to "iterate" between the record, quan_record, and thru values only.

Parameters

<code>ev</code>	Provides the MIDI event to potentially trigger a recording-control action.
-----------------	--

Returns

Returns true if the event was an active recording control event.

10.69.4.238 handle_midi_control()

```
bool seq64::perform::handle_midi_control (
    int ctl,
    bool state ) [private]
```

```
c_midi_control_mod_replace
c_midi_control_mod_snapshot
c_midi_control_mod_queue
c_midi_control_mod_gmute
c_midi_control_mod_glearn
```

Other values supported:

```

c_midi_control_bpm_up
c_midi_control_bpm_dn
c_midi_control_ss_up
c_midi_control_ss_dn
c_midi_control_play_ss

```

We have added the following extended values:

```

c_midi_control_playback      (for pause/toggle, start, and stop)
c_midi_control_song_record  (for performance record toggle/on/off)
c_midi_control_solo         (for toggle, on, or off)
c_midi_control_thru         (see record below)
c_midi_control_bpm_page_up
c_midi_control_bpm_page_dn
c_midi_control_ss_set
c_midi_control_record       (see thru above)
c_midi_control_quan_record  (quantized record, see thru above)
c_midi_control_reset_seq    (resets the sequence)

```

The extended values will actually be handled by a new function, `handle_midi_control_ex()`.

`c_midi_control_solo` probably will need a parameter.

Values from 32 through 2×32 are normalized by subtracting 32 and passed to the `select_and_mute_group()` function. Otherwise, the following apply:

We also reserve a few control values above that for expansion.

Parameters

<i>ctl</i>	The MIDI control value to use to perform an operation.
<i>state</i>	The state of the control, used with the following values:

Returns

Returns true if the event was handled. Mostly rote, for conformance with the newer [handle_midi_control_ex\(\)](#) function.

10.69.4.239 handle_midi_control_ex()

```

bool seq64::perform::handle_midi_control_ex (
    int ctl,
    midi_control::action a,
    int v ) [private]

```

Parameters

<i>ctl</i>	The MIDI control value to use to perform an operation.
<i>a</i>	The action of the control.
<i>v</i>	The value of the control (ie.: note velocity / control change value). Also called a "parameter" in the comments.

Returns

Returns true if the control was an extended control and was acted on.

10.69.4.240 handle_midi_control_event()

```
bool seq64::perform::handle_midi_control_event (
    const event & ev,
    int ctl,
    int offset = 0 ) [private]
```

Parameters

<i>ev</i>	Provides the MIDI event to potentially trigger a control action.
<i>ctl</i>	The specific MIDI control to check.
<i>offset</i>	The offset into the control list, used only for changing the playing status of a sequence/pattern in the current screen-set.

Returns

Returns true if the event was matched and handled.

10.69.4.241 get_screenset_notepad()

```
const std::string & seq64::perform::get_screenset_notepad (
    int screenset ) const [private]
```

Parameters

<i>screenset</i>	The ID number of the screen-set, an index into the m_screenset_notepad[] array. This value is validated.
------------------	--

Returns

Returns a reference to the desired string, or to an empty string if the screen-set number is invalid.

10.69.4.242 any_group_unmutes()

```
bool seq64::perform::any_group_unmutes ( ) const [private]
```

Returns

Returns true if there are any unmute statuses in the mute-group array. If they're all zero, we don't need to save them.

10.69.4.243 print_group_unmutes()

```
void seq64::perform::print_group_unmutes ( ) const [private]
```

10.69.4.244 mute_group_tracks()

```
void seq64::perform::mute_group_tracks ( ) [private]
```

It loops through every screen-set. In each screen-set, it acts on each active sequence. If the active sequence is in the current "in-view" screen-set (`m_screenset` as opposed to `m_playscreen`, and its `m_track_mute_state[]` is true, then the sequence is turned on, otherwise it is turned off. The result is that the in-view screen-set is activated as per the mute states, while all other screen-sets are muted.

Change Note tdeagan 2015-12-22 via git pull. Replaced `m_playscreen` with `m_screenset`.

We are disabling Tim's fix for a bit in order to make sure we're doing what Seq24 does with the playing set key. (Home).

It seems to us that the for (g) clause should have g range from 0 to `m_max_sets`, not `m_seqs_in_set`. Done.

10.69.4.245 select_and_mute_group()

```
void seq64::perform::select_and_mute_group (
    int group ) [private]
```

Called in `perform` and in `mainwnd`.

Parameters

<i>group</i>	Provides the group number for the group to be muted.
--------------	--

10.69.4.246 set_song_mute()

```
void seq64::perform::set_song_mute (
    mute_op_t op ) [private]
```

The [sequence::set_song_mute\(\)](#) and `toggle_song_mute()` functions do all the work, including mp-dirtying the sequence.

We've modified this function to call [mute_all_tracks\(\)](#) and [toggle_all_tracks\(\)](#) in order to consolidate the code and (cough cough) fix a bug in this functionality from the `mainwnd` menu.

Question Do we want to replace the call to [toggle_all_tracks\(\)](#) with a call to [toggle_playing_tracks\(\)](#)?

Parameters

<i>op</i>	Provides the "flag" that indicates if this function is to set mute on, off, or to toggle the mute status.
-----------	---

10.69.4.247 set_playing_screenset()

```
void seq64::perform::set_playing_screenset ( ) [private]
```

This function is called when one of the snapshot keys is pressed.

For each value up to `m_seqs_in_set` (32), the index of the current sequence in the current screen-set (`m_playscreen`) is obtained. If the sequence is active and the sequence actually exists, it is processed; null sequences are no longer flagged as an error, they are just ignored.

Modifies `m_playscreen`, `m_playscreen_offset`, stores the current playing-status of each sequence in `m_tracks_mute_state[]`, and then calls [mute_group_tracks\(\)](#), turns on unmuted tracks in the current screen-set.

Basically, this function retrieves and saves the playing status of the sequences in the current play-screen, sets the play-screen to the current screen-set, and then mutes the previous play-screen. It is called via the `c_midi_control_play_ss` value or via the set-playing-screen-set keystroke.

10.69.4.248 set_mode_group_mute()

```
void seq64::perform::set_mode_group_mute ( ) [inline], [private]
```

10.69.4.249 unset_mode_group_mute()

```
void seq64::perform::unset_mode_group_mute ( ) [inline], [private]
```

10.69.4.250 select_group_mute()

```
void seq64::perform::select_group_mute (
    int mutegroup ) [private]
```

Then, no matter what, it makes the desired mute-group the selected mute-group. Compare to [set_and_copy_mute_group\(\)](#).

One thing to note is that, once saved, then, if used, it is applied to the current screen-set, even if it is not the screen-set whose playing status were saved.

Also note that this function now supports variset mode, via `screen_offset()`.

Parameters

<i>mutegroup</i>	The number of the desired mute group, clamped to be between 0 and m_seqs_in_set-1. Obviously, it is the set whose state is to be stored, if in group-learn mode.
------------------	--

10.69.4.251 `set_mode_group_learn()`

```
void seq64::perform::set_mode_group_learn ( ) [private]
```

This function is called via a MIDI control `c_midi_control_mod_glearn` and via the group-learn keystroke.

10.69.4.252 `unset_mode_group_learn()`

```
void seq64::perform::unset_mode_group_learn ( ) [private]
```

Then unsets the group-learn mode flag. This function is called via a MIDI control `c_midi_control_mod_glearn`, via the group-learn keystroke, and in [mainwnd::on_key_press_event\(\)](#), to end the group-learn mode.

Shouldn't this function also call this one, to perfectly complement `set_mode_group_learn`: [unset_mode_group_mute\(\)](#). Too tricky.

10.69.4.253 `load_mute_group()`

```
bool seq64::perform::load_mute_group (
    int gmute,
    int gm[c_max_groups] ) [private]
```

In addition, a copy is saved in a second mute-group array in case we want to avoid "corrupting" the "rc" mute-groups with mute-groups from the MIDI file.

It also uses the constant `c_seqs_in_set` instead of the variable `m_seqs_in_set` so that the loading always handles $32 \times 32 = 1024$ settings.

Replaces "`set_group_mute_state(g, gm[s] != 0)`" and [mute_group_offset\(\)](#), which depend on the configured "seqs-in-set" value. Here, we are always tied to the legacy 32×32 sizes, to load and save to the configuration.

Parameters

<i>gmute</i>	The mute-group to load. If out-of-range (0 to c_max_groups), no setting is made. There's no need to call clamp_group() ; it is useful in playing, not in loading the configuration.
<i>gm</i>	The array of c_seqs_in_set values to be used for the load. The pointer is not checked.

Returns

Returns true if the group was valid.

10.69.4.254 save_mute_group()

```
bool seq64::perform::save_mute_group (
    int gmute,
    int gm[c_max_groups] ) const [private]
```

However, it has a wrinkle in what it saves. First, if there are no unmuted patterns in any of the mute-groups, then the original mute-groups read from the "rc" file are saved back. Second, this saving is also done if the `e_mute_group_preserve` flag is set. If the `e_mute_group_stomp` flag is set, then the active mute-group statuses are written to the "rc" file.

Parameters

<i>gmute</i>	The mute-group to save. If out-of-range (0 to c_max_groups), no saving is done.
<i>gm</i>	The array of c_seqs_in_set values to be used for the save. The pointer is not checked.

Returns

Returns true if the group was valid.

10.69.4.255 set_and_copy_mute_group()

```
void seq64::perform::set_and_copy_mute_group (
    int mutegroup ) [private]
```

Then the mute-group is stored in `m_tracks_mute_state[]`, which holds states for only the number of sequences in a set.

Compare to [select_group_mute\(\)](#); its main difference is that it will at least copy the states even if not in group-learn mode. And, if in group-learn mode, it will grab the playing states of the sequences before copying them.

This function is used only once, in [select_and_mute_group\(\)](#). It used to be called just `select_mute_group()`, but that's too easy to confuse with [select_group_mute\(\)](#).

Change Note tdeagan 2015-12-22 via git pull: [git pull https://github.com/TDeagan/sequencer64](https://github.com/TDeagan/sequencer64). ↩
[git](#) `mute_groups` `m_screenset` replaces `m_playscreen_offset`.

Parameters

<i>mutegroup</i>	Provides the mute-group to select.
------------------	------------------------------------

10.69.4.256 activate()

```
bool seq64::perform::activate ( ) [private]
```

Currently does work only for JACK; the [activate\(\)](#) calls for other APIs just return true without doing anything.

10.69.4.257 position_jack()

```
void seq64::perform::position_jack (
    bool songmode,
    midipulse tick = 0 ) [private]
```

Parameters

<i>songmode</i>	If true, playback is to be in Song mode. Otherwise, it is to be in Live mode.
<i>tick</i>	Provides the pulse position to be set. The default value is 0.

10.69.4.258 off_sequences()

```
void seq64::perform::off_sequences ( ) [private]
```

Replaces "for (int s = 0; s < m_sequence_max; ++s)"

10.69.4.259 unqueue_sequences()

```
void seq64::perform::unqueue_sequences (
    int current_seq ) [private]
```

This function supports the new queued-replace feature. This feature is activated when the keep-queue feature is turned on via its hot-key, and then the replace hot-key is used on a pattern. When that happens, the current sequence is queued to be toggled, and all unmuted sequences are queued to toggle off at the same time. Thus, this is a kind of queued-solo feature.

This function assumes we have called [save_current_screenset\(\)](#) first, so that the soloing can be exactly toggled. Only sequences that were initially on should be toggled.

Parameters

<i>current_seq</i>	This number is that of the sequence/pattern whose hot-key was struck. We don't want to toggle this one off, just on.
--------------------	--

10.69.4.260 all_notes_off()

```
void seq64::perform::all_notes_off ( ) [private]
```

Then flush the master MIDI buss.

10.69.4.261 set_active()

```
void seq64::perform::set_active (
    int seq,
    bool active ) [private]
```

If setting it active, the [sequence::number\(\)](#) setter is called. It won't modify the sequence's internal copy of the sequence number if it has already been set.

Parameters

<i>seq</i>	Provides the prospective sequence number.
<i>active</i>	True if the sequence is to be set to the active state.

10.69.4.262 set_was_active()

```
void seq64::perform::set_was_active (
    int seq ) [private]
```

Why do we need this routine?

Parameters

<i>seq</i>	The pattern number. It is checked for validity.
------------	---

10.69.4.263 reset_sequences()

```
void seq64::perform::reset_sequences (
    bool pause = false ) [private]
```

Note that these calls are folded into one member function of the sequence class. Finally, flush the master MIDI buss.

Could use a member function pointer to avoid having to code two loops. We did it.

Parameters

<i>pause</i>	Try to prevent notes from lingering on pause if true. By default, it is false.
--------------	--

10.69.4.264 play()

```
void seq64::perform::play (
    midipulse tick ) [private]
```

Starts the playing of all the patterns/sequences.

This function just runs down the list of sequences and has them dump their events. It skips sequences that have no playable MIDI events.

Note how often the "sp" (sequence) pointer was used. It was worth offloading all these calls to a new sequence function. Hence the new [sequence::play_queue\(\)](#) function.

Finally, we stop the looping at m_sequence_high rather than m_sequence_max, to save a little time.

Parameters

<i>tick</i>	Provides the tick at which to start playing. This value is also copied to m_tick.
-------------	---

10.69.4.265 set_orig_ticks()

```
void seq64::perform::set_orig_ticks (
    midipulse tick ) [private]
```

This is really the "last tick" value, so we renamed sequence::set_orig_tick() to [sequence::set_last_tick\(\)](#).

Parameters

<i>tick</i>	Provides the last-tick value to be set for each sequence that is active.
-------------	--

10.69.4.266 max_active_set()

```
int seq64::perform::max_active_set ( ) const [private]
```

Returns

Returns the value of the highest active set. A value of 0 represents the first set. If no sequences are active, then -1 is returned.

10.69.4.267 launch_input_thread()

```
void seq64::perform::launch_input_thread ( ) [private]
```

This might be a good candidate for a small thread class derived from a small base class.

10.69.4.268 launch_output_thread()

```
void seq64::perform::launch_output_thread ( ) [private]
```

This might be a good candidate for a small thread class derived from a small base class.

10.69.4.269 init_jack_transport()

```
bool seq64::perform::init_jack_transport ( ) [private]
```

Who calls this routine? The main() routine of the application [via [launch\(\)](#)], and the options module, when the Connect button is pressed.

Returns

Returns the result of the init() call; true if JACK sync is now running. If JACK support is not built into the application, then this function returns false, to indicate that JACK is (definitely) not running.

10.69.4.270 deinit_jack_transport()

```
bool seq64::perform::deinit_jack_transport ( ) [private]
```

Called by [launch\(\)](#) and in the options module, when the Disconnect button is pressed. This function operates only while Sequencer64 is not outputting, otherwise we have a race condition that can lead to a crash.

Returns

Returns the result of the init() call; false if JACK sync is now no longer running. If JACK support is not built into the application, then this function returns true, to indicate that JACK is (definitely) not running.

10.69.4.271 seq_in_playing_screen()

```
bool seq64::perform::seq_in_playing_screen (
    int seq ) [private]
```

Parameters

<i>seq</i>	Provides the index of the desired sequence.
------------	---

Returns

Returns true if the sequence adheres to the conditions noted above.

10.69.4.272 `is_modified()` [2/2]

```
void seq64::perform::is_modified (
    bool flag ) [inline], [private]
```

Parameters

<i>flag</i>	The value of the modified flag to be set.
-------------	---

10.69.4.273 `valid_midi_control_seq()`

```
bool seq64::perform::valid_midi_control_seq (
    int seq ) const [inline], [private]
```

We were checking against `c_midi_track_ctrl` as well, but that was a bug. This function is meant to check that the supplied sequence number does not exceed the value of `c_midi_controls_extended` ($32 * 2 + 10 + 10 = 84$). The track (sequence or pattern) controls range from 0 to 64. Next come the "c_midi_control" values: `bpm_up`, `bpm_dn`, ..., `play_ss`, plus some extended controls that are relatively new, and, lastly, `c_midi_controls_extended` itself.

Parameters

<i>seq</i>	The sequence number value that should be inside the <code>c_midi_controls_extended</code> range. This value can specify not only a sequence number, but larger control values as well, so the function and parameter are mildly mis-named.
------------	--

Returns

Returns true if the sequence number is valid for accessing the MIDI control values. For this function, no error print-out is generated.

10.69.4.274 `max_sets()` [2/2]

```
int seq64::perform::max_sets ( ) const [inline], [private]
```

10.69.4.275 `is_screenset_valid()`

```
bool seq64::perform::is_screenset_valid (
    int screenset ) const [inline], [private]
```

Parameters

<i>screenset</i>	The prospective screenset value.
------------------	----------------------------------

Returns

Returns true if the parameter is valid. For this function, no error print-out is generated.

10.69.4.276 is_running() [2/2]

```
void seq64::perform::is_running (
    bool running ) [inline], [private]
```

Parameters

<i>running</i>	The value of the running flag to be set.
----------------	--

10.69.4.277 screenset_offset() [2/2]

```
int seq64::perform::screenset_offset (
    int ss ) [inline], [private]
```

It supports variset mode (which is active if `m_seqs_in_set != c_seq_in_set`).

Parameters

<i>ss</i>	Provides the screen-set number, ranging from 0 to <code>c_max_sets-1</code> . This value is not validated, for speed.
-----------	---

Returns

Returns the product of *ss* and `m_seqs_in_set`.

10.69.4.278 is_seq_valid()

```
bool seq64::perform::is_seq_valid (
    int seq ) const [private]
```

Also see the function [is_mseq_valid\(\)](#), which also checks the pointer stored in the `m_seq[]` array.

We considered checking the *seq* param against [sequence_count\(\)](#), but this function is called while creating sequences that add to that count, so we continue checking against the "container" size. Also, it is possible to have holes in the array representing inactive sequences, so that `sequencer_count()` would be too limiting.

Parameters

<i>seq</i>	The sequencer number, in interval <code>[0, m_sequence_max)</code> .
------------	--

Returns

Returns true if the sequence number is valid.

10.69.4.279 is_mseq_valid()

```
bool seq64::perform::is_mseq_valid (
    int seq ) const [private]
```

It also evaluates the m_seq[seq] pointer value.

Note

Since we can have holes in the sequence array, where there are inactive sequences, we check if the sequence is even active before emitting a message about a null pointer for the sequence. We only want to see messages that indicate actual problems.

Parameters

<i>seq</i>	Provides the sequence number to be checked. It is checked for validity. We cannot compare the sequence number versus the sequence_count() , because the current implementation can have inactive holes (with null pointers) interspersed with active pointers.
------------	--

Returns

Returns true if the sequence number is valid as per [is_seq_valid\(\)](#), and the sequence pointer is not null.

10.69.4.280 install_sequence()

```
bool seq64::perform::install_sequence (
    sequence * seq,
    int seqnum ) [private]
```

It is common code and using it prevents inconsistencies. It assumes values have already been checked. It does not set the "is modified" flag, since adding a sequence by loading a MIDI file should not set it. Compare [new_↵sequence\(\)](#), used by mainwid and seqmenu, with [add_sequence\(\)](#), used by midifile.

Parameters

<i>seq</i>	The pointer to the pattern/sequence to add.
<i>seqnum</i>	The sequence number of the pattern to be added. Not validated, to save some time.

Returns

Returns true if a sequence was removed, or the sequence was successfully added. In other words, if a real change in sequence pointers occurred. It is up to the caller to decide if the change warrants setting the "is modified" flag.

10.69.4.281 inner_start()

```
void seq64::perform::inner_start (
    bool songmode ) [private]
```

Then, if not [is_running\(\)](#), the playback mode is set to the given state. If that state is true, call [off_sequences\(\)](#). Set the running status, and signal the condition. Then unlock.

Minor issue:

In ALSA mode, restarting the sequence moves the progress bar to the beginning of the sequence, even if just pausing. This is fixed by compiling with SEQ64_PAUSE_SUPPORT, which disables calling [off_sequences\(\)](#) when starting playback from the song editor / performance window.

Parameters

<i>songmode</i>	Sets the playback mode, and, if true, turns off all of the sequences before setting the is-running condition.
-----------------	---

10.69.4.282 inner_stop()

```
void seq64::perform::inner_stop (
    bool midiclock = false ) [private]
```

Sets `m_usemidiclock` to the given value. Note that we do need to set the running flag to false here, even when JACK is running. Otherwise, JACK starts ping-ponging back and forth between positions under some circumstances.

However, if JACK is running, we do not want to reset the sequences... this causes the progress bar for each sequence to move to near the end of the sequence.

Parameters

<i>midiclock</i>	If true, indicates that the MIDI clock should be used.
------------------	--

10.69.4.283 clamp_track()

```
int seq64::perform::clamp_track (
    int track ) const [private]
```

Fixed the bug we found, where we checked for `track > m_seqs_in_set`, instead of using the `>=` operator. Supports variset mode.

Parameters

<i>track</i>	The track value to be checked and rectified as necessary.
--------------	---

Returns

Returns the track parameter, clamped between 0 and `m_seqs_in_set-1`, inclusive.

10.69.4.284 clamp_group()

```
int seq64::perform::clamp_group (
    int group ) const [private]
```

Fixed the bug we found, where we checked for `track > m_seqs_in_set`, instead of using the `>=` operator. We now compare against the new `c_max_groups` value (32). But, if the set-size is larger, then we have fewer mute groups available.

Parameters

<i>group</i>	The group value to be checked and rectified as necessary.
--------------	---

Returns

Returns the group parameter, clamped between 0 and `c_max_sets-1`, inclusive. We use `c_max_groups` rather than `m_max_sets` because the new varisets feature reduces the number of sets (in general); this check is basically a sanity check. Additionally, if varisets mode is in force (i.e. there are more than 32 sequences in a set), the group number is clamped to the maximum number of sets under that constraint.

10.69.4.285 set_key_event()

```
void seq64::perform::set_key_event (
    unsigned keycode,
    int sequence_slot ) [inline], [private]
```

It is called 32 times, corresponding to the pattern/sequence slots in the Patterns window. It first removes the given key-code from the regular and reverse slot-maps. Then it removes the sequence-slot from the regular and reverse slot-maps. Finally, it adds the sequence-slot with a key value of key-code, and adds the key-code with a value of sequence-slot.

Parameters

<i>keycode</i>	The keycode for which to set the sequence slot.
<i>sequence_slot</i>	The sequence slot to be set.

10.69.4.286 set_key_group()

```
void seq64::perform::set_key_group (
    unsigned keycode,
    int group_slot ) [inline], [private]
```

It is called 32 times, corresponding the pattern/sequence slots in the Patterns window. Compare it to the `set_key↵_events()` function.

Parameters

<i>keycode</i>	The keycode for which to set the group slot.
<i>group_slot</i>	The group slot to be set.

10.69.4.287 log_current_tempo()

```
bool seq64::perform::log_current_tempo ( ) [private]
```

Note that, if the current tick position is past the end of pattern 0's length, then the length of the tempo track pattern (0 by default) is increased in order to hold the tempo event.

Note that we allow the user to change the tempo track from the default of 0 to any pattern from 0 to 1023. This is done in the File / Options / MIDI Clock tab, and is saved to the "rc" file.

Returns

Returns true if the tempo-track sequence exists.

10.69.4.288 create_master_bus()

```
bool seq64::perform::create_master_bus ( ) [private]
```

We need to delay creation until launch time, so that settings can be obtained before determining just how to set up the application.

Once the master buss is created, we then copy the clocks and input setting that were read from the "rc" file, via the `mastermidibus :: set_port_statuses()` function, to use in determining whether to initialize and connect the input ports at start-up. Seq24 wouldn't connect unconditionally, and Sequencer64 shouldn't, either.

However, the devices actually on the system at start time might be different from what was saved in the "rc" file after the last run of Sequencer64.

For output, both apps have always connected to all ports automatically. But we want to support disabling some output ports, both in the "rc" file and via the operating system indicating that it cannot open an output port. So how do we get the port-settings from the OS? Probably at initialization time. See the `mastermidibus` constructor for `PortMidi`.

Returns

Returns true if the creation succeeded, or if the buss already exists.

10.69.4.289 preallocate_clocks()

```
void seq64::perform::preallocate_clocks (
    int busscount ) [inline], [private]
```

This function and calls to [set_clock\(\)](#) are a more fool-proof option for reading the clocks from the "rc" file. Might eventually do this for inputs as well. LATER.

10.69.4.290 add_clock()

```
void seq64::perform::add_clock (
    clock_e clocktype ) [inline], [private]
```

Parameters

<i>clocktype</i>	The clock value read from the "rc" file.
------------------	--

10.69.4.291 set_clock()

```
void seq64::perform::set_clock (
    bussbyte bus,
    clock_e clocktype ) [inline], [private]
```

Mostly meant for use by the Options / MIDI Input tab.

10.69.4.292 get_clock()

```
clock_e seq64::perform::get_clock (
    bussbyte bus ) const [inline], [private]
```

Meant for use by the [optionsfile::write\(\)](#) function.

```
clock_e get_clock (bussbyte bus) const { return bus < bussbyte(m_master_clocks.size()) ? m_master_clocks[bus]
: e_clock_off ; } STILL THINKING ABOUT THIS. 'Getter' function for member m_master_bus->get_clock(bus);
```

10.69.4.293 add_input()

```
void seq64::perform::add_input (
    bool flag ) [inline], [private]
```

Parameters

<i>flag</i>	The input flag read from the "rc" file.
-------------	---

10.69.4.294 set_input()

```
void seq64::perform::set_input (
    bussbyte bus,
    bool inputing ) [inline], [private]
```

Mostly meant for use by the Options / MIDI Input tab.

10.69.4.295 get_input()

```
bool seq64::perform::get_input (
    bussbyte bus ) const [inline], [private]
```

Changed from:

```
return not_nullptr(m_master_bus) ?
    m_master_bus->get_input(bus) : false ;
```

```
bool get_input (bussbyte bus) const { return bus < bussbyte(m_master_inputs.size()) ? m_master_inputs[bus] :
false ; } STILL THINKING ABOUT THIS. 'Getter' function for member m_master_bus->get_input(bus);
```

10.69.4.296 is_input_system_port()

```
bool seq64::perform::is_input_system_port (
    bussbyte bus ) [inline], [private]
```

10.69.4.297 midi_mute_group_present() [2/2]

```
void seq64::perform::midi_mute_group_present (
    bool flag ) [inline], [private]
```

10.69.5 Friends And Related Function Documentation**10.69.5.1 jack_assistant**

```
friend class jack_assistant [friend]
```

10.69.5.2 keybindentry

```
friend class keybindentry [friend]
```

10.69.5.3 mainwnd

```
friend class mainwnd [friend]
```

10.69.5.4 midifile

```
friend class midifile [friend]
```

10.69.5.5 wrkfile

```
friend class wrkfile [friend]
```

10.69.5.6 optionsfile

```
friend class optionsfile [friend]
```

10.69.5.7 options

```
friend class options [friend]
```

10.69.5.8 perfedit

```
friend class perfedit [friend]
```

10.69.5.9 perfroll

```
friend class perfroll [friend]
```

10.69.5.10 sequence

```
friend class sequence [friend]
```

10.69.5.11 input_thread_func

```
void* input_thread_func (  
    void * myperf ) [friend]
```

Parameters

<i>myperf</i>	Provides the perform object instance that is to be used. Its output_func() is called. Currently, this parameter is not validated, for speed.
---------------	--

Returns

Always returns nullptr.

10.69.5.12 output_thread_func

```
void* output_thread_func (
    void * myperf ) [friend]
```

Set up the performance, set the process to realtime privileges, and then start the output function.

Parameters

<i>myperf</i>	Provides the perform object instance that is to be used. Its output_func() is called. Currently, this parameter is not validated, for speed.
---------------	--

Returns

Always returns nullptr.

10.69.5.13 jack_sync_callback

```
int jack_sync_callback (
    jack_transport_state_t state,
    jack_position_t * pos,
    void * arg ) [friend]
```

10.69.5.14 jack_transport_callback

```
int jack_transport_callback (
    jack_nframes_t nframes,
    void * arg ) [friend]
```

Added the following function. This function is supposed to allow seq24/sequencer64 to follow JACK transport.

For more advanced ideas, see the `MetronomeJack::process_callback()` function in the `klick` project. It plays a metronome tick after calculating if it needs to or not. (Maybe we could use it to provide our own tick for recording patterns.)

If debugging is enabled in the build, then this function outputs the current JACK position information every couple of seconds, which can be useful to examine the interactions with other JACK clients.

The code enabled via `USE_JACK_BBT_OFFSET` sets the JACK position field `bbt_offset` to 0. It doesn't seem to have any effect, though it can be seen when calling `show_position()` in the [jack_transport_callback\(\)](#) function.

Stazed:

```
This process callback is called by JACK whether stopped or rolling.
Assuming every JACK cycle... "...client supplied function that is
called by the engine anytime there is work to be done". There seems to
be no definition of '...work to be done'. nframes = buffer_size -- is
not used.
```

Parameters

<i>nframes</i>	Unused.
<i>arg</i>	Used for debug output now. Note that this function will be called very often, and this pointer will currently not be used unless debugging is turned on.

Returns

Returns 0 on success, non-zero on error.

10.69.5.15 jack_shutdown

```
void jack_shutdown (
    void * arg ) [friend]
```

10.69.5.16 jack_timebase_callback

```
void jack_timebase_callback (
    jack_transport_state_t state,
    jack_nframes_t nframes,
    jack_position_t * pos,
    int new_pos,
    void * arg ) [friend]
```

The original version of the function worked properly with Hydrogen, but not with Klick. The new code seems to work with both. More testing and clarification is needed. This new code was "discovered" in the source-code for the "SooperLooper" project:

<http://essey.net/sooperlooper/>

The first difference with the new code is that it handles the case where the JACK position is moved (`new_pos == true`). If this is true, and the `JackPositionBBT` bit is off in `pos->valid`, then the new BBT value is set.

The second set of differences are in the "else" clause. In the new code, it is very simple: calculate the new tick value, back it off by the number of ticks in a beat, and perhaps go to the first beat of the next bar.

In the old code (complex!), the simple BBT adjustment is always made. This changes (perhaps) the `beats_per_bar`, `beat_type`, etc. We need to make these settings use the actual global values for beats set for `Sequencer64`. Then, if transitioning from `JackTransportStarting` to `JackTransportRolling` (instead of checking `new_pos!`), the BBT values (bar, beat, and tick) are finally adjusted. Here are the steps, with old and new steps noted:


```

-# Calculate the "delta" ticks based on the current frame, the
   ticks_per_beat, the beats_per_minute, and the frame_rate. The old
   code saves this in a local, the new code assigns it to pos->tick.
-# Old code: save this delta as a positive value.
-# Figure out the settings and modify bar, beat, tick, and
   bar_start_tick. The old and new code seem to have the same intent,
   but it seems like the new code is faster and also correct.
    - Old code: Calculations are made by division and mod
      operations.
    - New code: Calculations are made by increments and decrements
      in a while loop.

```

Stazed:

The call to [jack_timebase_callback\(\)](#) to supply JACK with BBT, etc. would occasionally fail when the pos information had zero or some garbage in the pos.frame_rate variable. This would occur when there was a rapid change of frame position by another client... i.e. qjackctl. From the JACK API:

```

pos address of the position structure for the next cycle; pos->frame
will be its frame number. If new_pos is FALSE, this structure contains
extended position information from the current cycle. If TRUE, it
contains whatever was set by the requester. The timebase_callback's
task is to update the extended information here."

```

The "If TRUE" line seems to be the issue. It seems that qjackctl does not always set pos.frame_rate so we get garbage and some strange BBT calculations that display in qjackctl. So we need to set it here and just use `m_↵ jack_frame_rate` for calculations instead of pos.frame_rate.

Parameters

<i>state</i>	Indicates the current state of JACK transport.
<i>nframes</i>	The number of JACK frames in the current time period.
<i>pos</i>	Provides the position structure to be filled in, the address of the position structure for the next cycle; pos->frame will be its frame number. If new_pos is FALSE, this structure contains extended position information from the current cycle. If TRUE, it contains whatever was set by the requester. The timebase_callback's task is to update the extended information here.
<i>new_pos</i>	TRUE (non-zero) for a newly requested pos, or for the first cycle after the timebase_callback is defined. This is usually 0 in Sequencer64 at present, and 1 if one, say, presses "rewind" in qjackctl.
<i>arg</i>	Provides the jack_assistant pointer, currently unchecked for nullity.

Todo Shouldn't we process the first clause ONLY if new_pos is true?

10.69.5.17 get_current_jack_position

```

long get_current_jack_position (
    void * arg ) [friend]

```

The Seq32 version also uses its tempo map to adjust this, but Sequencer64 currently does not.

Warning

Currently valgrind flags `j->client()` as uninitialized.

Parameters

<i>arg</i>	Provides the putative jack_assistant pointer, assumed to be not null.
------------	---

Returns

Returns the calculated tick position if no errors occur. Otherwise, returns 0.

10.69.6 Field Documentation

10.69.6.1 `sm_mc_dummy`

```
midi_control seq64::perform::sm_mc_dummy [static], [private]
```

Instantiate the dummy [midi_control](#) object, which is used in lieu of a null pointer.

We're taking code that basically works already, in the sense that it never seems to access a null pointer. So we're not even risking data transfers between this dummy object and the ones we really want to use.

However, it would be nice to be able to detect any errors that occur. How?

10.69.6.2 `m_song_start_mode`

```
bool seq64::perform::m_song_start_mode [private]
```

This is a replacement for the global setting, but is essentially a global setting itself, and is saved to and restored from the "rc" configuration file. Sometimes called "JACK start mode", it used to be a JACK setting, but now applies to any playback. Do not confuse this setting with `m_playback_mode`, which has a similar meaning but is more transitory. Probably, the concept needs some clean-up.

10.69.6.3 `m_start_from_perfedit`

```
bool seq64::perform::m_start_from_perfedit [private]
```

10.69.6.4 `m_reposition`

```
bool seq64::perform::m_reposition [private]
```

10.69.6.5 m_excell_FF_RW

```
float seq64::perform::m_excell_FF_RW [private]
```

It starts out at 1.0, and can range up to 60.0, being multiplied by 1.1 by the FF/RW timeout function.

10.69.6.6 m_FF_RW_button_type

```
ff_rw_button_t seq64::perform::m_FF_RW_button_type [private]
```

It has values of FF_RW_REWIND, FF_RW_NONE, or FF_RW_FORWARD. This was a free (global in a namespace) int in perfedit.

10.69.6.7 m_mute_group

```
bool seq64::perform::m_mute_group[c_max_sequence] [private]
```

This value determines whether a particular track will be muted or unmuted, and it can handle all tracks available in the application (currently `c_max_sets * c_seqs_in_set`, i.e. 1024). Note that the current state of playing can be "learned", and stored herein as the desired state for the track.

10.69.6.8 m_mute_group_rc

```
bool seq64::perform::m_mute_group_rc[c_max_sequence] [private]
```

10.69.6.9 m_armed_saved

```
bool seq64::perform::m_armed_saved [private]
```

10.69.6.10 m_armed_statuses

```
bool seq64::perform::m_armed_statuses[c_max_sequence] [private]
```

10.69.6.11 m_seqs_in_set

```
int seq64::perform::m_seqs_in_set [private]
```

This change requires some arrays to be dynamically allocated (vectors). This cannot be a constant, because we may need to change it after creating the perform object.

10.69.6.12 m_max_groups

```
int seq64::perform::m_max_groups [private]
```

This value is the maximum number of sequences we can store (c_max_sequence) divided by the number of sequences in a set.

Groups are a set of sequence-states. They are held in a linear array of size c_max_sequence, subdivided into groups of size m_seqs_in_set.

10.69.6.13 m_tracks_mute_state

```
std::vector<bool> seq64::perform::m_tracks_mute_state [private]
```

Unlike the m_mute_group[] array, this holds the current state, rather than the state desired by activating a mute group, and it applies to only one screen-set.

```
bool m_tracks_mute_state[c_seqs_in_set];
```

Please be aware that vector<bool> might be optimized to use bits, and so taking the address of an element of this vector will not work. But we don't need that kind of access, so we are safe here. We might try using a char at some point, to see how performance is affected.

10.69.6.14 m_mode_group

```
bool seq64::perform::m_mode_group [private]
```

This value starts out true. It is altered by the c_midi_control_mod_gmute handler or when the [keys\(\).group_off\(\)](#) or the [keys\(\).group_on\(\)](#) keys are struck.

10.69.6.15 m_mode_group_learn

```
bool seq64::perform::m_mode_group_learn [private]
```

10.69.6.16 m_mute_group_selected

```
int seq64::perform::m_mute_group_selected [private]
```

A "group" is essentially a "set" that is selected for the saving and restoring of the status of all patterns in that set. We're going to add a value of -1 (SEQ64_NO_MUTE_GROUP_SELECTED) to indicate the value should not be used.

10.69.6.17 m_midi_mute_group_present

```
bool seq64::perform::m_midi_mute_group_present [private]
```

We need to know if valid mute-groups are present when deciding whether or not to write them to the "rc" file.

10.69.6.18 m_seqs

```
sequence* seq64::perform::m_seqs[c_max_sequence] [private]
```

Todo First, make the sequence array a vector, and second, put all of these flags into a structure and access those members indirectly.

10.69.6.19 m_seqs_active

```
bool seq64::perform::m_seqs_active[c_max_sequence] [private]
```

This array can have "holes" with inactive sequences, so every sequence needs to be checked before using it.

10.69.6.20 m_was_active_main

```
bool seq64::perform::m_was_active_main[c_max_sequence] [private]
```

This value seems to be used only in maintaining dirtiness-status; did some process modify the sequence? Was it's mute/unmute status changed?

10.69.6.21 m_was_active_edit

```
bool seq64::perform::m_was_active_edit[c_max_sequence] [private]
```

This value seems to be used only in maintaining dirtiness-status for editing the mute/unmute status during pattern editing.

10.69.6.22 m_was_active_perf

```
bool seq64::perform::m_was_active_perf[c_max_sequence] [private]
```

This value seems to be used only in maintaining dirtiness-status for editing the mute/unmute status during performance/song editing.

10.69.6.23 m_was_active_names

```
bool seq64::perform::m_was_active_names[c_max_sequence] [private]
```

This value seems to be used only in maintaining dirtiness-status for editing the mute/unmute status during performance names editing. Not sure that it serves a real purpose; perhaps created with an eye to editing the pattern name in the song editor?

10.69.6.24 m_sequence_state

```
bool seq64::perform::m_sequence_state[c_max_sequence] [private]
```

10.69.6.25 m_screenset_state

```
std::vector<bool> seq64::perform::m_screenset_state [private]
```

This is used in the new queue-replace (queue-solo) feature.

```
bool m_screenset_state[c_seqs_in_set];
```

10.69.6.26 m_queued_replace_slot

```
int seq64::perform::m_queued_replace_slot [private]
```

This value is set to -1 when queue mode is exited. See the SEQ64_NO_QUEUED_SOLO value.

10.69.6.27 m_transpose

```
int seq64::perform::m_transpose [private]
```

10.69.6.28 m_out_thread

```
pthread_t seq64::perform::m_out_thread [private]
```

Provides a "handle" to the output thread.

10.69.6.29 m_in_thread

```
pthread_t seq64::perform::m_in_thread [private]
```

10.69.6.30 m_out_thread_launched

```
bool seq64::perform::m_out_thread_launched [private]
```

10.69.6.31 m_in_thread_launched

```
bool seq64::perform::m_in_thread_launched [private]
```

10.69.6.32 m_is_running

```
bool seq64::perform::m_is_running [private]
```

However, this flag is conflated with some JACK support, and we have to supplement it with another flag, `m_is_↔pattern_playing`.

10.69.6.33 m_is_pattern_playing

```
bool seq64::perform::m_is_pattern_playing [private]
```

It replaces `rc_settings::is_pattern_playing()`, which is gone, since the perform object is now visible to all classes that care about it.

10.69.6.34 m_inputting

```
bool seq64::perform::m_inputting [private]
```

10.69.6.35 m_outputting

```
bool seq64::perform::m_outputting [private]
```

10.69.6.36 m_looping

```
bool seq64::perform::m_looping [private]
```

If true, the performance will loop between the L and R markers in the performance editor.

10.69.6.37 m_song_recording

```
bool seq64::perform::m_song_recording [private]
```

10.69.6.38 m_song_record_snap

```
bool seq64::perform::m_song_record_snap [private]
```

10.69.6.39 m_resume_note_ons

```
bool seq64::perform::m_resume_note_ons [private]
```

10.69.6.40 m_current_tick

```
double seq64::perform::m_current_tick [private]
```

10.69.6.41 m_playback_mode

```
bool seq64::perform::m_playback_mode [private]
```

There are two, "live" and "song", indicated by the following values:

```
m_playback_mode == false:    live mode
m_playback_mode == true:     playback/song mode
```

10.69.6.42 m_ppqn

```
int seq64::perform::m_ppqn [private]
```

10.69.6.43 m_bpm

```
midibpm seq64::perform::m_bpm [private]
```

10.69.6.44 m_beats_per_bar

```
int seq64::perform::m_beats_per_bar [private]
```

The default value is SEQ64_DEFAULT_BEATS_PER_MEASURE (4).

10.69.6.45 m_beat_width

```
int seq64::perform::m_beat_width [private]
```

The default value is SEQ64_DEFAULT_BEAT_WIDTH (4).

10.69.6.46 m_tempo_track_number

```
int seq64::perform::m_tempo_track_number [private]
```

Normally 0, it can be changed to any value from 1 to 1023 via the tempo-track-number setting in the "rc" file, and that can be overridden by the c_tempo_track SeqSpec possibly present in the song's MIDI file.

10.69.6.47 m_clocks_per_metronome

```
int seq64::perform::m_clocks_per_metronome [private]
```

This value provides the number of MIDI clocks between metronome clicks. The default value of this item is 24. It can also be read from some SMF 1 files, such as our hymne.mid example.

10.69.6.48 m_32nds_per_quarter

```
int seq64::perform::m_32nds_per_quarter [private]
```

Useful in export. A duplicate of the same member in the sequence class.

10.69.6.49 m_us_per_quarter_note

```
long seq64::perform::m_us_per_quarter_note [private]
```

Useful in export. A duplicate of the same member in the sequence class.

10.69.6.50 m_master_bus

```
mastermidibus* seq64::perform::m_master_bus [private]
```

We changed this item to a pointer so that we can delay the creation of this object until after all settings have been read.

10.69.6.51 m_filter_by_channel

```
bool seq64::perform::m_filter_by_channel [private]
```

10.69.6.52 m_master_clocks

```
std::vector<clock_e> seq64::perform::m_master_clocks [private]
```

10.69.6.53 m_master_inputs

```
std::vector<bool> seq64::perform::m_master_inputs [private]
```

10.69.6.54 m_one_measure

```
midipulse seq64::perform::m_one_measure [private]
```

We can save some multiplications, and, more importantly, later define a more flexible definition of "one measure's worth" than simply four quarter notes.

10.69.6.55 m_left_tick

```
midipulse seq64::perform::m_left_tick [private]
```

Note that "tick" is actually "pulses".

10.69.6.56 m_right_tick

```
midipulse seq64::perform::m_right_tick [private]
```

Note that "tick" is actually "pulses".

10.69.6.57 m_starting_tick

```
midipulse seq64::perform::m_starting_tick [private]
```

By default, this value is always reset to the value of the "left tick". We want to eventually be able to leave it at the last playing tick, to support a "pause" functionality. Note that "tick" is actually "pulses".

10.69.6.58 m_tick

```
midipulse seq64::perform::m_tick [mutable], [private]
```

The m_tick member holds the tick to be used in displaying the progress bars and the maintime pill. It is mutable because sometimes we want to adjust it in a const function for pause functionality.

10.69.6.59 m_jack_tick

```
midipulse seq64::perform::m_jack_tick [private]
```

10.69.6.60 m_usemidiclock

```
bool seq64::perform::m_usemidiclock [private]
```

10.69.6.61 m_midiclockrunning

```
bool seq64::perform::m_midiclockrunning [private]
```

10.69.6.62 m_midiclocktick

```
int seq64::perform::m_midiclocktick [private]
```

10.69.6.63 m_midiclockpos

```
int seq64::perform::m_midiclockpos [private]
```

10.69.6.64 m_dont_reset_ticks

```
bool seq64::perform::m_dont_reset_ticks [private]
```

All this member is used for is keeping the last tick from being reset.

10.69.6.65 m_screenset_notepad

```
std::string seq64::perform::m_screenset_notepad[c_max_sets] [private]
```

10.69.6.66 m_midi_cc_toggle

```
midi_control seq64::perform::m_midi_cc_toggle[c_midi_controls_extended] [private]
```

10.69.6.67 m_midi_cc_on

```
midi_control seq64::perform::m_midi_cc_on[c_midi_controls_extended] [private]
```

10.69.6.68 m_midi_cc_off

```
midi_control seq64::perform::m_midi_cc_off[c_midi_controls_extended] [private]
```

10.69.6.69 m_control_status

```
int seq64::perform::m_control_status [private]
```

Need to learn more about this one. It is used in the replace, snapshot, and queue functionality.

10.69.6.70 m_screenset

```
int seq64::perform::m_screenset [private]
```

This is merely the screen-set that is in view. The fix of tdeagan substitutes the "in-view" screen-set for the "playing" screen-set.

10.69.6.71 m_screenset_offset

```
int seq64::perform::m_screenset_offset [private]
```

Saves some multiplications. It is used in the MIDI control of the playback status of the sequences in the current screen-set. It is also used to offset the sequence numbers so that the control (mute/unmute) keys can be shown on any screen-set.

10.69.6.72 m_playscreen

```
int seq64::perform::m_playscreen [private]
```

In seq24, this value is altered by [set_playing_screenset\(\)](#), which is called by [handle_midi_control\(c_midi_control↵_play_ss, state\)](#).

10.69.6.73 m_playscreen_offset

```
int seq64::perform::m_playscreen_offset [private]
```

Saves some multiplications, should make the code easier to grok, and centralizes the use of [c_seqs_in_set/m_↵seqs_in_set](#), which we want to be able to change at run-time, as a future enhancement.

10.69.6.74 m_max_sets

```
int seq64::perform::m_max_sets [private]
```

Again, currently set to the old value, which is used in hard-wired array sizes. To make it variable will require a move from arrays to vectors.

10.69.6.75 m_sequence_count

```
int seq64::perform::m_sequence_count [private]
```

Used by the [install_sequence\(\)](#) function. Note that this value is not a suitable replacement for `c_max_sequence/m_sequence_max`, because there can be inactive sequences amidst the active sequences. See the `m_sequence_limit` member.

10.69.6.76 m_sequence_max

```
int seq64::perform::m_sequence_max [private]
```

However, this value is already $32 * 32 = 1024$, and is probably enough for any usage. Famous last words?

10.69.6.77 m_sequence_high

```
int seq64::perform::m_sequence_high [private]
```

This value starts as 0, to indicate no sequences loaded, and then contains the highest sequence number hitherto loaded, plus 1 so that it can be used as a for-loop limit similar to `m_sequence_max`. It's maximum value should be `m_sequence_max` (`c_max_sequence`).

Currently meant only for limited context to try to squeeze a little extra speed out of playback. There's no easy way to lower this value when the highest sequence is deleted, though.

10.69.6.78 m_edit_sequence

```
int seq64::perform::m_edit_sequence [private]
```

Moving this status from `seqmenu` into `perform` for better centralized management.

10.69.6.79 m_is_modified

```
bool seq64::perform::m_is_modified [private]
```

All the GUIs seem to use a `perform` object.

10.69.6.80 m_condition_var

```
condition_var seq64::perform::m_condition_var [private]
```

It is signalled if playback has been started. The output thread function waits on this variable until `m_is_running` and `m_outputing` are false. This variable is also signalled in the perform destructor.

10.69.6.81 m_jack_asst

```
jack_assistant seq64::perform::m_jack_asst [private]
```

It implements most of the JACK stuff.

10.69.6.82 m_have_undo

```
bool seq64::perform::m_have_undo [private]
```

10.69.6.83 m_undo_vect

```
std::vector<int> seq64::perform::m_undo_vect [private]
```

See the [push_trigger_undo\(\)](#) function.

10.69.6.84 m_have_redo

```
bool seq64::perform::m_have_redo [private]
```

10.69.6.85 m_redo_vect

```
std::vector<int> seq64::perform::m_redo_vect [private]
```

See the [pop_trigger_undo\(\)](#) function.

10.69.6.86 m_notify

```
std::vector<performcallback *> seq64::perform::m_notify [private]
```

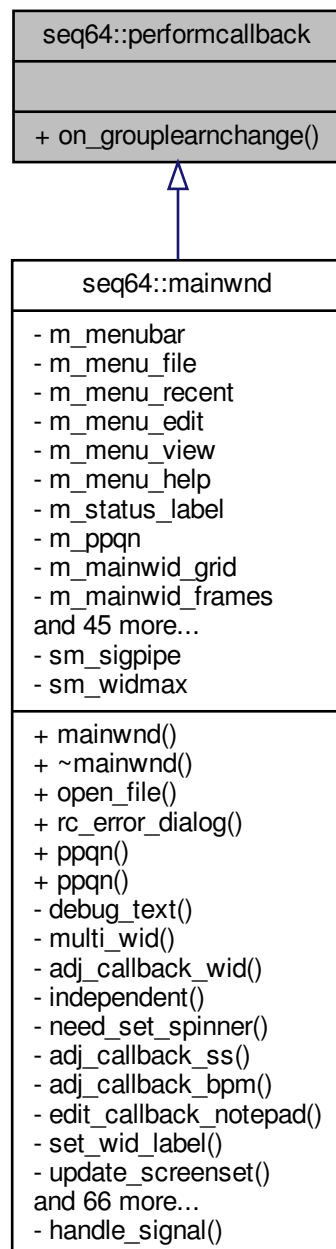
10.69.6.87 m_gui_support

```
gui_assistant& seq64::perform::m_gui_support [private]
```

10.70 seq64::performcallback Struct Reference

Provides for notification of events.

Inheritance diagram for seq64::performcallback:



Public Member Functions

- virtual void `on_grouplearnchange` (bool)
A do-nothing callback.

10.70.1 Detailed Description

Provide a response to a group-learn change event.

10.70.2 Member Function Documentation

10.70.2.1 on_grouplearnchange()

```
virtual void seq64::performcallback::on_grouplearnchange (  
    bool ) [inline], [virtual]
```

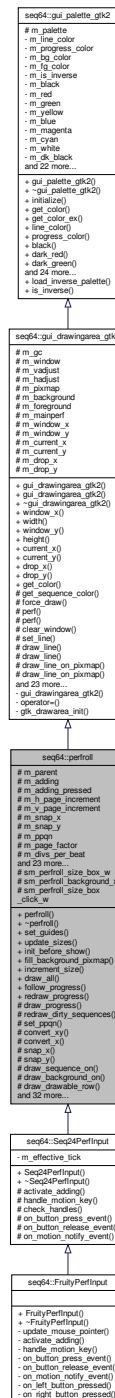
"state" is an Unused parameter.

Reimplemented in [seq64::mainwnd](#).

10.71 seq64::perfroll Class Reference

This class implements the performance roll user interface.

Inheritance diagram for seq64::perfroll:



Public Member Functions

- [perfroll](#) ([perform](#) &[perf](#), [perfedit](#) &parent, Gtk::Adjustment &hadjust, Gtk::Adjustment &vadjust, int ppqn=S↳EQ64_USE_DEFAULT_PPQN)

Principal constructor.

- virtual [~perfroll](#) ()

This destructor deletes the interaction object.

- void [set_guides](#) (int snap, int measure, int beat)
This function sets the `m_snap_x`, `m_measure_length`, and `m_beat_length` members directly from the function parameters, which are in units of pulses (sometimes misleadingly called "ticks").
- void [update_sizes](#) ()
Updates the sizes of various items.
- void [init_before_show](#) ()
Sets the roll-lengths ticks member.
- void [fill_background_pixmap](#) ()
This function updates the background of the piano roll.
- void [increment_size](#) ()
*Increments the value of `m_roll_length_ticks` by the `PPQN * 512`, then calls [update_sizes\(\)](#).*
- void [draw_all](#) ()
Provides a very common sequence of calls used in `perfroll_input`.
- void [follow_progress](#) ()
Checks the position of the tick, and, if it is in a different piano-roll "page" than the last page, moves the page to the next page.
- void [redraw_progress](#) ()
Helper function to simplify the client call.

Protected Member Functions

- void [draw_progress](#) ()
Draws the progress line that shows where we are in the performance.
- void [redraw_dirty_sequences](#) ()
Redraws patterns/sequences that have been modified.
- void [set_ppqn](#) (int ppqn)
Handles changes to the PPQN value in one place.
- void [convert_xy](#) (int x, int y, [midipulse](#) &tick, int &seq)
Converts (x, y) coordinates on the piano roll to tick (pulse) and sequence numbers.
- void [convert_x](#) (int x, [midipulse](#) &tick)
Converts an x-coordinate to a tick-offset on the x axis.
- void [snap_x](#) (int &x)
This function performs a 'snap' action on x.
- void [snap_y](#) (int &y)
This function performs a 'snap' action on y.
- void [draw_sequence_on](#) (int seqnum)
Draws the given pattern/sequence on the given drawable area.
- void [draw_background_on](#) (int seqnum)
Draws the given pattern/sequence background on the given drawable area.
- void [draw_drawable_row](#) (int y)
Not quite sure what this draws yet.
- void [draw_sequence](#) (int seqnum)
To be used in iterating through a set.
- void [offset_sequence](#) (int seqnum, [midipulse](#) offset)
- void [change_horz](#) ()
Changes the 4-bar horizontal offset member and queues up a draw operation.
- void [change_vert](#) ()
Changes the vertical offset member and queues up a draw operation.
- void [split_trigger](#) (int [sequence](#), [midipulse](#) tick)
Splits a trigger, whatever that means.

- void [enqueue_draw](#) ()
Wraps queue_draw() and forwards the call to the parent perfrdit, so that it can forward it to any other perfrdit that exists.
- void [set_zoom](#) (int z)
Implements the horizontal zoom feature.
- void [convert_drop_xy](#) ()
A convenience function.
- void [horizontal_adjust](#) (double step)
This function provides optimization for the [on_scroll_event\(\)](#) function.
- void [vertical_adjust](#) (double step)
This function provides optimization for the [on_scroll_event\(\)](#) function.
- void [horizontal_set](#) (double value)
Sets the exact position of a horizontal scroll-bar.
- void [vertical_set](#) (double value)
Sets the exact position of a vertical scroll-bar.
- virtual void [activate_adding](#) (bool adding)=0
- virtual bool [handle_motion_key](#) (bool is_left)=0
- bool [is_adding](#) () const
'Getter' function for member m_adding
- void [set_adding](#) (bool flag)
'Setter' function for member m_adding
- bool [is_adding_pressed](#) () const
'Getter' function for member m_adding_pressed
- void [set_adding_pressed](#) (bool flag)
'Setter' function for member m_adding_pressed
- bool [growing](#) () const
'Getter' function for member m_growing
- bool [moving](#) () const
'Getter' function for member m_moving
- bool [drop_action](#) () const
Indicates if we're moving.
- virtual void [on_realize](#) ()
Provides the on-realization callback.
- virtual bool [on_expose_event](#) (GdkEventExpose *ev)
Handles the on-expose event.
- virtual bool [on_button_press_event](#) (GdkEventButton *ev)
This callback function handles the follow-on work of a button press, and is called by overridden versions such as [Seq24PerfrInput::on_button_press_event\(\)](#)
- virtual bool [on_button_release_event](#) (GdkEventButton *ev)
This callback function handles a button release by forwarding it to the interaction object's button-release function.
- virtual bool [on_motion_notify_event](#) (GdkEventMotion *ev)
Handles motion notification by forwarding it to the interaction object's motion-notification callback function.
- virtual bool [on_scroll_event](#) (GdkEventScroll *ev)
Handles horizontal and vertical scrolling.
- virtual bool [on_focus_in_event](#) (GdkEventFocus *ev)
This callback handles an in-focus event by setting the flag to HAS_FOCUS.
- virtual bool [on_focus_out_event](#) (GdkEventFocus *ev)
This callback handles an out-of-focus event by resetting the flag HAS_FOCUS.
- virtual void [on_size_allocate](#) (Gtk::Allocation &al)
Upon a size allocation event, this callback calls the base-class version of this function, then sets m_window_x and m_window_y, and calls [update_sizes\(\)](#).

- virtual bool [on_key_press_event](#) (GdkEventKey *ev)
This callback function handles a key-press event.
- virtual void [on_size_request](#) (GtkRequisition *)
This do-nothing callback effectively throws away a size request.

Protected Attributes

- [perfedit](#) & [m_parent](#)
Provides a link to the perfedit that created this object.
- bool [m_adding](#)
Indicates we are in the middle of adding a sequence segment to the performance.
- bool [m_adding_pressed](#)
Indicates if the left mouse button is pressed while in adding mode.
- int [m_h_page_increment](#)
Provides the horizontal page increment for the horizontal scrollbar.
- int [m_v_page_increment](#)
Provides the vertical page increment for the vertical scrollbar.
- int [m_snap_x](#)
Amount of horizontal snap, pulses.
- int [m_snap_y](#)
The amount of vertical snap.
- int [m_ppqn](#)
Parts-per-quarter-note value.
- int [m_page_factor](#)
4096, horizontal page sizing.
- int [m_divs_per_beat](#)
Holds current tick scaling value.
- midipulse [m_ticks_per_bar](#)
Holds current bar scaling value.
- int [m_perf_scale_x](#)
Scaling based on zoom and PPQN.
- int [m_w_scale_x](#)
Scaling based on zoom and PPQN.
- int [m_zoom](#)
New value to attempt a rudimentary time-zoom feature.
- int [m_names_y](#)
The maximum height of the perfroll names box, in pixels.
- int [m_background_x](#)
The width of the perfroll background.
- int [m_size_box_w](#)
This is a basically constant value set to `s_perfroll_size_box_w = 3`.
- int [m_measure_length](#)
The length of a measure, in beat units.
- int [m_beat_length](#)
The length of a beat, in parts-per-quarter note.
- midipulse [m_old_progress_ticks](#)
Saves the position of the progress bar, for erasing it in preparation for drawing it at the next tick value.
- int [m_scroll_page](#)
Provides the current scroll page in which the progress bar resides.
- bool [m_have_button_press](#)

- Used in the fruity and seq24 perfroll input classes to help with trigger push/pop management.*
- [midipulse m_4bar_offset](#)
Holds the horizontal offset related to the horizontal scroll-bar position.
- [int m_sequence_offset](#)
This value is the vertical version of m_4bar_offset.
- [int m_roll_length_ticks](#)
Provides the width of the piano roll in ticks.
- [midipulse m_drop_tick](#)
The horizontal location for section movement.
- [midipulse m_drop_tick_offset](#)
The horizontal trigger location for section movement.
- [int m_drop_sequence](#)
Holds the currently-selected sequence being moved.
- [int m_sequence_max](#)
Currently, just a class-specific version of c_max_sequence, meant for the future.
- [bool m_sequence_active](#) [[c_max_sequence](#)]
Used when drawing an active sequence.
- [bool m_moving](#)
Used in the Seq24 or Fruity processing when moving a section of triggers.
- [bool m_growing](#)
Used in the Seq24 or Fruity processing when growing a section of triggers.
- [bool m_grow_direction](#)
Used in the Seq24 or Fruity processing when growing a section of triggers.

Static Protected Attributes

- [static int sm_perfroll_size_box_w](#)
Static sizing members for initial zoom of the perfroll.
- [static int sm_perfroll_background_x](#)
sm_perfroll_background_x is copied into m_background_x; it gets adjusted in [perfroll::set_ppqn\(\)](#).
- [static int sm_perfroll_size_box_click_w](#)
Provides sizing information for the perfroll-derived classes, using in the [Seq24PerfrInput](#) and [FruityPerfrInput](#) classes.

Friends

- [class perfedit](#)
These friend implement interaction-specific behavior, although only the Seq24 interactions support full keyboard processing, except for some common functionality provided by [perform::perfroll_key_event\(\)](#).

Additional Inherited Members

10.71.1 Constructor & Destructor Documentation

10.71.1.1 perfroll()

```
seq64::perfroll::perfroll (
    perform & p,
    perfedit & parent,
    Gtk::Adjustment & hadjust,
    Gtk::Adjustment & vadjust,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

Parameters

<i>p</i>	The perform object that this class will affect via user interaction.
<i>parent</i>	The perfedit object that holds this user-interface class.
<i>hadjust</i>	A horizontal adjustment object, used here to control scrolling.
<i>vadjust</i>	A vertical adjustment object, used here to control scrolling.
<i>ppqn</i>	The "resolution" of the MIDI file, used in zooming and scaling.

10.71.1.2 `~perfroll()`

```
seq64::perfroll::~~perfroll ( ) [virtual]
```

Well, now there are two objects, so no explicit deletion necessary.

10.71.2 Member Function Documentation

10.71.2.1 `set_guides()`

```
void seq64::perfroll::set_guides (
    int snap,
    int measure,
    int beat )
```

This function then fills in the background, and queues up a draw operation.

Parameters

<i>snap</i>	Provides the number of snap-pulses (pulses per snap interval) as calculated in perfedit::set_guides() . This is actually equal to the measure-pulses divided by the snap value in perfedit; the snap value defaults to 8.
<i>measure</i>	Provides the number of measure-pulses (pulses per measure) as calculated in perfedit::set_guides() .
<i>beat</i>	Provides the number of beat-pulses (pulses per beat) as calculated in perfedit::set_guides() .

10.71.2.2 `update_sizes()`

```
void seq64::perfroll::update_sizes ( )
```

Note

Trying to figure out what the 16 is. So take the "bars-visible" calculation, the `c_perf_scale_x` value, assume that "ticks" is another name for "pulses", and assume that "beats" is a quarter note. Ignoring the numbers, the units come out to:

$$\text{bars} = \frac{\text{pixels} * \text{ticks} / \text{pixel}}{\text{ticks} / \text{beat} * \text{beats} / \text{bar}}$$

Thus, the 16 is a "beats per bar" or "beats per measure" value. This doesn't quite make sense, but there are 16 divisions per beat on the perffroll user-interface. So for now we'll call it the latter, and make a variable called "m_divs_per_beat", see its definition in the class initializer list.

10.71.2.3 init_before_show()

```
void seq64::perffroll::init_before_show ( )
```

First, it gets the largest trigger value among the active sequences. Then it truncates this value to the nearest PPQN * 16 ticks. Then it adds PPQN * 4096 ticks.

10.71.2.4 fill_background_pixmap()

```
void seq64::perffroll::fill_background_pixmap ( )
```

The first thing done is to clear the background by painting it with a filled white rectangle.

This function is called whenever something occurs (e.g. zoom) that can affect how the piano roll is drawn.

10.71.2.5 increment_size()

```
void seq64::perffroll::increment_size ( )
```

10.71.2.6 draw_all()

```
void seq64::perffroll::draw_all ( )
```

`m_drop_y` is adjusted by [perffroll::change_vert\(\)](#) for any scroll after it was originally selected. The call below to [draw_drawable_row\(\)](#) will have the wrong y location and un-select will not occur if the user scrolls the track up or down to a new y location, if not adjusted.

For the box set/selections, we create a `perform::SeqOperation` functor by binding [&perffroll::draw_sequence\(\)](#) to this object and passing it to `perform::selection_operation()`. The whole set of operations replaces the single operation of the "drop" sequence.

Note:

We could bind additional parameters as needed, either as place-holders or constant parameter values.

10.71.2.7 follow_progress()

```
void seq64::perfroll::follow_progress ( )
```

10.71.2.8 redraw_progress()

```
void seq64::perfroll::redraw_progress ( ) [inline]
```

10.71.2.9 draw_progress()

```
void seq64::perfroll::draw_progress ( ) [protected]
```

We would like to be able to leave the line there when the progress is paused while running off of JACK transport. How? The `perf().get_tick()` call always returns 0 when stop is in force.

If we comment out the erasure of the old line, we see that the progress bar is also erased when a pattern boundary is hit (triggers), and when the sequence is stopped by the user.

In order to support true pause in the song editor, we tried to replace `perform::get_tick()` with `perform::get_start_tick()` and `perform::get_last_tick()` [a new experimental function]. But those replacements here always return 0, even as `perform::get_tick()` increases. (Note that the `draw_progress` function is called at every timeout, that is, constantly.)

10.71.2.10 redraw_dirty_sequences()

```
void seq64::perfroll::redraw_dirty_sequences ( ) [protected]
```

Change Note ca 2016-05-30 Lets try not drawing sequences greater than the maximum, at all.

10.71.2.11 set_ppqn()

```
void seq64::perfroll::set_ppqn (
    int ppqn ) [protected]
```

The `m_ticks_per_bar` member replaces the global `ppqn` times 16. This construct is parts-per-quarter-note times 4 quarter notes times 4 sixteenth notes in a bar. (We think...)

The `m_perf_scale_x` member starts out at `c_perf_scale_x`, which is 32 ticks per pixel at the default tick rate of 192 PPQN. We adjust this now. But note that this calculation still involves the `c_perf_scale_x` constant.

Todo Resolve the issue of `c_perf_scale_x` versus `m_perf_scale_x` in `perfroll`.

10.71.2.12 convert_xy()

```
void seq64::perfroll::convert_xy (
    int x,
    int y,
    midipulse & d_tick,
    int & d_seq ) [protected]
```

The results are returned via the `d_tick` and `d_seq` parameters. The sequence number is clipped to a legal value (0 to `m_sequence_max`).

Parameters

	<i>x</i>	The x coordinate of the mouse pointer.
	<i>y</i>	The y coordinate of the mouse pointer.
out	<i>d_tick</i>	Holds the calculated tick value.
out	<i>d_seq</i>	Holds the calculated sequence-number value.

10.71.2.13 convert_x()

```
void seq64::perfroll::convert_x (
    int x,
    midipulse & tick ) [protected]
```

The result is returned via the tick parameter. Note that `m_4bar_offset` already includes the `m_ticks_per_bar = ppqn * 16` factor, for speed.

Parameters

	<i>x</i>	The input x (pixel) value.
out	<i>tick</i>	Holds the result of the calculation.

10.71.2.14 snap_x()

```
void seq64::perfroll::snap_x (
    int & x ) [protected]
```

- `m_snap_x` = number pulses to snap to
- `m_perf_scale_x` = number of pulses per pixel

Therefore `mod = m_snap_x/m_perf_scale_x` equals the number pixels to snap to.

Parameters

out	<i>x</i>	The destination for the x-snap calculation.
-----	----------	---

10.71.2.15 snap_y()

```
void seq64::perfroll::snap_y (
    int & y ) [protected]
```

We don't do vertical zoom, so this function is simpler than [snap_x\(\)](#).

Parameters

out	y	The destination for the y-snap calculation.
-----	---	---

10.71.2.16 draw_sequence_on()

```
void seq64::perfroll::draw_sequence_on (
    int seqnum ) [protected]
```

Statement nesting from hell! Items drawn on the Song editor piano roll:

1. Main trigger box (also called a "segment") background.
2. Trigger outline (the rectangle around a "segment").
3. The left hand side little sequence grab handle, or segment handle.
4. The right-side segment handle.

```
printf ( "seq %d trigger: %s\n", seqnum, selected ? "selected" : "unselected" );
```

10.71.2.17 draw_background_on()

```
void seq64::perfroll::draw_background_on (
    int seqnum ) [protected]
```

10.71.2.18 draw_drawable_row()

```
void seq64::perfroll::draw_drawable_row (
    int y ) [protected]
```

It is involved in the drawing of a greyed (selected) row.

What's weird is that we divide y by m_names_y, then multiply it by m_names_y, before passing the result to [draw_drawable\(\)](#). However, if we just use y casted to an int, then the drawing of the row is only partial, vertically.

10.71.2.19 draw_sequence()

```
void seq64::perfroll::draw_sequence (
    int seqnum ) [inline], [protected]
```

10.71.2.20 offset_sequence()

```
void seq64::perfroll::offset_sequence (
    int seqnum,
    midipulse offset ) [protected]
```

10.71.2.21 change_horz()

```
void seq64::perfroll::change_horz ( ) [protected]
```

Since the `m_4bar_offset` value is always multiplied by `m_ticks_per_bar` before usage, let's just do it here and not have to multiply it later.

10.71.2.22 change_vert()

```
void seq64::perfroll::change_vert ( ) [protected]
```

Stazed:

Must adjust `m_drop_y` or `perfroll_input`'s `unselect_triggers()` will not work if scrolled up or down to a new location. See the note in `on_button_press_event()` in the `perfroll_input` module. Also see the note in the `draw_all()` function.

10.71.2.23 split_trigger()

```
void seq64::perfroll::split_trigger (
    int sequence,
    midipulse tick ) [protected]
```

10.71.2.24 enqueue_draw()

```
void seq64::perfroll::enqueue_draw ( ) [protected]
```

The parent `perfed`it will call `perfroll::queue_draw()` on behalf of this object, and it will pass a `perfroll::enqueue_draw()` to the peer `perfed`it's `perfroll`, if the peer exists.

10.71.2.25 set_zoom()

```
void seq64::perfroll::set_zoom (
    int z ) [protected]
```

Change Note ca 2016-04-05 The initial zoom value is `c_perf_scale_x` (32). We allow it to range from 1 to 128, for now. Smaller values zoom in.

10.71.2.26 `convert_drop_xy()`

```
void seq64::perfroll::convert_drop_xy ( ) [inline], [protected]
```

10.71.2.27 `horizontal_adjust()`

```
void seq64::perfroll::horizontal_adjust (
    double step ) [inline], [protected]
```

A duplicate of the one in `seqroll`.

Parameters

<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. See gui_drawingarea_gtk2::scroll_hadjust() for more information.
-------------	---

10.71.2.28 `vertical_adjust()`

```
void seq64::perfroll::vertical_adjust (
    double step ) [inline], [protected]
```

A near-duplicate of the one in `seqroll`.

Parameters

<i>step</i>	Provides the step value to use for adjusting the vertical scrollbar. See gui_drawingarea_gtk2::scroll_vadjust() for more information.
-------------	---

10.71.2.29 `horizontal_set()`

```
void seq64::perfroll::horizontal_set (
    double value ) [inline], [protected]
```

Parameters

<i>value</i>	The desired position. Mostly this is either 0.0 or 9999999.0 (an "infinite" value to select the start or end position).
--------------	---

10.71.2.30 vertical_set()

```
void seq64::perfroll::vertical_set (
    double value ) [inline], [protected]
```

Parameters

<i>value</i>	The desired position. Mostly this is either 0.0 or 9999999.0 (an "infinite" value to select the start or end position).
--------------	---

10.71.2.31 activate_adding()

```
virtual void seq64::perfroll::activate_adding (
    bool adding ) [protected], [pure virtual]
```

Implemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

10.71.2.32 handle_motion_key()

```
virtual bool seq64::perfroll::handle_motion_key (
    bool is_left ) [protected], [pure virtual]
```

Implemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

10.71.2.33 is_adding()

```
bool seq64::perfroll::is_adding ( ) const [inline], [protected]
```

10.71.2.34 set_adding()

```
void seq64::perfroll::set_adding (
    bool flag ) [inline], [protected]
```

10.71.2.35 is_adding_pressed()

```
bool seq64::perfroll::is_adding_pressed ( ) const [inline], [protected]
```

10.71.2.36 set_adding_pressed()

```
void seq64::perfroll::set_adding_pressed (
    bool flag ) [inline], [protected]
```

10.71.2.37 growing()

```
bool seq64::perfroll::growing ( ) const [inline], [protected]
```

10.71.2.38 moving()

```
bool seq64::perfroll::moving ( ) const [inline], [protected]
```

10.71.2.39 drop_action()

```
bool seq64::perfroll::drop_action ( ) const [inline], [protected]
```

Returns

Returns true if one of those two flags are set.

10.71.2.40 on_realize()

```
void seq64::perfroll::on_realize ( ) [protected], [virtual]
```

Calls the base-class version first.

Then it allocates the additional resources need, that couldn't be initialized in the constructor, and makes some connections.

Stazed:

This creation of `m_background` needs to be set to the max width for proper drawing of zoomed measures or they will get truncated with high beats per measure and low beat width. Since this is a constant size, it cannot be adjusted later for zoom. The constant `c_perfroll_background_x` is set to the max amount by default for use here. The drawing functions `fill_background_pixmap()` and `draw_background_on()` which use `c_perfroll_background_x` also, could be adjusted by zoom with a substituted variable. Not sure if there is any benefit to doing the adjustment... Perhaps a small benefit in speed? Maybe FIXME if really, really bored...

10.71.2.41 on_expose_event()

```
bool seq64::perfroll::on_expose_event (
    GdkEventExpose * ev ) [protected], [virtual]
```

Draws a vertical page of the performance editor. The part drawn starts at `m_sequence_offset` and continues until the last sequence that can be at least partially seen given the height of the window.

If we're at the bottom of the sequences (1024, a non-existent sequence) would be the last sequence shown, we don't bother drawing it. This prevents debug messages about an illegal sequence, and can show a black bottom row that is a clear sign we're at the end of the legal sequences.

Parameters

<code>ev</code>	Provides the expose event.
-----------------	----------------------------

Returns

Always returns true.

10.71.2.42 on_button_press_event()

```
bool seq64::perfroll::on_button_press_event (
    GdkEventButton * ev ) [protected], [virtual]
```

One minor issue: Fruity behavior doesn't yet provide the keystroke behavior we now handle for the Seq24 mode of operation.

Reimplemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

10.71.2.43 on_button_release_event()

```
bool seq64::perfroll::on_button_release_event (
    GdkEventButton * ev ) [protected], [virtual]
```

This gives us Seq24 versus Fruity behavior.

Reimplemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

10.71.2.44 on_motion_notify_event()

```
bool seq64::perfroll::on_motion_notify_event (
    GdkEventMotion * ev ) [protected], [virtual]
```

Actually, this is backwards; this function is called within the derived object's override.

Reimplemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

10.71.2.45 on_scroll_event()

```
bool seq64::perfroll::on_scroll_event (
    GdkEventScroll * ev ) [protected], [virtual]
```

If the Shift key is held while scrolling, then the scrolling is horizontal, otherwise it is vertical. This matches the convention of the seqedit class.

Note that, unlike the seqedit class, Ctrl-Scroll is not used to modify the zoom value. Rather than mess up legacy behavior, we will rely on keystrokes (z, 0, Z, and Ctrl-Page-Up and Ctrl-Page-Down) to implement this zoom.

Parameters

<code>ev</code>	Provides the scroll event.
-----------------	----------------------------

Returns

Currently always returns true.

10.71.2.46 on_focus_in_event()

```
bool seq64::perfrroll::on_focus_in_event (
    GdkEventFocus * ev ) [protected], [virtual]
```

10.71.2.47 on_focus_out_event()

```
bool seq64::perfrroll::on_focus_out_event (
    GdkEventFocus * ev ) [protected], [virtual]
```

10.71.2.48 on_size_allocate()

```
void seq64::perfrroll::on_size_allocate (
    Gtk::Allocation & al ) [protected], [virtual]
```

10.71.2.49 on_key_press_event()

```
bool seq64::perfrroll::on_key_press_event (
    GdkEventKey * ev ) [protected], [virtual]
```

If we don't check the event type first, then the `ev->keyval` value is something weird like 65507. Note that we pass the functionality on to the [perform::perfrroll_key_event\(\)](#) function for the handling of delete, cut, copy, paste, and undo operations. If the keystroke is not handled by that function, then we handle it here.

Note that only the Seq24 input interaction object handles additional keystrokes not handled by the `perfrroll_key_↔event()` function.

The `perfrroll_key_event()` call handles Del, Ctrl-X, Ctrl-C, Ctrl-V, and Ctrl-Z (which does nothing at present).

We've also added support for moving up and down in the piano roll (Up and Down arrows), paging up and down (Page-Up and Page-Down keys), paging left and right (Shift Page-Up and Page-Down), paging to top and bottom (Home and End), and paging to start and end (Shift Home and End).

The Keypad-End key is an issue on our ASUS "gaming" laptop. Whether it is seen as a "1" or an "End" key depends on an interaction between the Shift and the Num Lock key. Annoying, takes some time to get used to.

Stazed: there are many changes from seq32 that need to be studied before including them here. Note that, even though we filter out the Ctrl key here, it still works for Ctrl-X (cut) and Ctrl-V (paste). For undo, the Undo button can be used, Ctrl-Z never worked in this view anyway.

Warning

We see that 'x' and 'z' are already handled in `perfrroll_key_event()` if the Ctrl key was pressed. Be careful.

10.71.2.50 on_size_request()

```
virtual void seq64::perftroll::on_size_request (
    GtkRequisition * ) [inline], [protected], [virtual]
```

10.71.3 Friends And Related Function Documentation

10.71.3.1 perfedit

```
friend class perfedit [friend]
```

The perfedit class needs access to the private [enqueue_draw\(\)](#) function.

10.71.4 Field Documentation

10.71.4.1 sm_perftroll_size_box_w

```
int seq64::perftroll::sm_perftroll_size_box_w [static], [protected]
```

Static (private) convenience values.

We need to be able to adjust `sm_perftroll_background_x` per the selected PPQN value. This adjustment is made in the constructor, and assigned to the [perftroll::m_background_x](#) member.

`sm_perftroll_size_box_w` is copied into the `m_size_box_w` member, and is the width of the small square handle in the corner of each trigger segment. It gets adjusted in [perftroll::set_ppqn\(\)](#).

10.71.4.2 sm_perftroll_background_x

```
int seq64::perftroll::sm_perftroll_background_x [static], [protected]
```

10.71.4.3 sm_perftroll_size_box_click_w

```
int seq64::perftroll::sm_perftroll_size_box_click_w [static], [protected]
```

10.71.4.4 m_parent

```
perfedit& seq64::perfroll::m_parent [protected]
```

We want to support two perfedit windows, but the children of perfedit will have to communicate changes requiring a redraw through the parent.

10.71.4.5 m_adding

```
bool seq64::perfroll::m_adding [protected]
```

Moved from AbstractPerfInput.

10.71.4.6 m_adding_pressed

```
bool seq64::perfroll::m_adding_pressed [protected]
```

Moved from AbstractPerfInput.

10.71.4.7 m_h_page_increment

```
int seq64::perfroll::m_h_page_increment [protected]
```

It was set to 1, the same as the step increment. That is too little. This value will be set to 4, for now. Might be a useful "user" configuration option.

10.71.4.8 m_v_page_increment

```
int seq64::perfroll::m_v_page_increment [protected]
```

It was set to 1, the same as the step increment. That is too little. This value will be set to 8, for now. Might be a useful "user" configuration option.

10.71.4.9 m_snap_x

```
int seq64::perfroll::m_snap_x [protected]
```

10.71.4.10 m_snap_y

```
int seq64::perfroll::m_snap_y [protected]
```

10.71.4.11 m_ppqn

```
int seq64::perfroll::m_ppqn [protected]
```

10.71.4.12 m_page_factor

```
int seq64::perfroll::m_page_factor [protected]
```

10.71.4.13 m_divs_per_beat

```
int seq64::perfroll::m_divs_per_beat [protected]
```

10.71.4.14 m_ticks_per_bar

```
midipulse seq64::perfroll::m_ticks_per_bar [protected]
```

10.71.4.15 m_perf_scale_x

```
int seq64::perfroll::m_perf_scale_x [protected]
```

10.71.4.16 m_w_scale_x

```
int seq64::perfroll::m_w_scale_x [protected]
```

10.71.4.17 m_zoom

```
int seq64::perfroll::m_zoom [protected]
```

It seems to work pretty well now.

10.71.4.18 m_names_y

```
int seq64::perfroll::m_names_y [protected]
```

This is currently semantically a constant set to `c_names_y = 24`.

10.71.4.19 m_background_x

```
int seq64::perfroll::m_background_x [protected]
```

This is based on the m_ppqn value and the value of c_perf_scale_x (or is m_perf_scale_x preferable?)

10.71.4.20 m_size_box_w

```
int seq64::perfroll::m_size_box_w [protected]
```

It is used in drawing the short lines of the small box that sits at the top-left and bottom-right corners of each segment in the pattern editor. These can be used to lengthen and shorten a section in the song editor. We will increase this size, perhaps double it, to make it easier to grab.

10.71.4.21 m_measure_length

```
int seq64::perfroll::m_measure_length [protected]
```

10.71.4.22 m_beat_length

```
int seq64::perfroll::m_beat_length [protected]
```

10.71.4.23 m_old_progress_ticks

```
midipulse seq64::perfroll::m_old_progress_ticks [protected]
```

See the [draw_progress\(\)](#) function. This could almost be static inside that function.

10.71.4.24 m_scroll_page

```
int seq64::perfroll::m_scroll_page [protected]
```

10.71.4.25 m_have_button_press

```
bool seq64::perfroll::m_have_button_press [protected]
```

10.71.4.26 m_4bar_offset

```
midipulse seq64::perfroll::m_4bar_offset [protected]
```

Used in drawing the progress bar and the sequence events. Also used in [convert_x\(\)](#) and [convert_xy\(\)](#). This used to be the offset in units of bar ticks, but now we use it as a full-fledged ticks value. See the [change_horz\(\)](#) function.

10.71.4.27 m_sequence_offset

```
int seq64::perfroll::m_sequence_offset [protected]
```

It is obtained or changed when the vertical scroll-bar moves. It is used for drawing the correct vertical window in the piano roll.

10.71.4.28 m_roll_length_ticks

```
int seq64::perfroll::m_roll_length_ticks [protected]
```

Calculated in [init_before_show\(\)](#) based on the maximum trigger found in the perform object, the ticks/bar, the P↔PQN, and the page factor. Also can be increased in size in the [increment_size\(\)](#) function [tied to the Grow button]. Used in [update_sizes\(\)](#).

10.71.4.29 m_drop_tick

```
midipulse seq64::perfroll::m_drop_tick [protected]
```

Used only by the friend modules `perfroll_input` and `fruityperfroll_input`.

10.71.4.30 m_drop_tick_offset

```
midipulse seq64::perfroll::m_drop_tick_offset [protected]
```

Used only by the modules `perfroll_input` and `fruityperfroll_input`.

10.71.4.31 m_drop_sequence

```
int seq64::perfroll::m_drop_sequence [protected]
```

Used for redrawing the sequence.

Extension?

We would like to extend this to a list of sequencens so that we could move more than one sequence at once.

10.71.4.32 m_sequence_max

```
int seq64::perfroll::m_sequence_max [protected]
```

10.71.4.33 m_sequence_active

```
bool seq64::perfroll::m_sequence_active[c_max_sequence] [protected]
```

Not sure yet why we can't just use the sequence's member function to access this status boolean.

10.71.4.34 m_moving

```
bool seq64::perfroll::m_moving [protected]
```

10.71.4.35 m_growing

```
bool seq64::perfroll::m_growing [protected]
```

10.71.4.36 m_grow_direction

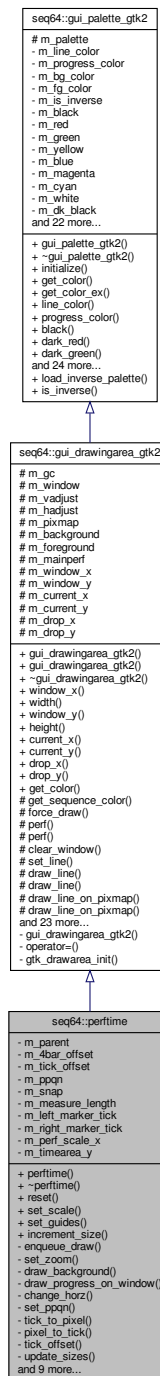
```
bool seq64::perfroll::m_grow_direction [protected]
```

Determines whether the section is growing to the left or to the right.

10.72 seq64::perftime Class Reference

This class implements drawing the piano time at the top of the "performance window" (the "song editor").

Inheritance diagram for seq64::perftime:



Public Member Functions

- [perftime](#) ([perform](#) &[perf](#), [perfedit](#) &parent, Gtk::Adjustment &hadjust, int ppqn=SEQ64_USE_DEFAULT_P↵PQN)

Principal constructor.

- virtual [~perftime](#) ()

Let's provide a do-nothing virtual destructor.

- void `reset` ()
- void `set_scale` (int scale)
- void `set_guides` (int snap, int measure)
Sets the `m_snap` value and the `m_measure_length` members directly from the function parameters, which are in units of pulses (sometimes misleadingly called "ticks".)
- void `increment_size` ()
This function does nothing.

Private Member Functions

- void `enqueue_draw` ()
Wraps `queue_draw()` and forwards the call to the parent `perfed`, so that it can forward it to any other `perfed` that exists.
- void `set_zoom` (int z)
Implements the horizontal zoom feature.
- void `draw_background` ()
Separated out the drawing done in `on_expose_event()`, so that it can be redone when the zoom changes.
- void `draw_progress_on_window` ()
- void `change_horz` ()
Changes the `m_4bar_offset` and queues a draw operation.
- void `set_ppqn` (int ppqn)
Handles changes to the PPQN value in one place.
- long `tick_to_pixel` (midipulse tick)
Common calculation to convert a pulse/tick value to a perftime x value.
- midipulse `pixel_to_tick` (long pixel)
The inverse of `tick_to_pixel()`.
- int `tick_offset` ()
Centralizes calculation of the tick offset of the time bar.
- void `update_sizes` ()
This function does nothing.
- int `idle_progress` ()
This function just returns true.
- void `update_pixmap` ()
This function does nothing.
- void `draw_pixmap_on_window` ()
This function does nothing.
- void `on_realize` ()
Implements the on-realization event, then allocates some resources the could not be allocated in the constructor.
- bool `on_expose_event` (GdkEventExpose *ev)
Implements the on-expose event.
- bool `on_button_press_event` (GdkEventButton *ev)
Implement the button-press event to set the L and R ticks.
- void `on_size_allocate` (Gtk::Allocation &r)
Implements a size-allocation event.
- bool `on_button_release_event` (GdkEventButton *)
This button-release handler does nothing.
- bool `key_press_event` (GdkEventKey *ev)
This callback function handles a key-press event.

Private Attributes

- [perfedit](#) & [m_parent](#)
Provides a link to the perfedit that created this object.
- int [m_4bar_offset](#)
Not yet sure exactly what this member represents.
- int [m_tick_offset](#)
This member is m_4bar_offset times 16 times the current PPQN, to save some calculations and centralize this value.
- int [m_ppqn](#)
The current value of PPQN, which we are trying to get to work everywhere, when PPQN is changed from the global ppqn = 192.
- int [m_snap](#)
Snap value, starts out very small, equal to m_ppqn.
- int [m_measure_length](#)
Provides the length of a measure in pulses or ticks.
- int [m_left_marker_tick](#)
Holds the current location of the left (L) marker when arrow movement is in force.
- int [m_right_marker_tick](#)
Holds the current location of the right (R) marker when arrow movement is in force.
- int [m_perf_scale_x](#)
A class version of the global c_perf_scale_x factor.
- int [m_timearea_y](#)
A class version of the global c_timerarea_y factor.

Friends

- class [perfedit](#)

Additional Inherited Members

10.72.1 Constructor & Destructor Documentation

10.72.1.1 perftime()

```
seq64::perftime::perftime (
    perform & p,
    perfedit & parent,
    Gtk::Adjustment & hadjust,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

In the constructor you can only allocate colors; get_window() returns 0 because we have not been realized.

Note

Note that we still have to use a global constant in the base-class constructor; we cannot assign it to the corresponding member beforehand.

Parameters

<i>p</i>	Provides a reference to the main performance object of the application.
<i>parent</i>	Provides a reference to the object that contains this object, so that this object can tell the parent to queue up a drawing operation.
<i>hadjust</i>	Provides the horizontal scrollbar object needed so that perftime can respond to scrollbar cursor/thumb movement.
<i>ppqn</i>	An optional override of the default PPQN value for the application.

10.72.1.2 `~perftime()`

```
virtual seq64::perftime::~~perftime ( ) [inline], [virtual]
```

10.72.2 Member Function Documentation

10.72.2.1 `reset()`

```
void seq64::perftime::reset ( )
```

10.72.2.2 `set_scale()`

```
void seq64::perftime::set_scale (
    int scale )
```

10.72.2.3 `set_guides()`

```
void seq64::perftime::set_guides (
    int snap,
    int measure )
```

This function then fills in the background, and queues up a draw operation.

Parameters

<i>snap</i>	Provides the number of snap-pulses (pulses per snap interval) as calculated in perfedit::set_guides() . This is actually equal to the measure-pulses divided by the snap value in perfedit; the snap value defaults to 8.
<i>measure</i>	Provides the number of measure-pulses (pulses per measure) as calculated in perfedit::set_guides() .

10.72.2.4 increment_size()

```
void seq64::perftime::increment_size ( ) [inline]
```

Compare it to [perftroll::increment_size\(\)](#).

10.72.2.5 enqueue_draw()

```
void seq64::perftime::enqueue_draw ( ) [private]
```

The parent perfdit will call `perftime::queue_draw()` on behalf of this object, and it will pass a [perftime::enqueue_↔draw\(\)](#) to the peer perfdit's `perftime`, if the peer exists.

10.72.2.6 set_zoom()

```
void seq64::perftime::set_zoom (
    int z ) [private]
```

Redraws the background if the new zoom checked out.

Parameters

z	Provides the zoom value, which is checked, and then copied into <code>m_perf_scale_x</code> .
---	---

10.72.2.7 draw_background()

```
void seq64::perftime::draw_background ( ) [private]
```

Note that `m_measure_length == 0` will cause integer overflow.

10.72.2.8 draw_progress_on_window()

```
void seq64::perftime::draw_progress_on_window ( ) [private]
```

10.72.2.9 change_horz()

```
void seq64::perftime::change_horz ( ) [private]
```

Again, uses the constant, 16 [now offloaded to the new [tick_offset\(\)](#) function].

10.72.2.10 `set_ppqn()`

```
void seq64::perftime::set_ppqn (
    int ppqn ) [private]
```

It also modifies `m_snap`, `m_measure_length` (but always for four measures!), and `m_tick_offset`.

Todo We need make the 4 constant variable per the number of beats (quarter-notes) per bar, and also at least make 16 (4x4) a meaningful manifest constant.

Parameters

<i>ppqn</i>	The override value for the PPQN.
-------------	----------------------------------

10.72.2.11 `tick_to_pixel()`

```
long seq64::perftime::tick_to_pixel (
    midipulse tick ) [inline], [private]
```

Parameters

<i>tick</i>	The horizontal tick value to convert to an x pixel value, based on tick-offset and the x-scale.
-------------	---

Returns

Returns the x-pixel representing the time location parameter.

10.72.2.12 `pixel_to_tick()`

```
midipulse seq64::perftime::pixel_to_tick (
    long pixel ) [inline], [private]
```

Parameters

<i>pixel</i>	The pixel value.
--------------	------------------

Returns

Returns the time value represented b the pixel.

10.72.2.13 tick_offset()

```
int seq64::perftime::tick_offset ( ) [inline], [private]
```

Returns

Returns `m_4bar_offset * 16 * m_ppqn`.

10.72.2.14 update_sizes()

```
void seq64::perftime::update_sizes ( ) [inline], [private]
```

10.72.2.15 idle_progress()

```
int seq64::perftime::idle_progress ( ) [inline], [private]
```

10.72.2.16 update_pixmap()

```
void seq64::perftime::update_pixmap ( ) [inline], [private]
```

10.72.2.17 draw_pixmap_on_window()

```
void seq64::perftime::draw_pixmap_on_window ( ) [inline], [private]
```

10.72.2.18 on_realize()

```
void seq64::perftime::on_realize ( ) [private]
```

It is important to call the base-class version of this function.

The former work of this function is now done in base-class's [on_realize\(\)](#) and in its constructor now.

```
m_window = get_window();
m_gc = Gdk::GC::create(m_window);
m_window->clear();
set_size_request(10, m_timearea_y);
```

10.72.2.19 on_expose_event()

```
bool seq64::perftime::on_expose_event (
    GdkEventExpose * ev ) [private]
```

Redraws the background.

Note

The `perfed` object is created early on. When brought on-screen from `mainwnd` (the main window), first, [perftime::on_realize\(\)](#) is called, then this event is called.

Parameters

<i>ev</i>	The expose event, not used.
-----------	-----------------------------

Returns

Always returns true.

10.72.2.20 on_button_press_event()

```
bool seq64::perftime::on_button_press_event (
    GdkEventButton * p0 ) [private]
```

Added functionality to try to set the start-tick if ctrl-left-click is pressed.

Parameters

<i>p0</i>	The button event.
-----------	-------------------

Returns

Always returns true.

Why is setting the start-tick disabled? We re-enable it and see if it works. To our surprise, it works, but it sticks between stop/pause and the next playback in the performance editor. We added a feature where stop sets the start-tick to the left tick (or the beginning tick).

10.72.2.21 on_size_allocate()

```
void seq64::perftime::on_size_allocate (
    Gtk::Allocation & r ) [private]
```

10.72.2.22 on_button_release_event()

```
bool seq64::perftime::on_button_release_event (
    GdkEventButton * ) [inline], [private]
```

"ev", The button event parameter, is not used.

Returns

Always returns false

10.72.2.23 key_press_event()

```
bool seq64::perftime::key_press_event (
    GdkEventKey * ev ) [private]
```

Can't get the keystroke events to be seen by perffroll or perftime here using the normal callback function for keystrokes, and not sure why. The perfedit object can call this function, and that call works, so the perfedit class, which does get keystrokes, calls this function to do the work.

This function uses the "l" key to activate the movement of the "L" marker with the arrow keys, by the interval of on snap value for each press. It also uses the "r" key to activate the movement of the "R" marker, and the "x" to deactivate either movement move.

Be aware that there is no visual feedback, as yet, that one is in the movement mode.

Also be aware the changing the name of this function from "key_press_event()" to "on_key_press_event()" will disrupt the process, causing keystrokes to not get here. Too tricky.

10.72.3 Friends And Related Function Documentation

10.72.3.1 perfedit

```
friend class perfedit [friend]
```

10.72.4 Field Documentation

10.72.4.1 m_parent

```
perfedit& seq64::perftime::m_parent [private]
```

We want to support two perfedit windows, but the children of perfedit will have to communicate changes requiring a redraw through the parent.

10.72.4.2 m_4bar_offset

```
int seq64::perftime::m_4bar_offset [private]
```

Also, why always 4/16 in the calculations of this value? Might be able to get rid of this member, though it's a bit tricky.

10.72.4.3 m_tick_offset

```
int seq64::perftime::m_tick_offset [private]
```

Why 16?

10.72.4.4 m_ppqn

```
int seq64::perftime::m_ppqn [private]
```

10.72.4.5 m_snap

```
int seq64::perftime::m_snap [private]
```

10.72.4.6 m_measure_length

```
int seq64::perftime::m_measure_length [private]
```

This value is `m_ppqn * 4`, though eventually we want to employ a more flexible representation of measure length. Supports perftime's keystroke processing.

10.72.4.7 m_left_marker_tick

```
int seq64::perftime::m_left_marker_tick [private]
```

Otherwise it is -1. Supports perftime's keystroke processing.

10.72.4.8 m_right_marker_tick

```
int seq64::perftime::m_right_marker_tick [private]
```

Otherwise it is -1. Supports perftime's keystroke processing.

10.72.4.9 m_perf_scale_x

```
int seq64::perftime::m_perf_scale_x [private]
```

10.72.4.10 m_timearea_y

```
int seq64::perftime::m_timearea_y [private]
```

10.73 seq64::midi_port_info::port_info_t Struct Reference

Hold the information for a single port.

Data Fields

- int [m_client_number](#)
The major buss number of the port.
- std::string [m_client_name](#)
The system's name for the client.
- int [m_port_number](#)
The minor port number of the port.
- std::string [m_port_name](#)
The system's name for the port.
- int [m_queue_number](#)
A number used in some APIs.
- bool [m_is_input](#)
Indicates an input port.
- bool [m_is_virtual](#)
Indicates an manual/virtual port.
- bool [m_is_system](#)
Built-in port, almost always false.

10.73.1 Detailed Description

Except for the virtual-vs-normal status, this information is obtained by scanning the system at the startup time of the application.

10.73.2 Field Documentation

10.73.2.1 m_client_number

```
int seq64::midi_port_info::port_info_t::m_client_number
```

10.73.2.2 m_client_name

```
std::string seq64::midi_port_info::port_info_t::m_client_name
```

10.73.2.3 m_port_number

```
int seq64::midi_port_info::port_info_t::m_port_number
```

10.73.2.4 m_port_name

```
std::string seq64::midi_port_info::port_info_t::m_port_name
```

10.73.2.5 m_queue_number

```
int seq64::midi_port_info::port_info_t::m_queue_number
```

10.73.2.6 m_is_input

```
bool seq64::midi_port_info::port_info_t::m_is_input
```

10.73.2.7 m_is_virtual

```
bool seq64::midi_port_info::port_info_t::m_is_virtual
```

10.73.2.8 m_is_system

```
bool seq64::midi_port_info::port_info_t::m_is_system
```

10.74 seq64::rc_settings Class Reference

This class contains the options formerly named "global_xxxxxx".

Public Member Functions

- [rc_settings](#) ()
Default constructor.
- [rc_settings](#) (const [rc_settings](#) &rhs)
Copy constructor.
- [rc_settings](#) & [operator=](#) (const [rc_settings](#) &rhs)
Principal assignment operator.
- `std::string config_filespec () const`
Constructs the full path and file specification for the "rc" file based on whether or not the legacy Seq24 filenames are being used.
- `std::string user_filespec () const`
Constructs the full path and file specification for the "user" file based on whether or not the legacy Seq24 filenames are being used.
- `void set_defaults ()`
Sets the default values.
- `const std::string & comments_block () const`
'Getter' function for member m_comments_block
- `void clear_comments ()`
'Setter' function for member m_comments_block
- `void append_comment_line (const std::string &line)`
'Setter' function for member m_comments_block
- `bool verbose_option () const`
'Getter' function for member m_verbose_option
- `bool auto_option_save () const`
'Getter' function for member m_auto_option_save
- `bool legacy_format () const`
'Getter' function for member m_legacy_format
- `bool lash_support () const`
'Getter' function for member m_lash_support
- `bool allow_mod4_mode () const`
'Getter' function for member m_allow_mod4_mode
- `bool allow_snap_split () const`
'Getter' function for member m_allow_snap_split
- `bool allow_click_edit () const`
'Getter' function for member m_allow_click_edit
- `bool show_midi () const`
'Getter' function for member m_show_midi
- `bool priority () const`
'Getter' function for member m_priority
- `bool stats () const`
'Getter' function for member m_stats
- `bool pass_sysex () const`
'Getter' function for member m_pass_sysex
- `bool with_jack_transport () const`
'Getter' function for member m_with_jack_transport
- `bool with_jack_master () const`
'Getter' function for member m_with_jack_master
- `bool with_jack_master_cond () const`
'Getter' function for member m_with_jack_master_cond
- `bool with_jack_midi () const`

- *'Getter' function for member m_with_jack_midi*
- void [with_jack_transport](#) (bool flag)
- *'Setter' function for member m_with_jack_transport*
- void [with_jack_master](#) (bool flag)
- *'Setter' function for member m_with_jack_master*
- void [with_jack_master_cond](#) (bool flag)
- *'Setter' function for member m_with_jack_master_cond*
- bool [with_jack](#) () const
- *'Getter' function for member m_with_jack_transport m_with_jack_master, and m_with_jack_master_cond, to save client code some trouble.*
- bool [filter_by_channel](#) () const
- *'Getter' function for member m_filter_by_channel*
- bool [manual_alsa_ports](#) () const
- *'Getter' function for member m_manual_alsa_ports*
- bool [reveal_alsa_ports](#) () const
- *'Getter' function for member m_reveal_alsa_ports*
- bool [print_keys](#) () const
- *'Getter' function for member m_print_keys*
- bool [device_ignore](#) () const
- *'Getter' function for member m_device_ignore*
- int [device_ignore_num](#) () const
- *'Getter' function for member m_device_ignore_num*
- [interaction_method_t](#) [interaction_method](#) () const
- *'Getter' function for member m_interaction_method*
- [mute_group_handling_t](#) [mute_group_saving](#) () const
- *'Getter' function for member m_mute_group_saving*
- const std::string & [filename](#) () const
- *'Getter' function for member m_filename*
- void [filename](#) (const std::string &value)
- *'Setter' function for member m_filename*
- const std::string & [jack_session_uuid](#) () const
- *'Getter' function for member m_jack_session_uuid*
- const std::string & [last_used_dir](#) () const
- *'Getter' function for member m_last_used_dir*
- void [last_used_dir](#) (const std::string &value)
- *'Setter' function for member m_last_used_dir*
- bool [add_recent_file](#) (const std::string &[filename](#))
- *'Setter' function for member m_recent_files*
- bool [append_recent_file](#) (const std::string &[filename](#))
- bool [remove_recent_file](#) (const std::string &[filename](#))
- const std::string & [config_directory](#) () const
- *'Getter' function for member m_config_directory*
- std::string [home_config_directory](#) () const
- *Provides the directory for the configuration file, and also creates the directory if necessary.*
- void [set_config_files](#) (const std::string &value)
- *'Setter' function for member m_config_filename and m_user_filename*
- const std::string & [config_filename](#) () const
- *'Getter' function for member m_config_filename*
- const std::string & [user_filename](#) () const
- *'Getter' function for member m_user_filename*
- const std::string & [config_filename_alt](#) () const

- 'Getter' function for member m_config_filename_alt;*
- const std::string & [user_filename_alt](#) () const
- 'Getter' function for member m_user_filename_alt*
- const std::string [application_name](#) () const
- 'Getter' function for member m_application_name*
- const std::string & [app_client_name](#) () const
- 'Getter' function for member m_app_client_name*
- int [tempo_track_number](#) () const
- 'Getter' function for member m_tempo_track_number*
- std::string [recent_file](#) (int index, bool shorten=true) const
- 'Getter' function for member m_recent_files*
- int [recent_file_count](#) () const
- 'Getter' function for member m_recent_files.size()*

Protected Member Functions

- void [verbose_option](#) (bool flag)
- 'Setter' function for member m_verbose_option*
- void [auto_option_save](#) (bool flag)
- 'Setter' function for member m_auto_option_save*
- void [legacy_format](#) (bool flag)
- 'Setter' function for member m_legacy_format*
- void [lash_support](#) (bool flag)
- 'Setter' function for member m_lash_support*
- void [allow_mod4_mode](#) (bool flag)
- 'Setter' function for member m_allow_mod4_mode*
- void [allow_snap_split](#) (bool flag)
- 'Setter' function for member m_allow_snap_split*
- void [allow_click_edit](#) (bool flag)
- 'Setter' function for member m_allow_click_edit*
- void [show_midi](#) (bool flag)
- 'Setter' function for member m_show_midi*
- void [priority](#) (bool flag)
- 'Setter' function for member m_priority*
- void [stats](#) (bool flag)
- 'Setter' function for member m_stats*
- void [pass_sysex](#) (bool flag)
- 'Setter' function for member m_pass_sysex*
- void [with_jack_midi](#) (bool flag)
- 'Setter' function for member m_with_jack_midi*
- void [filter_by_channel](#) (bool flag)
- 'Setter' function for member m_filter_by_channel*
- void [manual_alsa_ports](#) (bool flag)
- 'Setter' function for member m_manual_alsa_ports*
- void [reveal_alsa_ports](#) (bool flag)
- 'Setter' function for member m_reveal_alsa_ports*
- void [print_keys](#) (bool flag)
- 'Setter' function for member m_print_keys*
- void [device_ignore](#) (bool flag)
- 'Setter' function for member m_device_ignore*

- void [tempo_track_number](#) (int track)
'Setter' function for member m_tempo_track_number
- void [device_ignore_num](#) (int value)
'Setter' function for member m_device_ignore_num However, please note that this value, while set in the options processing of the main module, does not appear to be used anywhere in the code in seq24, Sequencer24, and this application.
- bool [interaction_method](#) ([interaction_method_t](#) value)
'Setter' function for member m_interaction_method
- bool [mute_group_saving](#) ([mute_group_handling_t](#) mgh)
'Setter' function for member m_mute_group_saving
- void [jack_session_uuid](#) (const std::string &value)
'Setter' function for member m_jack_session_uuid
- void [config_directory](#) (const std::string &value)
'Setter' function for member m_config_directory
- void [config_filename](#) (const std::string &value)
'Setter' function for member m_config_filename ("rc")
- void [user_filename](#) (const std::string &value)
'Setter' function for member m_user_filename ("usr")
- void [config_filename_alt](#) (const std::string &value)
'Setter' function for member m_config_filename_alt
- void [user_filename_alt](#) (const std::string &value)
'Setter' function for member m_user_filename_alt

Private Attributes

- std::string [m_comments_block](#)
[comments]
- bool [m_verbose_option](#)
[auto-option-save] setting.
- bool [m_auto_option_save](#)
[auto-option-save] setting.
- bool [m_legacy_format](#)
Write files in legacy format.
- bool [m_lash_support](#)
Enable LASH, if compiled in.
- bool [m_allow_mod4_mode](#)
Allow Mod4 to hold drawing mode.
- bool [m_allow_snap_split](#)
Allow snap-split of a trigger.
- bool [m_allow_click_edit](#)
Allow double-click edit pattern.
- bool [m_show_midi](#)
Show MIDI events to console.
- bool [m_priority](#)
Run at high priority (Linux only).
- bool [m_stats](#)
Show some output statistics.
- bool [m_pass_sysex](#)
Pass SysEx to outputs, not ready.
- bool [m_with_jack_transport](#)

- Enable synchrony with JACK.*

 - bool [m_with_jack_master](#)

Serve as a JACK transport Master.
 - bool [m_with_jack_master_cond](#)

Serve as JACK Master if possible.
 - bool [m_with_jack_midi](#)

Use JACK MIDI.
 - bool [m_filter_by_channel](#)

Record only sequence channel data.
 - bool [m_manual_alsa_ports](#)

[manual-alsa-ports] setting.
 - bool [m_reveal_alsa_ports](#)

[reveal-alsa-ports] setting.
 - bool [m_print_keys](#)

Show hot-key in main window slot.
 - bool [m_device_ignore](#)

From seq24 module, unused!
 - int [m_device_ignore_num](#)

From seq24 module, unused!
 - [interaction_method_t m_interaction_method](#)

[interaction-method]
 - [mute_group_handling_t m_mute_group_saving](#)

Handling of mutes.
 - std::string [m_filename](#)

Provides the name of current MIDI file.
 - std::string [m_jack_session_uuid](#)

Holds the JACK UUID value that makes this JACK connection unique.
 - std::string [m_last_used_dir](#)

Holds the directory from which the last MIDI file was opened (or saved).
 - std::string [m_config_directory](#)

Holds the current "rc" and "user" configuration directory.
 - std::string [m_config_filename](#)

Holds the current "rc" configuration filename.
 - std::string [m_user_filename](#)

Holds the current "user" configuration filename.
 - std::string [m_config_filename_alt](#)

Holds the legacy "rc" filename, ".seq24rc".
 - std::string [m_user_filename_alt](#)

Holds the legacy "user" filename, ".seq24usr".
 - const std::string [m_application_name](#)

Holds the application name, e.g.
 - std::string [m_app_client_name](#)

Holds the client name for the application.
 - int [m_tempo_track_number](#)

New value to allow the user to violate the MIDI specification and use a track other than the first track (#0) as the MIDI tempo track.
 - recent [m_recent_files](#)

Holds a few MIDI file-names most recently used.

Friends

- class [optionsfile](#)
- class [options](#)
- class [mainwnd](#)
- class [rtmidi_info](#)
- int [parse_command_line_options](#) ([perform](#) &, int, char *[])
Parses the command-line options on behalf of the application.
- bool [help_check](#) (int, char *[])
Checks to see if the first option is a help or version argument, just so we can skip the "Reading configuration ..." messages.

10.74.1 Detailed Description

It gives us a whole lot more encapsulation and control over how the options of the "rc" file ([optionsfile](#)) are set and used. Note that this class does not support the hot-keys options; those are handled in the [keys_perform](#) class.

10.74.2 Constructor & Destructor Documentation

10.74.2.1 [rc_settings\(\)](#) [1/2]

```
seq64::rc_settings::rc_settings ( )
```

10.74.2.2 [rc_settings\(\)](#) [2/2]

```
seq64::rc_settings::rc_settings (
    const rc\_settings & rhs )
```

Parameters

<i>rhs</i>	The source of the data for the copy.
------------	--------------------------------------

10.74.3 Member Function Documentation

10.74.3.1 [operator=\(\)](#)

```
rc\_settings & seq64::rc_settings::operator= (
    const rc\_settings & rhs )
```


Parameters

<i>rhs</i>	The source of the data for the assignment.
------------	--

Returns

Returns a reference to the destination for use in serial assignments.

10.74.3.2 config_filespec()

```
std::string seq64::rc_settings::config_filespec ( ) const
```

Returns

If [home_config_directory\(\)](#) returns a non-empty string, then the legacy or normal "rc" configuration file-name is appended to that result, and returned. Otherwise, an empty string is returned.

10.74.3.3 user_filespec()

```
std::string seq64::rc_settings::user_filespec ( ) const
```

Returns

If [home_config_directory\(\)](#) returns a non-empty string, then the legacy or normal "user" configuration file-name is appended to that result, and returned. Otherwise, an empty string is returned.

10.74.3.4 set_defaults()

```
void seq64::rc_settings::set_defaults ( )
```

10.74.3.5 comments_block()

```
const std::string& seq64::rc_settings::comments_block ( ) const [inline]
```

10.74.3.6 clear_comments()

```
void seq64::rc_settings::clear_comments ( ) [inline]
```

10.74.3.7 append_comment_line()

```
void seq64::rc_settings::append_comment_line (
    const std::string & line ) [inline]
```

10.74.3.8 verbose_option() [1/2]

```
bool seq64::rc_settings::verbose_option ( ) const [inline]
```

10.74.3.9 auto_option_save() [1/2]

```
bool seq64::rc_settings::auto_option_save ( ) const [inline]
```

10.74.3.10 legacy_format() [1/2]

```
bool seq64::rc_settings::legacy_format ( ) const [inline]
```

10.74.3.11 lash_support() [1/2]

```
bool seq64::rc_settings::lash_support ( ) const [inline]
```

10.74.3.12 allow_mod4_mode() [1/2]

```
bool seq64::rc_settings::allow_mod4_mode ( ) const [inline]
```

10.74.3.13 allow_snap_split() [1/2]

```
bool seq64::rc_settings::allow_snap_split ( ) const [inline]
```

10.74.3.14 allow_click_edit() [1/2]

```
bool seq64::rc_settings::allow_click_edit ( ) const [inline]
```

10.74.3.15 show_midi() [1/2]

```
bool seq64::rc_settings::show_midi ( ) const [inline]
```

10.74.3.16 priority() [1/2]

```
bool seq64::rc_settings::priority ( ) const [inline]
```

10.74.3.17 stats() [1/2]

```
bool seq64::rc_settings::stats ( ) const [inline]
```

10.74.3.18 pass_sysex() [1/2]

```
bool seq64::rc_settings::pass_sysex ( ) const [inline]
```

10.74.3.19 with_jack_transport() [1/2]

```
bool seq64::rc_settings::with_jack_transport ( ) const [inline]
```

10.74.3.20 with_jack_master() [1/2]

```
bool seq64::rc_settings::with_jack_master ( ) const [inline]
```

10.74.3.21 with_jack_master_cond() [1/2]

```
bool seq64::rc_settings::with_jack_master_cond ( ) const [inline]
```

10.74.3.22 with_jack_midi() [1/2]

```
bool seq64::rc_settings::with_jack_midi ( ) const [inline]
```

10.74.3.23 with_jack_transport() [2/2]

```
void seq64::rc_settings::with_jack_transport (
    bool flag )
```

10.74.3.24 with_jack_master() [2/2]

```
void seq64::rc_settings::with_jack_master (
    bool flag )
```

10.74.3.25 with_jack_master_cond() [2/2]

```
void seq64::rc_settings::with_jack_master_cond (
    bool flag )
```

10.74.3.26 with_jack()

```
bool seq64::rc_settings::with_jack ( ) const [inline]
```

Do not confuse these original options with the new "no JACK MIDI" option.

10.74.3.27 filter_by_channel() [1/2]

```
bool seq64::rc_settings::filter_by_channel ( ) const [inline]
```

10.74.3.28 manual_alsa_ports() [1/2]

```
bool seq64::rc_settings::manual_alsa_ports ( ) const [inline]
```

10.74.3.29 reveal_alsa_ports() [1/2]

```
bool seq64::rc_settings::reveal_alsa_ports ( ) const [inline]
```

10.74.3.30 print_keys() [1/2]

```
bool seq64::rc_settings::print_keys ( ) const [inline]
```

10.74.3.31 device_ignore() [1/2]

```
bool seq64::rc_settings::device_ignore ( ) const [inline]
```

10.74.3.32 device_ignore_num() [1/2]

```
int seq64::rc_settings::device_ignore_num ( ) const [inline]
```

10.74.3.33 interaction_method() [1/2]

```
interaction_method_t seq64::rc_settings::interaction_method ( ) const [inline]
```

10.74.3.34 mute_group_saving() [1/2]

```
mute_group_handling_t seq64::rc_settings::mute_group_saving ( ) const [inline]
```

10.74.3.35 filename() [1/2]

```
const std::string& seq64::rc_settings::filename ( ) const [inline]
```

10.74.3.36 filename() [2/2]

```
void seq64::rc_settings::filename (
    const std::string & value )
```

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

10.74.3.37 jack_session_uuid() [1/2]

```
const std::string& seq64::rc_settings::jack_session_uuid ( ) const [inline]
```

10.74.3.38 last_used_dir() [1/2]

```
const std::string& seq64::rc_settings::last_used_dir ( ) const [inline]
```

10.74.3.39 last_used_dir() [2/2]

```
void seq64::rc_settings::last_used_dir (
    const std::string & value )
```

Parameters

<i>value</i>	The value to use to make the setting. It needs to be a directory, not a file. Also, we now expand a relative directory to the full path to that directory, to avoid ambiguity should the application be run from a different directory.
--------------	---

10.74.3.40 add_recent_file()

```
bool seq64::rc_settings::add_recent_file (
    const std::string & filename ) [inline]
```

First makes sure the filename is not already present, and removes the back entry from the list, if it is full (SEQ64↵_RECENT_FILES_MAX) before adding it.

Now the full pathname is added.

Parameters

<i>fname</i>	Provides the full path to the MIDI file that is to be added to the recent-files list.
--------------	---

Returns

Returns true if the file-name was able to be added.

10.74.3.41 append_recent_file()

```
bool seq64::rc_settings::append_recent_file (
    const std::string & filename ) [inline]
```

10.74.3.42 remove_recent_file()

```
bool seq64::rc_settings::remove_recent_file (
    const std::string & filename ) [inline]
```

10.74.3.43 config_directory() [1/2]

```
const std::string& seq64::rc_settings::config_directory ( ) const [inline]
```

10.74.3.44 home_config_directory()

```
std::string seq64::rc_settings::home_config_directory ( ) const
```

If the legacy format is in force, then the home directory for the configuration is (in Linux) `"/home/username"`, and the configuration file is `".seq24rc"`.

If the new format is in force, then the home directory is (in Linux) `"/home/username/.config/sequencer64"`, and the configuration file is `"sequencer64.rc"`.

This function should also adapt to Windows conventions automatically. We shall see. No, it does not. But all we have to do is replace Window's `HOMEPATH` with its `LOCALAPPDATA` value.

`getenv(HOME)`:

- Linux returns `"/home/ahlstrom"`. Append `"/.config/sequencer64"`.
- Windows returns `"\Users\ahlstrom"`. A better value than `HOMEPATH` is `LOCALAPPDATA`, which gives us most of what we want: `"C:\Users\ahlstrom\AppData\local"`, and then we append simply `"sequencer64"`.

Returns

Returns the selected home configuration directory. If it does not exist, or could not be created, then an empty string is returned.

10.74.3.45 set_config_files()

```
void seq64::rc_settings::set_config_files (
    const std::string & value )
```

Implements the `--config` option to change both configuration files ("rc" and "usr") with one option.

Parameters

<i>value</i>	The value to use to make the setting, if the string is not empty. If the value has an extension, it is stripped first.
--------------	--

10.74.3.46 config_filename() [1/2]

```
const std::string& seq64::rc_settings::config_filename ( ) const [inline]
```

10.74.3.47 user_filename() [1/2]

```
const std::string& seq64::rc_settings::user_filename ( ) const [inline]
```

10.74.3.48 config_filename_alt() [1/2]

```
const std::string& seq64::rc_settings::config_filename_alt ( ) const [inline]
```

10.74.3.49 user_filename_alt() [1/2]

```
const std::string& seq64::rc_settings::user_filename_alt ( ) const [inline]
```

10.74.3.50 application_name()

```
const std::string seq64::rc_settings::application_name ( ) const [inline]
```

10.74.3.51 app_client_name()

```
const std::string& seq64::rc_settings::app_client_name ( ) const [inline]
```

10.74.3.52 tempo_track_number() [1/2]

```
int seq64::rc_settings::tempo_track_number ( ) const [inline]
```

10.74.3.53 recent_file()

```
std::string seq64::rc_settings::recent_file (
    int index,
    bool shorten = true ) const
```

Gets the desired recent MIDI file-name, if present.

Parameters

<i>index</i>	Provides the desired index into the recent-files vector.
<i>shorten</i>	If true, remove the path-name from the file-name. True by default. It needs to be short for the menu entry, but the full path-name for the "rc" file.

Returns

Returns `m_recent_files[index]`, perhaps shortened. An empty string is returned if there is no such animal.

10.74.3.54 recent_file_count()

```
int seq64::rc_settings::recent_file_count ( ) const [inline]
```

10.74.3.55 verbose_option() [2/2]

```
void seq64::rc_settings::verbose_option (
    bool flag ) [inline], [protected]
```

10.74.3.56 auto_option_save() [2/2]

```
void seq64::rc_settings::auto_option_save (
    bool flag ) [inline], [protected]
```

10.74.3.57 legacy_format() [2/2]

```
void seq64::rc_settings::legacy_format (
    bool flag ) [inline], [protected]
```

10.74.3.58 lash_support() [2/2]

```
void seq64::rc_settings::lash_support (
    bool flag ) [inline], [protected]
```

10.74.3.59 allow_mod4_mode() [2/2]

```
void seq64::rc_settings::allow_mod4_mode (
    bool flag ) [inline], [protected]
```

10.74.3.60 allow_snap_split() [2/2]

```
void seq64::rc_settings::allow_snap_split (
    bool flag ) [inline], [protected]
```

10.74.3.61 allow_click_edit() [2/2]

```
void seq64::rc_settings::allow_click_edit (
    bool flag ) [inline], [protected]
```

10.74.3.62 show_midi() [2/2]

```
void seq64::rc_settings::show_midi (
    bool flag ) [inline], [protected]
```

10.74.3.63 priority() [2/2]

```
void seq64::rc_settings::priority (
    bool flag ) [inline], [protected]
```

10.74.3.64 stats() [2/2]

```
void seq64::rc_settings::stats (
    bool flag ) [inline], [protected]
```

10.74.3.65 pass_sysex() [2/2]

```
void seq64::rc_settings::pass_sysex (
    bool flag ) [inline], [protected]
```

10.74.3.66 with_jack_midi() [2/2]

```
void seq64::rc_settings::with_jack_midi (
    bool flag )    [inline], [protected]
```

10.74.3.67 filter_by_channel() [2/2]

```
void seq64::rc_settings::filter_by_channel (
    bool flag )    [inline], [protected]
```

10.74.3.68 manual_alsa_ports() [2/2]

```
void seq64::rc_settings::manual_alsa_ports (
    bool flag )    [inline], [protected]
```

10.74.3.69 reveal_alsa_ports() [2/2]

```
void seq64::rc_settings::reveal_alsa_ports (
    bool flag )    [inline], [protected]
```

10.74.3.70 print_keys() [2/2]

```
void seq64::rc_settings::print_keys (
    bool flag )    [inline], [protected]
```

10.74.3.71 device_ignore() [2/2]

```
void seq64::rc_settings::device_ignore (
    bool flag )    [inline], [protected]
```

10.74.3.72 tempo_track_number() [2/2]

```
void seq64::rc_settings::tempo_track_number (
    int track )    [protected]
```

10.74.3.73 device_ignore_num() [2/2]

```
void seq64::rc_settings::device_ignore_num (
    int value )    [protected]
```

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

10.74.3.74 `interaction_method()` [2/2]

```
bool seq64::rc_settings::interaction_method (  
    interaction_method_t value ) [protected]
```

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

Returns

Returns true if the value was legal.

10.74.3.75 `mute_group_saving()` [2/2]

```
bool seq64::rc_settings::mute_group_saving (  
    mute_group_handling_t mgh ) [protected]
```

10.74.3.76 `jack_session_uuid()` [2/2]

```
void seq64::rc_settings::jack_session_uuid (  
    const std::string & value ) [protected]
```

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

10.74.3.77 `config_directory()` [2/2]

```
void seq64::rc_settings::config_directory (  
    const std::string & value ) [protected]
```

Parameters

<i>value</i>	The value to use to make the setting. Currently, we do not handle relative paths. To do so seems... iffy.
--------------	---

10.74.3.78 config_filename() [2/2]

```
void seq64::rc_settings::config_filename (
    const std::string & value ) [protected]
```

Parameters

<i>value</i>	The value to use to make the setting, if the string is not empty. If there is no period in the string, then ".rc" is appended to the end of the filename.
--------------	---

10.74.3.79 user_filename() [2/2]

```
void seq64::rc_settings::user_filename (
    const std::string & value ) [protected]
```

Parameters

<i>value</i>	The value to use to make the setting, if the string is not empty. If there is no period in the string, then ".usr" is appended to the end of the filename.
--------------	--

10.74.3.80 config_filename_alt() [2/2]

```
void seq64::rc_settings::config_filename_alt (
    const std::string & value ) [protected]
```

Parameters

<i>value</i>	The value to use to make the setting, if the string is not empty.
--------------	---

10.74.3.81 user_filename_alt() [2/2]

```
void seq64::rc_settings::user_filename_alt (
    const std::string & value ) [protected]
```

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

10.74.4 Friends And Related Function Documentation

10.74.4.1 optionsfile

```
friend class optionsfile [friend]
```

10.74.4.2 options

```
friend class options [friend]
```

10.74.4.3 mainwnd

```
friend class mainwnd [friend]
```

10.74.4.4 rtmidi_info

```
friend class rtmidi_info [friend]
```

10.74.4.5 parse_command_line_options

```
int parse_command_line_options (
    perform & p,
    int argc,
    char * argv[] ) [friend]
```

Note that, since we call this function twice (once before the configuration files are parsed, and once after), we have to make sure that the global value `optind` is reset to 0 before calling this function. Note that the traditional reset value for `optind` is 1, but 0 is used in GNU code to trigger the internal initialization routine of `get_opt()`.

Parameters

<i>p</i>	The performance object that implements some of the command-line options.
<i>argc</i>	The number of command-line arguments.
<i>argv</i>	The array of command-line argument pointers.

Returns

Returns the value of `optind` if no help-related options were provided.

10.74.4.6 help_check

```
bool help_check (
    int argc,
    char * argv[] ) [friend]
```

Also check for the `--legacy` option. Finally, it also checks for the `"?"` option that people sometimes use as a guess to get help.

Parameters

<i>argc</i>	The number of command-line arguments.
<i>argv</i>	The array of command-line argument pointers.

Returns

Returns true only if `-v`, `-V`, `--version`, `-#`, `-h`, `--help`, or `"?"` were encountered. If the legacy options occurred, then `rc().legacy_format(true)` is called, as a side effect, because it will be needed before we parse the options.

10.74.5 Field Documentation

10.74.5.1 m_comments_block

```
std::string seq64::rc_settings::m_comments_block [private]
```

Provides a way to embed comments in the "rc" file and not lose them when the "rc" file is auto-saved.

10.74.5.2 m_verbose_option

```
bool seq64::rc_settings::m_verbose_option [private]
```

10.74.5.3 m_auto_option_save

```
bool seq64::rc_settings::m_auto_option_save [private]
```

10.74.5.4 m_legacy_format

```
bool seq64::rc_settings::m_legacy_format [private]
```

10.74.5.5 m_lash_support

```
bool seq64::rc_settings::m_lash_support [private]
```

10.74.5.6 m_allow_mod4_mode

```
bool seq64::rc_settings::m_allow_mod4_mode [private]
```

10.74.5.7 m_allow_snap_split

```
bool seq64::rc_settings::m_allow_snap_split [private]
```

10.74.5.8 m_allow_click_edit

```
bool seq64::rc_settings::m_allow_click_edit [private]
```

10.74.5.9 m_show_midi

```
bool seq64::rc_settings::m_show_midi [private]
```

10.74.5.10 m_priority

```
bool seq64::rc_settings::m_priority [private]
```


10.74.5.11 m_stats

```
bool seq64::rc_settings::m_stats [private]
```

10.74.5.12 m_pass_sysex

```
bool seq64::rc_settings::m_pass_sysex [private]
```

10.74.5.13 m_with_jack_transport

```
bool seq64::rc_settings::m_with_jack_transport [private]
```

10.74.5.14 m_with_jack_master

```
bool seq64::rc_settings::m_with_jack_master [private]
```

10.74.5.15 m_with_jack_master_cond

```
bool seq64::rc_settings::m_with_jack_master_cond [private]
```

10.74.5.16 m_with_jack_midi

```
bool seq64::rc_settings::m_with_jack_midi [private]
```

10.74.5.17 m_filter_by_channel

```
bool seq64::rc_settings::m_filter_by_channel [private]
```

10.74.5.18 m_manual_alsa_ports

```
bool seq64::rc_settings::m_manual_alsa_ports [private]
```

10.74.5.19 m_reveal_alsa_ports

```
bool seq64::rc_settings::m_reveal_alsa_ports [private]
```

10.74.5.20 m_print_keys

```
bool seq64::rc_settings::m_print_keys [private]
```

10.74.5.21 m_device_ignore

```
bool seq64::rc_settings::m_device_ignore [private]
```

10.74.5.22 m_device_ignore_num

```
int seq64::rc_settings::m_device_ignore_num [private]
```

10.74.5.23 m_interaction_method

```
interaction_method_t seq64::rc_settings::m_interaction_method [private]
```

10.74.5.24 m_mute_group_saving

```
mute_group_handling_t seq64::rc_settings::m_mute_group_saving [private]
```

10.74.5.25 m_filename

```
std::string seq64::rc_settings::m_filename [private]
```

10.74.5.26 m_jack_session_uuid

```
std::string seq64::rc_settings::m_jack_session_uuid [private]
```

10.74.5.27 m_last_used_dir

```
std::string seq64::rc_settings::m_last_used_dir [private]
```

10.74.5.28 m_config_directory

```
std::string seq64::rc_settings::m_config_directory [private]
```

This value is "~/config/sequencer64" by default.

10.74.5.29 m_config_filename

```
std::string seq64::rc_settings::m_config_filename [private]
```

This value is "sequencer64.rc" by default.

10.74.5.30 m_user_filename

```
std::string seq64::rc_settings::m_user_filename [private]
```

This value is "sequencer64.rc" by default.

10.74.5.31 m_config_filename_alt

```
std::string seq64::rc_settings::m_config_filename_alt [private]
```

10.74.5.32 m_user_filename_alt

```
std::string seq64::rc_settings::m_user_filename_alt [private]
```

10.74.5.33 m_application_name

```
const std::string seq64::rc_settings::m_application_name [private]
```

"sequencer64", "seq64portmidi", or "seq64". This is a constant, set to SEQ64_APP_NAME. Also see the [seq_↩ app_name\(\)](#) function.

10.74.5.34 m_app_client_name

```
std::string seq64::rc_settings::m_app_client_name [private]
```

This is much like the application name, but in the future will be a configuration option. For now it is just the value of the SEQ64_CLIENT_NAME macro. Also see the [seq_client_name\(\)](#) function.

10.74.5.35 m_tempo_track_number

```
int seq64::rc_settings::m_tempo_track_number [private]
```

10.74.5.36 m_recent_files

```
recent seq64::rc_settings::m_recent_files [private]
```

Although this is a vector, we do not let it grow past SEQ64_RECENT_FILES_MAX.

New feature from Oli Kester's kepler34 project.

```
std::vector<std::string> m_recent_files;
```

10.75 seq64::rect Class Reference

Supports a simple rectangle and some common manipulations needed by the user-interface.

Public Member Functions

- [rect](#) ()
Default constructor
- [rect](#) (int *x*, int *y*, int *width*, int *height*)
Principal constructor.
- void [get](#) (int &*x*, int &*y*, int &*width*, int &*height*) const
Gets the rectangle values for primitive callers that don't store them as a rectangle.
- void [set](#) (int *x*, int *y*, int *width*, int *height*)
Sets all of the members of the rectangle directly.
- void [clear](#) ()
- void [xy_to_rect](#) (int *x1*, int *y1*, int *x2*, int *y2*)
- int [x](#) () const
'Getter' function for member m_x
- void [x](#) (int *v*)
'Setter' function for member m_x The width is assumed to be unchanged by this function.
- void [x_incr](#) (int *v*)
'Setter' function for member m_x Provides a setter that uses the parameter to increment the member.
- int [y](#) () const
'Getter' function for member m_y
- void [y](#) (int *v*)

- *'Setter' function for member m_y The height is assumed to be unchanged by this function.*
- void `y_incr` (int v)
 - *'Setter' function for member m_y Provides a setter that uses the parameter to increment the member.*
- int `width` () const
 - *'Getter' function for member m_width*
- void `width` (int w)
 - *'Setter' function for member m_width*
- void `width_incr` (int w)
 - *'Setter' function for member m_width*
- int `height` () const
 - *'Getter' function for member m_height*
- void `height` (int h)
 - *'Setter' function for member m_height*
- void `height_incr` (int h)
 - *'Setter' function for member m_height*
- void `xy_incr` (int xv, int yv)

Static Public Member Functions

- static void `xy_to_rect` (int x1, int y1, int x2, int y2, `rect` &r)
 - *Converts rectangle corner coordinates to a rect object, which includes width and height.*
- static void `xy_to_rect_get` (int x1, int y1, int x2, int y2, int &x, int &y, int &w, int &h)
 - *Converts rectangle corner coordinates to a starting coordinate, plus a width and height.*

Static Private Member Functions

- static int `calculated_width` (int x1, int x2)
 - *The calculated width is always positive.*
- static int `calculated_height` (int y1, int y2)
 - *The calculated height is always positive.*

Private Attributes

- int `m_x`
 - *The x coordinate of the first corner.*
- int `m_y`
 - *The y coordinate of the first corner.*
- int `m_width`
 - *The width of the rectangle.*
- int `m_height`
 - *The height of the rectangle.*

10.75.1 Detailed Description

Will eventually replace the `gui_drawingarea_gtk2::rect` structure.

One minor issue that may crop up in the transition from Gtkmm to Qt 5 is the exact meaning of the coordinates. To be clarified later. For now, it uses the current Gtkmm conventions.

10.75.2 Constructor & Destructor Documentation

10.75.2.1 `rect()` [1/2]

```
seq64::rect::rect ( )
```

10.75.2.2 `rect()` [2/2]

```
seq64::rect::rect (
    int x,
    int y,
    int width,
    int height )
```

Parameters

<i>x</i>	The x coordinate.
<i>y</i>	The y coordinate.
<i>width</i>	The width value.
<i>height</i>	The width value.

10.75.3 Member Function Documentation

10.75.3.1 `get()`

```
void seq64::rect::get (
    int & x,
    int & y,
    int & width,
    int & height ) const
```

Parameters

out	<i>x</i>	The destination x coordinate.
out	<i>y</i>	The destination y coordinate.
out	<i>width</i>	The destination width value.
out	<i>height</i>	The destination width value.

10.75.3.2 set()

```
void seq64::rect::set (
    int x,
    int y,
    int width,
    int height )
```

Parameters

<i>x</i>	The x coordinate.
<i>y</i>	The y coordinate.
<i>width</i>	The width value.
<i>height</i>	The width value.

10.75.3.3 clear()

```
void seq64::rect::clear ( ) [inline]
```

10.75.3.4 xy_to_rect() [1/2]

```
void seq64::rect::xy_to_rect (
    int x1,
    int y1,
    int x2,
    int y2,
    rect & r ) [static]
```

Parameters

	<i>x1</i>	The x value of the first corner.
	<i>y1</i>	The y value of the first corner.
	<i>x2</i>	The x value of the second corner.
	<i>y2</i>	The y value of the second corner.
out	<i>r</i>	The destination for the coordinate, width, and height.

10.75.3.5 xy_to_rect_get()

```
void seq64::rect::xy_to_rect_get (
    int x1,
    int y1,
    int x2,
```

```

    int y2,
    int & x,
    int & y,
    int & w,
    int & h ) [static]

```

This function checks the mins / maxes, and then fills in the x, y, width, and height values. It picks the lowest x and y coordinate to use as the corner coordinate, so that the width and height are always positive.

Parameters

	<i>x1</i>	The x value of the first corner.
	<i>y1</i>	The y value of the first corner.
	<i>x2</i>	The x value of the second corner.
	<i>y2</i>	The y value of the second corner.
out	<i>x</i>	The destination for the x value in pixels.
out	<i>y</i>	The destination for the y value in pixels.
out	<i>w</i>	The destination for the rectangle width in pixels.
out	<i>h</i>	The destination for the rectangle height value in pixels.

10.75.3.6 xy_to_rect() [2/2]

```

void seq64::rect::xy_to_rect (
    int x1,
    int y1,
    int x2,
    int y2 ) [inline]

```

10.75.3.7 x() [1/2]

```

int seq64::rect::x ( ) const [inline]

```

10.75.3.8 x() [2/2]

```

void seq64::rect::x (
    int v ) [inline]

```

10.75.3.9 x_incr()

```

void seq64::rect::x_incr (
    int v ) [inline]

```

The width is assumed to be unchanged by this function.

10.75.3.10 `y()` [1/2]

```
int seq64::rect::y ( ) const [inline]
```

10.75.3.11 `y()` [2/2]

```
void seq64::rect::y (
    int v ) [inline]
```

10.75.3.12 `y_incr()`

```
void seq64::rect::y_incr (
    int v ) [inline]
```

The height is assumed to be unchanged by this function.

10.75.3.13 `width()` [1/2]

```
int seq64::rect::width ( ) const [inline]
```

10.75.3.14 `width()` [2/2]

```
void seq64::rect::width (
    int w ) [inline]
```

10.75.3.15 `width_incr()`

```
void seq64::rect::width_incr (
    int w ) [inline]
```

10.75.3.16 `height()` [1/2]

```
int seq64::rect::height ( ) const [inline]
```

10.75.3.17 height() [2/2]

```
void seq64::rect::height (
    int h ) [inline]
```

10.75.3.18 height_incr()

```
void seq64::rect::height_incr (
    int h ) [inline]
```

10.75.3.19 xy_incr()

```
void seq64::rect::xy_incr (
    int xv,
    int yv ) [inline]
```

10.75.3.20 calculated_width()

```
static int seq64::rect::calculated_width (
    int x1,
    int x2 ) [inline], [static], [private]
```

Follows the conventions of the [xy_to_rect_get\(\)](#) function.

10.75.3.21 calculated_height()

```
static int seq64::rect::calculated_height (
    int y1,
    int y2 ) [inline], [static], [private]
```

Follows the conventions of the [xy_to_rect_get\(\)](#) function.

10.75.4 Field Documentation

10.75.4.1 m_x

```
int seq64::rect::m_x [private]
```

10.75.4.2 m_y

```
int seq64::rect::m_y [private]
```

10.75.4.3 m_width

```
int seq64::rect::m_width [private]
```

10.75.4.4 m_height

```
int seq64::rect::m_height [private]
```

10.76 seq64::rterror Class Reference

Exception handling class for rtxmidi.

Inherits exception.

Public Types

- enum [Type](#) {
 [WARNING](#),
 [DEBUG_WARNING](#),
 [UNSPECIFIED](#),
 [NO_DEVICES_FOUND](#),
 [INVALID_DEVICE](#),
 [MEMORY_ERROR](#),
 [INVALID_PARAMETER](#),
 [INVALID_USE](#),
 [DRIVER_ERROR](#),
 [SYSTEM_ERROR](#),
 [THREAD_ERROR](#) }

Public Member Functions

- [rterror](#) (const std::string &message, [Type](#) type=[rterror::UNSPECIFIED](#))
- virtual [~rterror](#) ()
- virtual void [print_message](#) () const
Prints thrown error message to stderr.
- virtual const [Type](#) & [getType](#) () const
Returns the thrown error message type.
- virtual const std::string & [get_message](#) () const
Returns the thrown error message string.
- virtual const char * [what](#) () const noexcept
Returns the thrown error message as a c-style string.

Private Attributes

- `std::string m_message`
Holds the latest message information for the exception.
- `Type m_type`
Holds the type or severity of the exception.

10.76.1 Detailed Description

The `rterror` class is quite simple but it does allow errors to be "caught" by `rterror::Type`. See the `rtexmidi` documentation to know which methods can throw an `rterror`.

Please note that, in this refactoring of `rtmidi`, we've done away with all the exception specifications, on the advice of Herb Sutter. They may be more relevant to C++11 and beyond, but this library is too small to worry about them, for now.

10.76.2 Member Enumeration Documentation

10.76.2.1 Type

enum `seq64::rterror::Type`

Enumerator

<code>WARNING</code>	A non-critical error.
<code>DEBUG_WARNING</code>	Non-critical error useful for debugging.
<code>UNSPECIFIED</code>	The default, unspecified error type.
<code>NO_DEVICES_FOUND</code>	No devices found on system.
<code>INVALID_DEVICE</code>	An invalid device ID was specified.
<code>MEMORY_ERROR</code>	An error occurred during memory allocation.
<code>INVALID_PARAMETER</code>	Invalid parameter specified to a function.
<code>INVALID_USE</code>	The function was called incorrectly.
<code>DRIVER_ERROR</code>	A system driver error occurred.
<code>SYSTEM_ERROR</code>	A system error occurred.
<code>THREAD_ERROR</code>	A thread error occurred.

10.76.3 Constructor & Destructor Documentation

10.76.3.1 `rterror()`

```
seq64::rterror::rterror (
    const std::string & message,
    Type type = rterror::UNSPECIFIED ) [inline]
```

10.76.3.2 ~rterror()

```
virtual seq64::rterror::~~rterror ( ) [inline], [virtual]
```

10.76.4 Member Function Documentation

10.76.4.1 print_message()

```
virtual void seq64::rterror::print_message ( ) const [inline], [virtual]
```

10.76.4.2 getType()

```
virtual const Type& seq64::rterror::getType ( ) const [inline], [virtual]
```

10.76.4.3 get_message()

```
virtual const std::string& seq64::rterror::get_message ( ) const [inline], [virtual]
```

10.76.4.4 what()

```
virtual const char* seq64::rterror::what ( ) const [inline], [virtual], [noexcept]
```

10.76.5 Field Documentation

10.76.5.1 m_message

```
std::string seq64::rterror::m_message [private]
```

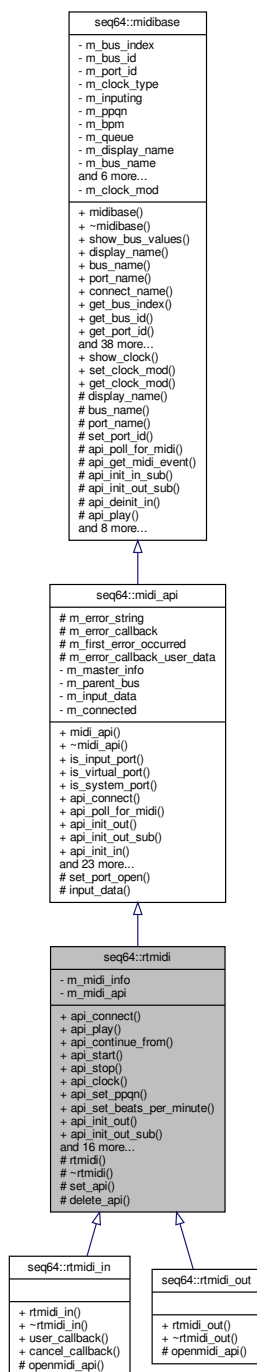
10.76.5.2 m_type

Type seq64::rterror::m_type [private]

10.77 seq64::rtmidi Class Reference

The main class of the rtmidi API.

Inheritance diagram for seq64::rtmidi:



Public Member Functions

- virtual bool [api_connect](#) ()
- virtual void [api_play](#) ([event](#) *e24, [midibyte](#) channel)
- virtual void [api_continue_from](#) ([midipulse](#) tick, [midipulse](#) beats)
- virtual void [api_start](#) ()
- virtual void [api_stop](#) ()
- virtual void [api_clock](#) ([midipulse](#) tick)
- virtual void [api_set_ppqn](#) (int [ppqn](#))
- virtual void [api_set_beats_per_minute](#) ([midibpm](#) bpm)
- virtual bool [api_init_out](#) ()
- virtual bool [api_init_out_sub](#) ()
- virtual bool [api_init_in](#) ()
- virtual bool [api_init_in_sub](#) ()
- virtual bool [api_deinit_in](#) ()
- virtual bool [api_get_midi_event](#) ([event](#) *inev)
- virtual int [api_poll_for_midi](#) ()
- virtual void [api_sysex](#) ([event](#) *e24)
- virtual void [api_flush](#) ()
- virtual bool [is_port_open](#) () const

Returns true if a port is open and false if not.

- virtual int [get_bus_id](#) ()
- Gets the buss/client ID for a MIDI interfaces.*
- virtual std::string [get_bus_name](#) ()
- virtual int [get_port_id](#) ()
- virtual std::string [get_port_name](#) ()
- int [get_port_count](#) ()
- int [full_port_count](#) ()
- const [midi_api](#) * [get_api](#) () const
- 'Getter' function for member m_midi_api const version*
- [midi_api](#) * [get_api](#) ()
- 'Getter' function for member m_midi_api non-const version*

Protected Member Functions

- [rtmidi](#) ([midibus](#) &parentbus, [rtmidi_info](#) &info)
- virtual [~rtmidi](#) ()
- void [set_api](#) ([midi_api](#) *ma)
- 'Setter' function for member m_midi_api*
- void [delete_api](#) ()
- 'Setter' function for member m_midi_api*

Private Attributes

- [rtmidi_info](#) & [m_midi_info](#)
- Holds a reference to the "global" [midi_info](#) wrapper object.*
- [midi_api](#) * [m_midi_api](#)
- Points to the API I/O object (e.g.*

Friends

- class [midibus](#)

Additional Inherited Members

10.77.1 Detailed Description

We moved the enum Api definition into the new `rtmidi_types.hpp` module to make refactoring the code easier.

10.77.2 Constructor & Destructor Documentation

10.77.2.1 `rtmidi()`

```
seq64::rtmidi::rtmidi (
    midibus & parentbus,
    rtmidi\_info & info ) [protected]
```

10.77.2.2 `~rtmidi()`

```
virtual seq64::rtmidi::~~rtmidi ( ) [protected], [virtual]
```

10.77.3 Member Function Documentation

10.77.3.1 `api_connect()`

```
virtual bool seq64::rtmidi::api_connect ( ) [inline], [virtual]
```

Reimplemented from [seq64::midi_api](#).

10.77.3.2 `api_play()`

```
virtual void seq64::rtmidi::api_play (
    event * e24,
    midibyte channel ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.3 `api_continue_from()`

```
virtual void seq64::rtmidi::api_continue_from (
    midipulse tick,
    midipulse beats ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.4 `api_start()`

```
virtual void seq64::rtmidi::api_start ( ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.5 `api_stop()`

```
virtual void seq64::rtmidi::api_stop ( ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.6 `api_clock()`

```
virtual void seq64::rtmidi::api_clock (
    midipulse tick ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.7 `api_set_ppqn()`

```
virtual void seq64::rtmidi::api_set_ppqn (
    int ppqn ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.8 `api_set_beats_per_minute()`

```
virtual void seq64::rtmidi::api_set_beats_per_minute (
    midibpm bpm ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.9 `api_init_out()`

```
virtual bool seq64::rtmidi::api_init_out ( ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.10 `api_init_out_sub()`

```
virtual bool seq64::rtmidi::api_init_out_sub ( ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.11 `api_init_in()`

```
virtual bool seq64::rtmidi::api_init_in ( ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.12 `api_init_in_sub()`

```
virtual bool seq64::rtmidi::api_init_in_sub ( ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.13 `api_deinit_in()`

```
virtual bool seq64::rtmidi::api_deinit_in ( ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.14 `api_get_midi_event()`

```
virtual bool seq64::rtmidi::api_get_midi_event (
    event * inev ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.15 api_poll_for_midi()

```
virtual int seq64::rtmidi::api_poll_for_midi ( ) [inline], [virtual]
```

Reimplemented from [seq64::midi_api](#).

10.77.3.16 api_sysex()

```
virtual void seq64::rtmidi::api_sysex (
    event * e24 ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.17 api_flush()

```
virtual void seq64::rtmidi::api_flush ( ) [inline], [virtual]
```

Implements [seq64::midi_api](#).

10.77.3.18 is_port_open()

```
virtual bool seq64::rtmidi::is_port_open ( ) const [inline], [virtual]
```

10.77.3.19 get_bus_id()

```
virtual int seq64::rtmidi::get_bus_id ( ) [inline], [virtual]
```

This is the left-hand side of a X:Y pair (such as 128:0).

This function is a new part of the RtMidi interface.

Returns

Returns the buss/client value as provided by the selected API.

10.77.3.20 get_bus_name()

```
virtual std::string seq64::rtmidi::get_bus_name ( ) [inline], [virtual]
```

Returns

Returns the buss name from the selected API subsystem.

10.77.3.21 get_port_id()

```
virtual int seq64::rtmidi::get_port_id ( ) [inline], [virtual]
```

Returns

Returns the port ID number from the selected API subsystem.

10.77.3.22 get_port_name()

```
virtual std::string seq64::rtmidi::get_port_name ( ) [inline], [virtual]
```

Returns

Returns the port name from the selected API subsystem.

10.77.3.23 get_port_count()

```
int seq64::rtmidi::get_port_count ( ) [inline]
```

Returns

This value depends on the MIDI mode setting (input versus output).

10.77.3.24 full_port_count()

```
int seq64::rtmidi::full_port_count ( ) [inline]
```

Returns

This value is the sum of the number of input and output ports.

10.77.3.25 get_api() [1/2]

```
const midi_api* seq64::rtmidi::get_api ( ) const [inline]
```

10.77.3.26 get_api() [2/2]

```
midi_api* seq64::rtmidi::get_api ( ) [inline]
```

10.77.3.27 set_api()

```
void seq64::rtmidi::set_api (
    midi_api * ma ) [inline], [protected]
```

10.77.3.28 delete_api()

```
void seq64::rtmidi::delete_api ( ) [inline], [protected]
```

10.77.4 Friends And Related Function Documentation

10.77.4.1 midibus

```
friend class midibus [friend]
```

10.77.5 Field Documentation

10.77.5.1 m_midi_info

```
rtmidi_info& seq64::rtmidi::m_midi_info [private]
```

Unlike the original RtMidi library, this library separates the port-enumeration code ("info") from the port-usage code ("api").

We might make it a static object at some point.

10.77.5.2 m_midi_api

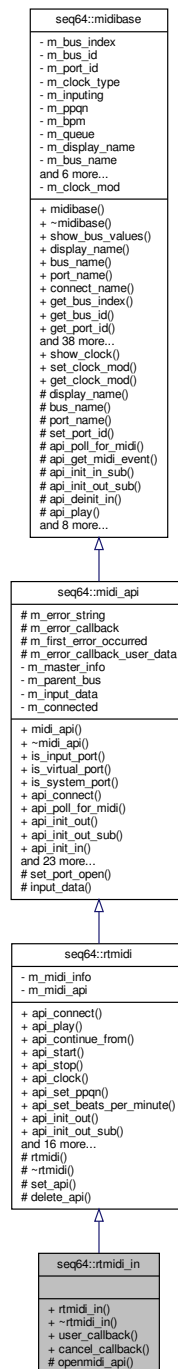
```
midi_api* seq64::rtmidi::m_midi_api [private]
```

[midi_alsa](#) or [midi_jack](#)) for which this class is a wrapper.

10.78 seq64::rtmidi_in Class Reference

A realtime MIDI input class.

Inheritance diagram for seq64::rtmidi_in:



Public Member Functions

- [rtmidi_in](#) ([midibus](#) &parentbus, [rtmidi_info](#) &info)
- virtual [~rtmidi_in](#) ()
- void [user_callback](#) ([rtmidi_callback_t](#) callback, void *userdata=nullptr)
Set a callback function to be invoked for incoming MIDI messages.
- void [cancel_callback](#) ()
Cancel use of the current callback function (if one exists).

Protected Member Functions

- void `openmidi_api` (`rtmidi_api` api, `rtmidi_info` &info)

Additional Inherited Members

10.78.1 Detailed Description

This class provides a common, platform-independent API for realtime MIDI input. It allows access to a single MIDI input port. Incoming MIDI messages are either saved to a queue for retrieval using the `get_message()` function or immediately passed to a user-specified callback function. Create multiple instances of this class to connect to more than one MIDI device at the same time. With the OS-X, Linux ALSA, and JACK MIDI APIs, it is also possible to open a virtual input port to which other MIDI software clients can connect.

10.78.2 Constructor & Destructor Documentation

10.78.2.1 `rtmidi_in()`

```
seq64::rtmidi_in::rtmidi_in (
    midibus & parentbus,
    rtmidi_info & info )
```

10.78.2.2 `~rtmidi_in()`

```
virtual seq64::rtmidi_in::~~rtmidi_in ( ) [virtual]
```

10.78.3 Member Function Documentation

10.78.3.1 `user_callback()`

```
void seq64::rtmidi_in::user_callback (
    rtmidi_callback_t callback,
    void * userdata = nullptr ) [inline]
```

The callback function will be called whenever an incoming MIDI message is received. While not absolutely necessary, it is best to set the callback function before opening a MIDI port to avoid leaving some messages in the queue.

Parameters

<i>callback</i>	A callback function must be given.
<i>userdata</i>	Optionally, a pointer to additional data can be passed to the callback function whenever it is called.

10.78.3.2 cancel_callback()

```
void seq64::rtmidi_in::cancel_callback ( ) [inline]
```

Subsequent incoming MIDI messages will be written to the queue and can be retrieved with the *get_message* function.

10.78.3.3 openmidi_api()

```
void seq64::rtmidi_in::openmidi_api (
    rtmidi_api api,
    rtmidi_info & info ) [protected]
```

10.79 seq64::rtmidi_in_data Class Reference

The *rtmidi_in_data* structure is used to pass private class data to the MIDI input handling function or thread.

Public Member Functions

- *rtmidi_in_data* ()
- const *midi_queue* & *queue* () const
'Getter' function for member m_queue const
- *midi_queue* & *queue* ()
'Getter' function for member m_queue non-const
- const *midi_message* & *message* () const
- *midi_message* & *message* ()
- *midibyte* *ignore_flags* () const
- bool *test_ignore_flags* (*midibyte* testbits)
- void *ignore_flags* (*midibyte* setbits)
- bool *do_input* () const
- void *do_input* (bool flag)
- bool *first_message* () const
- void *first_message* (bool flag)
- bool *continue_sysex* () const
- void *continue_sysex* (bool flag)
- bool *using_callback* () const
- void *using_callback* (bool flag)
- const void * *api_data* () const
'Getter' function for member m_api_data const
- void * *api_data* ()

'Getter' function for member m_api_data non-const

- void [api_data](#) (void *dataptr)
- const void * [user_data](#) () const

'Getter' function for member m_user_data const

- void * [user_data](#) ()

'Getter' function for member m_user_data const

- void [user_data](#) (void *dataptr)

'Setter' function for member m_user_data

- [rtmidi_callback_t](#) [user_callback](#) () const

'Getter' function for member m_user_callback

- void [user_callback](#) ([rtmidi_callback_t](#) cbptr)

'Setter' function for member m_user_callback This should be done immediately after opening the port to avoid having incoming messages written to the queue instead of sent to the callback function.

Private Attributes

- [midi_queue](#) [m_queue](#)
- [midi_message](#) [m_message](#)
- [midibyte](#) [m_ignore_flags](#)
- bool [m_do_input](#)
- bool [m_first_message](#)
- void * [m_api_data](#)
- bool [m_using_callback](#)
- [rtmidi_callback_t](#) [m_user_callback](#)
- void * [m_user_data](#)
- bool [m_continue_sysex](#)

10.79.1 Detailed Description

Used to be nested in the [rtmidi_in](#) class.

10.79.2 Constructor & Destructor Documentation

10.79.2.1 [rtmidi_in_data](#)()

```
seq64::rtmidi_in_data::rtmidi_in_data ( )
```

10.79.3 Member Function Documentation

10.79.3.1 queue() [1/2]

```
const midi_queue& seq64::rtmidi_in_data::queue ( ) const [inline]
```

10.79.3.2 queue() [2/2]

```
midi_queue& seq64::rtmidi_in_data::queue ( ) [inline]
```

10.79.3.3 message() [1/2]

```
const midi_message& seq64::rtmidi_in_data::message ( ) const [inline]
```

10.79.3.4 message() [2/2]

```
midi_message& seq64::rtmidi_in_data::message ( ) [inline]
```

10.79.3.5 ignore_flags() [1/2]

```
midibyte seq64::rtmidi_in_data::ignore_flags ( ) const [inline]
```

10.79.3.6 test_ignore_flags()

```
bool seq64::rtmidi_in_data::test_ignore_flags (
    midibyte testbits ) [inline]
```

10.79.3.7 ignore_flags() [2/2]

```
void seq64::rtmidi_in_data::ignore_flags (
    midibyte setbits ) [inline]
```

10.79.3.8 do_input() [1/2]

```
bool seq64::rtmidi_in_data::do_input ( ) const [inline]
```

10.79.3.9 do_input() [2/2]

```
void seq64::rtmidi_in_data::do_input (
    bool flag ) [inline]
```

10.79.3.10 first_message() [1/2]

```
bool seq64::rtmidi_in_data::first_message ( ) const [inline]
```

10.79.3.11 first_message() [2/2]

```
void seq64::rtmidi_in_data::first_message (
    bool flag ) [inline]
```

10.79.3.12 continue_sysex() [1/2]

```
bool seq64::rtmidi_in_data::continue_sysex ( ) const [inline]
```

10.79.3.13 continue_sysex() [2/2]

```
void seq64::rtmidi_in_data::continue_sysex (
    bool flag ) [inline]
```

10.79.3.14 using_callback() [1/2]

```
bool seq64::rtmidi_in_data::using_callback ( ) const [inline]
```

10.79.3.15 using_callback() [2/2]

```
void seq64::rtmidi_in_data::using_callback (
    bool flag ) [inline]
```

10.79.3.16 api_data() [1/3]

```
const void* seq64::rtmidi_in_data::api_data ( ) const [inline]
```

10.79.3.17 api_data() [2/3]

```
void* seq64::rtmidi_in_data::api_data ( ) [inline]
```

10.79.3.18 api_data() [3/3]

```
void seq64::rtmidi_in_data::api_data (
    void * dataptr ) [inline]
```

10.79.3.19 user_data() [1/3]

```
const void* seq64::rtmidi_in_data::user_data ( ) const [inline]
```

10.79.3.20 user_data() [2/3]

```
void* seq64::rtmidi_in_data::user_data ( ) [inline]
```

10.79.3.21 user_data() [3/3]

```
void seq64::rtmidi_in_data::user_data (
    void * dataptr ) [inline]
```

10.79.3.22 user_callback() [1/2]

```
rtmidi_callback_t seq64::rtmidi_in_data::user_callback ( ) const [inline]
```

10.79.3.23 user_callback() [2/2]

```
void seq64::rtmidi_in_data::user_callback (
    rtmidi_callback_t cbptr ) [inline]
```

10.79.4 Field Documentation

10.79.4.1 m_queue

```
midi_queue seq64::rtmidi_in_data::m_queue [private]
```

10.79.4.2 m_message

```
midi_message seq64::rtmidi_in_data::m_message [private]
```

10.79.4.3 m_ignore_flags

```
midibyte seq64::rtmidi_in_data::m_ignore_flags [private]
```

10.79.4.4 m_do_input

```
bool seq64::rtmidi_in_data::m_do_input [private]
```

10.79.4.5 m_first_message

```
bool seq64::rtmidi_in_data::m_first_message [private]
```

10.79.4.6 m_api_data

```
void* seq64::rtmidi_in_data::m_api_data [private]
```

10.79.4.7 m_using_callback

```
bool seq64::rtmidi_in_data::m_using_callback [private]
```

10.79.4.8 m_user_callback

```
rtmidi_callback_t seq64::rtmidi_in_data::m_user_callback [private]
```

10.79.4.9 m_user_data

```
void* seq64::rtmidi_in_data::m_user_data [private]
```

10.79.4.10 m_continue_sysex

```
bool seq64::rtmidi_in_data::m_continue_sysex [private]
```

10.80 seq64::rtmidi_info Class Reference

A class for enumerating MIDI clients and ports.

Public Member Functions

- `rtmidi_info` (`rtmidi_api` `api=RTMIDI_API_UNSPECIFIED`, `const std::string &appname="rtmidiapp"`, `int ppqn=SEQ64_DEFAULT_PPQN`, `midibpm bpm=SEQ64_DEFAULT_BPM`)
- `virtual ~rtmidi_info ()`
- `bool midi_mode () const`
Sets the input or output mode for getting data.
- `void midi_mode (bool flag)`
Sets the input or output mode for getting data.
- `void clear ()`
Clear the MIDI port container.
- `void add_input (const midibus *m)`
Add midibus information to the input ports.
- `void add_output (const midibus *m)`
Add midibus information to the output ports.
- `void add_bus (const midibus *m)`
Adds the bus to a list of busses to be connected by the API at the right time (currently applies only to JACK).
- `int get_bus_id (int index) const`
Gets the buss/client ID for a MIDI interfaces.
- `std::string get_bus_name (int index) const`
- `int get_port_count () const`
- `int full_port_count () const`
- `int get_port_id (int index) const`
- `std::string get_port_name (int index) const`
- `bool get_input (int index) const`
- `bool get_virtual (int index) const`
- `bool get_system (int index) const`
- `int get_all_port_info ()`
- `int queue_number (int index) const`
- `const std::string & app_name () const`
- `int global_queue () const`
- `int ppqn () const`
- `void api_set_ppqn (int p)`
- `midibpm bpm () const`
- `void api_set_beats_per_minute (midibpm b)`
- `void api_port_start (mastermidibus &masterbus, int bus, int port)`
- `bool api_get_midi_event (event *inev)`
- `void api_flush ()`
- `int api_poll_for_midi ()`
- `std::string port_list () const`
Returns a list of all the ports as an ASCII string.
- `const rtmidi_info * get_api_info () const`
'Getter' function for member m_info_api const version
- `rtmidi_info * get_api_info ()`
'Getter' function for member m_info_api non-const version

Static Public Member Functions

- `static std::string get_version ()`
- `static void get_compiled_api (std::vector< rtmidi_api > &apis)`
- `static rtmidi_api & selected_api ()`
'Getter' function for member sm_selected_api

Protected Member Functions

- bool [api_connect](#) ()
- bool [set_api_info](#) ([midi_info](#) *ma)
'Setter' function for member m_info_api This function also checks the pointer and returns false if it is not valid.
- void [delete_api](#) ()
'Setter' function for member m_info_api
- bool [openmidi_api](#) ([rtmidi_api](#) api, const std::string &appname, int [ppqn](#), [midibpm](#) bpm)

Static Protected Member Functions

- static void [selected_api](#) (const [rtmidi_api](#) &api)
'Setter' function for member sm_selected_api

Private Attributes

- [midi_info](#) * [m_info_api](#)
Provides access to the selected API (currently only JACK or ALSA).

Static Private Attributes

- static [rtmidi_api](#) [sm_selected_api](#)
To save repeated queries, we save this value.

Friends

- class [mastermidibus](#)
- class [midibus](#)
- class [rtmidi_in](#)
- class [rtmidi_out](#)

10.80.1 Detailed Description

New, but ripe for refactoring nonetheless.

10.80.2 Constructor & Destructor Documentation

10.80.2.1 rtmidi_info()

```
seq64::rtmidi_info::rtmidi_info (
    rtmidi\_api api = RTMIDI\_API\_UNSPECIFIED,
    const std::string & appname = "rtmidiapp",
    int ppqn = SEQ64\_DEFAULT\_PPQN,
    midibpm bpm = SEQ64\_DEFAULT\_BPM )
```

10.80.2.2 ~rtmidi_info()

```
virtual seq64::rtmidi_info::~~rtmidi_info ( ) [virtual]
```

10.80.3 Member Function Documentation

10.80.3.1 get_version()

```
static std::string seq64::rtmidi_info::get_version ( ) [static]
```

10.80.3.2 get_compiled_api()

```
static void seq64::rtmidi_info::get_compiled_api (
    std::vector< rtmidi_api > & apis ) [static]
```

10.80.3.3 midi_mode() [1/2]

```
bool seq64::rtmidi_info::midi_mode ( ) const [inline]
```

10.80.3.4 midi_mode() [2/2]

```
void seq64::rtmidi_info::midi_mode (
    bool flag ) [inline]
```

10.80.3.5 clear()

```
void seq64::rtmidi_info::clear ( ) [inline]
```

10.80.3.6 add_input()

```
void seq64::rtmidi_info::add_input (
    const midibus * m ) [inline]
```

Also adds the midibus to a list of busses to connect in mastermidibus. This function is meant for virtual ports.

10.80.3.7 add_output()

```
void seq64::rtmidi_info::add_output (
    const midibus * m ) [inline]
```

Also adds the midibus to a list of busses to connect in mastermidibus. This function is meant for virtual ports.

10.80.3.8 add_bus()

```
void seq64::rtmidi_info::add_bus (
    const midibus * m ) [inline]
```

See the calls to this function in mastermidibus.

10.80.3.9 get_bus_id()

```
int seq64::rtmidi_info::get_bus_id (
    int index ) const [inline]
```

This is the left-hand side of a X:Y pair (such as 128:0).

This function is a new part of the RtMidi interface.

Parameters

<i>index</i>	The ordinal index of the desired interface to look up.
--------------	--

Returns

Returns the buss/client value as provided by the selected API.

10.80.3.10 get_bus_name()

```
std::string seq64::rtmidi_info::get_bus_name (
    int index ) const [inline]
```

10.80.3.11 get_port_count()

```
int seq64::rtmidi_info::get_port_count ( ) const [inline]
```

10.80.3.12 full_port_count()

```
int seq64::rtmidi_info::full_port_count ( ) const [inline]
```

10.80.3.13 get_port_id()

```
int seq64::rtmidi_info::get_port_id (
    int index ) const [inline]
```

10.80.3.14 get_port_name()

```
std::string seq64::rtmidi_info::get_port_name (
    int index ) const [inline]
```

10.80.3.15 get_input()

```
bool seq64::rtmidi_info::get_input (
    int index ) const [inline]
```

10.80.3.16 get_virtual()

```
bool seq64::rtmidi_info::get_virtual (
    int index ) const [inline]
```

10.80.3.17 get_system()

```
bool seq64::rtmidi_info::get_system (
    int index ) const [inline]
```

10.80.3.18 get_all_port_info()

```
int seq64::rtmidi_info::get_all_port_info ( ) [inline]
```

10.80.3.19 queue_number()

```
int seq64::rtmidi_info::queue_number (
    int index ) const [inline]
```

10.80.3.20 app_name()

```
const std::string& seq64::rtmidi_info::app_name ( ) const [inline]
```

10.80.3.21 global_queue()

```
int seq64::rtmidi_info::global_queue ( ) const [inline]
```

10.80.3.22 ppqn()

```
int seq64::rtmidi_info::ppqn ( ) const [inline]
```

10.80.3.23 api_set_ppqn()

```
void seq64::rtmidi_info::api_set_ppqn (
    int p ) [inline]
```

10.80.3.24 bpm()

```
midibpm seq64::rtmidi_info::bpm ( ) const [inline]
```

10.80.3.25 api_set_beats_per_minute()

```
void seq64::rtmidi_info::api_set_beats_per_minute (
    midibpm b ) [inline]
```

10.80.3.26 `api_port_start()`

```
void seq64::rtmidi_info::api_port_start (
    mastermidibus & masterbus,
    int bus,
    int port ) [inline]
```

10.80.3.27 `api_get_midi_event()`

```
bool seq64::rtmidi_info::api_get_midi_event (
    event * inev ) [inline]
```

10.80.3.28 `api_flush()`

```
void seq64::rtmidi_info::api_flush ( ) [inline]
```

10.80.3.29 `api_poll_for_midi()`

```
int seq64::rtmidi_info::api_poll_for_midi ( ) [inline]
```

10.80.3.30 `port_list()`

```
std::string seq64::rtmidi_info::port_list ( ) const [inline]
```

10.80.3.31 `selected_api()` [1/2]

```
static rtmidi_api& seq64::rtmidi_info::selected_api ( ) [inline], [static]
```

10.80.3.32 `get_api_info()` [1/2]

```
const midi_info* seq64::rtmidi_info::get_api_info ( ) const [inline]
```

10.80.3.33 get_api_info() [2/2]

```
midi_info* seq64::rtmidi_info::get_api_info ( ) [inline]
```

10.80.3.34 api_connect()

```
bool seq64::rtmidi_info::api_connect ( ) [inline], [protected]
```

10.80.3.35 selected_api() [2/2]

```
static void seq64::rtmidi_info::selected_api (
    const rtmidi_api & api ) [inline], [static], [protected]
```

10.80.3.36 set_api_info()

```
bool seq64::rtmidi_info::set_api_info (
    midi_info * ma ) [inline], [protected]
```

This feature is important to allow a missing API (e.g. the JACK server is not running) to be detected.

10.80.3.37 delete_api()

```
void seq64::rtmidi_info::delete_api ( ) [inline], [protected]
```

10.80.3.38 openmidi_api()

```
bool seq64::rtmidi_info::openmidi_api (
    rtmidi_api api,
    const std::string & appname,
    int ppqn,
    midibpm bpm ) [protected]
```

10.80.4 Friends And Related Function Documentation

10.80.4.1 mastermidibus

```
friend class mastermidibus [friend]
```

10.80.4.2 midibus

```
friend class midibus [friend]
```

10.80.4.3 rtmidi_in

```
friend class rtmidi_in [friend]
```

10.80.4.4 rtmidi_out

```
friend class rtmidi_out [friend]
```

10.80.5 Field Documentation

10.80.5.1 m_info_api

```
midi_info* seq64::rtmidi_info::m_info_api [private]
```

10.80.5.2 sm_selected_api

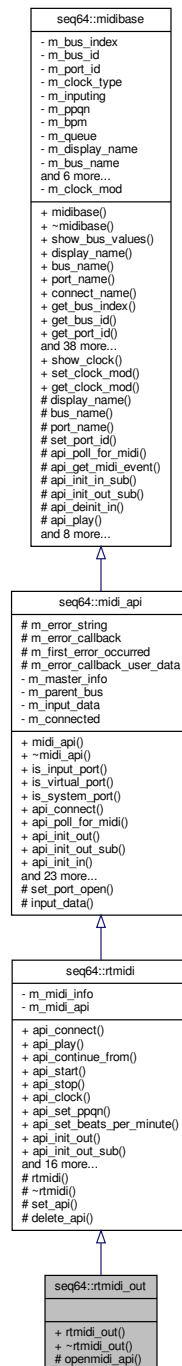
```
rtmidi_api seq64::rtmidi_info::sm_selected_api [static], [private]
```

Its default value is RTMIDI_API_UNSPECIFIED.

10.81 seq64::rtmidi_out Class Reference

A realtime MIDI output class.

Inheritance diagram for seq64::rtmidi_out:



Public Member Functions

- [rtmidi_out](#) ([midibus](#) &parentbus, [rtmidi_info](#) &info)

- virtual `~rtmidi_out()`

The destructor closes any open MIDI connections.

Protected Member Functions

- void `openmidi_api` (`rtmidi_api` api, `rtmidi_info` &info)

Additional Inherited Members

10.81.1 Detailed Description

This class provides a common, platform-independent API for MIDI output. It allows one to probe available MIDI output ports, to connect to one such port, and to send MIDI bytes immediately over the connection. Create multiple instances of this class to connect to more than one MIDI device at the same time. With the OS-X, Linux ALSA and JACK MIDI APIs, it is also possible to open a virtual port to which other MIDI software clients can connect.

10.81.2 Constructor & Destructor Documentation

10.81.2.1 `rtmidi_out()`

```
seq64::rtmidi_out::rtmidi_out (
    midibus & parentbus,
    rtmidi_info & info )
```

10.81.2.2 `~rtmidi_out()`

```
virtual seq64::rtmidi_out::~~rtmidi_out ( ) [virtual]
```

10.81.3 Member Function Documentation

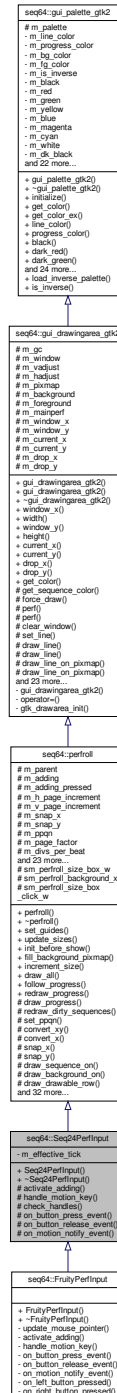
10.81.3.1 `openmidi_api()`

```
void seq64::rtmidi_out::openmidi_api (
    rtmidi_api api,
    rtmidi_info & info ) [protected]
```

10.82 seq64::Seq24PerfInput Class Reference

Implements the default (Seq24) performance input characteristics of this application.

Inheritance diagram for seq64::Seq24PerfInput:



Public Member Functions

- [Seq24PerfInput](#) (perform &perf, perfedit &parent, Gtk::Adjustment &hadjust, Gtk::Adjustment &vadjust, int ppqn=SEQ64_USE_DEFAULT_PPQN)

Principal constructor.

- [~Seq24PerfInput](#) ()
virtual destructor

Protected Member Functions

- virtual void [activate_adding](#) (bool a_adding)
Turns on/off the mode of adding triggers to the song performance.
- virtual bool [handle_motion_key](#) (bool is_left)
Handles the keystroke motion-notify event for moving a pattern back and forth in the performance.
- bool [check_handles](#) ()
Pulled this function out to simplify the on-button-press callback.
- virtual bool [on_button_press_event](#) (GdkEventButton *ev)
Initial unselection:
- virtual bool [on_button_release_event](#) (GdkEventButton *ev)
Handles various button-release events.
- virtual bool [on_motion_notify_event](#) (GdkEventMotion *ev)
Handles the normal motion-notify event.

Private Attributes

- [midipulse m_effective_tick](#)
The current tick for the current segment?

Additional Inherited Members

10.82.1 Constructor & Destructor Documentation

10.82.1.1 Seq24PerfInput()

```
seq64::Seq24PerfInput::Seq24PerfInput (
    perform & p,
    perfedit & parent,
    Gtk::Adjustment & hadjust,
    Gtk::Adjustment & vadjust,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

Parameters

<i>p</i>	The perform object that this class will affect via user interaction.
<i>parent</i>	The perfedit object that holds this user-interface class.
<i>hadjust</i>	A horizontal adjustment object, passed along to the perfrroll class.
<i>vadjust</i>	A vertical adjustment object, passed along to the perfrroll class.
<i>ppqn</i>	The "resolution" of the MIDI file, used in zooming and scaling.

10.82.1.2 ~Seq24PerfInput()

```
seq64::Seq24PerfInput::~~Seq24PerfInput ( ) [inline]
```

10.82.2 Member Function Documentation

10.82.2.1 activate_adding()

```
void seq64::Seq24PerfInput::activate_adding (
    bool adding ) [protected], [virtual]
```

Changes both the flag and the mouse cursor icon.

Parameters

<i>adding</i>	Indicates the adding-triggers state to be set.
---------------	--

Implements [seq64::perfroll](#).

Reimplemented in [seq64::FruityPerfInput](#).

10.82.2.2 handle_motion_key()

```
bool seq64::Seq24PerfInput::handle_motion_key (
    bool is_left ) [protected], [virtual]
```

What happens when the mouse is used to drag the pattern is that, first, `m_drop_tick` is set by left-clicking into the pattern to select it. As the pattern is dragged, the drop-tick value does not change, but the tick (converted from the moving x value) does.

Then the button-handler sets `m_moving = true`, and calculates `m_drop_tick_offset = m_drop_tick - (drop sequence)->selected_trigger_start()`. The motion handler sees that `m_moving` is true, gets the new tick value from the new x value, offsets it, and calls `(drop sequence)->move_triggers(tick, true)`. When the user releases the left button, then `m_growing` is turned off and the roll [draw_all\(\)](#)'s.

Parameters

<i>is_left</i>	False denotes the right arrow key, and true denotes the left arrow key.
----------------	---

Returns

Returns true if there was some action able to happen that would necessitate a window update. We've updated `triggers::move_triggers()` [called indirectly near the end of this routine] to return false if no more movement could be made. This prevents this routine from moving way ahead after movement of the selected (in the user-interface) trigger stops.

Implements [seq64::perfroll](#).

Reimplemented in [seq64::FruityPerfInput](#).

10.82.2.3 check_handles()

```
bool seq64::Seq24PerfInput::check_handles ( ) [protected]
```

Set this flag to tell [on_motion_notify_event\(\)](#) to call [perf\(\)](#).push_trigger_undo(). This section handles motions of the held mouse that grow or shrink the selected trigger, or else the moving of the selected trigger.

`wscalex` is $4 * 32$ ticks/pixel. What about zoom? See `perfroll`; it sets `m_perf_scale_x` and `m_zoom` to `c_perf_scale_x`. Let's use `m_perf_scale_x` instead and make it a member, `m_w_scale_x`. This section is almost common code with the `fruityperfrol_input.cpp` module.

Check for a corner drag (on the small box at the left of the trigger segment) to "grow" the sequence start. Otherwise, check for a corner drag (on the small box at the right of the sequence) to "grow" the sequence end. Otherwise, we are moving the sequence.

Returns

Returns true if the mouse pointer is inside a trigger handle.

10.82.2.4 on_button_press_event()

```
bool seq64::Seq24PerfInput::on_button_press_event (
    GdkEventButton * ev ) [protected], [virtual]
```

This code causes the un-greying of the previously selected trigger segment. If commented out, then we can seemingly select more than one trigger segment, but only the last one "selected" can be moved. We'd like to be able to select and move a bunch at once by holding a modifier key.

[convert_drop_xy\(\)](#):

`convert_drop_xy()` uses `m_drop_x` and `m_drop_y`, and sets `m_drop_tick` and `m_drop_sequence`. It gets the sequence number of the pattern clicked. It doesn't matter which button was clicked.

[is_adding\(\)](#):

The `is_adding()` status is set on a right-click with no trigger under the mouse, and unset when the right-click is released.

Ctrl key:

Note the `is_ctrl_key()` test. We make better use of the Ctrl key here. Let Ctrl-left be handled exactly like the middle click (split the segment/trigger), then bug out. Middle click in seq24 mouse mode is either for splitting the triggers or for setting the paste location of copy/paste. The `"a_or_b_trigger()"` functions check the trigger status of the sequence to decide what to do, so that the caller doesn't have to do those checks.

Shift key:

TODO.

No modifier key:

Add a new sequence if nothing is selected. The adding flag is set on a right-click where there is no trigger under the mouse, and is unset when the right-click is released [`if not in allow_mod4_mode()`].

Mouse buttons:

The middle click in seq24 interaction mode is either for splitting the triggers or for setting the paste location of copy/paste.

Stazed:

`m_drop_y` will be adjusted by `perfroll::change_vert()` for any scroll after it was originally selected. The call here to `draw_drawable_row()` [now folded into `draw_all()`] will have the wrong `y` location and un-select will not occur (or the wrong sequence will be unselected) if the user scrolls the track up or down to a new `y` location, if not adjusted.

Box set selection:

This new setup is meant to support selecting multiple sequences in the `perfroll`, and later to support Kepler34's box-selection of multiple sequences for movement. Here are the rules we want to implement:

- Nothing selected.
 - Outside a trigger segment (determined by the `"intersect"` function.)
 - Inside a trigger segment.
- Click or Shift-Click with nothing already selected will select the current drop sequence and add it to the set.
- Click with at least one `*other*` sequence selected will unselect the others and select the current sequence.
- Click or Shift-Click outside a sequence will unselect them all.
- Shift-Click will leave the other selected sequences alone and add the current drop sequence.

Returns

Returns true if a modification occurred, necessitating a redraw.

Reimplemented from [seq64::perfroll](#).

Reimplemented in [seq64::FruityPerfInput](#).

10.82.2.5 on_button_release_event()

```
bool seq64::Seq24PerfInput::on_button_release_event (
    GdkEventButton * ev ) [protected], [virtual]
```

Any use for the middle-button or ctrl-left-click we can add?

Parameters

<i>ev</i>	The button event to be handles as a button-release.
-----------	---

Returns

Returns true if any modification occurred.

Reimplemented from [seq64::perftroll](#).

Reimplemented in [seq64::FruityPerfInput](#).

10.82.2.6 on_motion_notify_event()

```
bool seq64::Seq24PerfInput::on_motion_notify_event (
    GdkEventMotion * ev ) [protected], [virtual]
```

We now check to see if it is a drag-motion (left/middle/right-click-drag) to avoid flickering when just moving the mouse across the surface.

Parameters

<i>ev</i>	Provides a pointer to the motion event.
-----------	---

Returns

Returns true if a modification occurs. This function used to always return true.

Reimplemented from [seq64::perftroll](#).

Reimplemented in [seq64::FruityPerfInput](#).

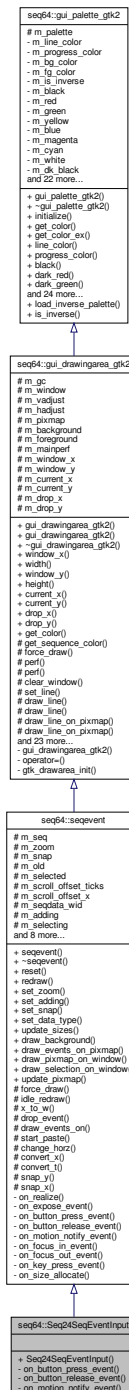
10.82.3 Field Documentation**10.82.3.1 m_effective_tick**

```
midipulse seq64::Seq24PerfInput::m_effective_tick [private]
```


10.83 seq64::Seq24SeqEventInput Class Reference

This structure implement the normal interaction methods for Seq24.

Inheritance diagram for seq64::Seq24SeqEventInput:



Public Member Functions

- [Seq24SeqEventInput](#) (perform &p, sequence &seq, int zoom, int snap, seqdata &seqdata_wid, Gtk::Adjustment &hadjust,)

Private Member Functions

- virtual bool [on_button_press_event](#) (GdkEventButton *ev)
Implements the on-button-press event callback.
- virtual bool [on_button_release_event](#) (GdkEventButton *ev)
Implements the on-button-release callback.
- virtual bool [on_motion_notify_event](#) (GdkEventMotion *ev)
Implements the on-motion-notify event.

Additional Inherited Members

10.83.1 Constructor & Destructor Documentation

10.83.1.1 Seq24SeqEventInput()

```
seq64::Seq24SeqEventInput::Seq24SeqEventInput (
    perform & p,
    sequence & seq,
    int zoom,
    int snap,
    seqdata & seqdata_wid,
    Gtk::Adjustment & hadjust )
```

10.83.2 Member Function Documentation

10.83.2.1 on_button_press_event()

```
bool seq64::Seq24SeqEventInput::on_button_press_event (
    GdkEventButton * ev ) [private], [virtual]
```

Set values for dragging, then reset the box that holds dirty redraw spot. Then do the rest.

Parameters

<code>ev</code>	The button event for the press of a mouse button.
-----------------	---

Returns

Returns true if a likely modification was made. This function used to return true all the time.

Reimplemented from [seq64::sequevent](#).

10.83.2.2 on_button_release_event()

```
bool seq64::Seq24SeqEventInput::on_button_release_event (
    GdkEventButton * ev ) [private], [virtual]
```

Parameters

ev	The button event for the release of a mouse button.
----	---

Returns

Returns true if a likely modification was made. This function used to return true all the time.

Reimplemented from [seq64::seqevent](#).

10.83.2.3 on_motion_notify_event()

```
bool seq64::Seq24SeqEventInput::on_motion_notify_event (
    GdkEventMotion * ev ) [private], [virtual]
```

Parameters

ev	The button event for the motion of the mouse cursor.
----	--

Returns

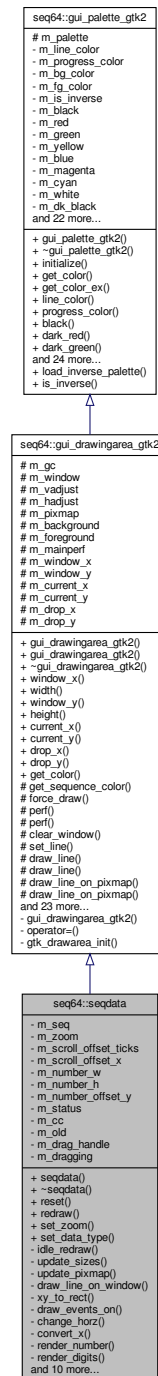
Returns true if a likely modification was made. This function used to return true all the time.

Reimplemented from [seq64::seqevent](#).

10.84 seq64::seqdata Class Reference

This class supports drawing piano-roll events on a window.

Inheritance diagram for seq64::seqdata:



Public Member Functions

- [seqdata](#) ([sequence](#) &seq, [perform](#) &p, int zoom, Gtk::Adjustment &hadjust)

Principal constructor.

- virtual [~seqdata](#) ()

Let's provide a do-nothing virtual destructor.

- void [reset](#) ()

- This function calls `update_size()`.*
- void `redraw` ()
Calls `change_horz()` to update the pixmap and queue up a redraw operation.
- void `set_zoom` (int a_zoom)
Sets the zoom to the given value and resets the view via the reset function.
- void `set_data_type` (midibyte status, midibyte control)
Sets the status to the given value, and the control to the optional given value, which defaults to 0, then calls `redraw()`.

Private Member Functions

- int `idle_redraw` ()
Draws events on this object's built-in window and pixmap.
- void `update_sizes` ()
Updates the sizes in the pixmap if the view is realized, and queues up a draw operation.
- void `update_pixmap` ()
Simply calls `draw_events_on_pixmap()`.
- void `draw_line_on_window` ()
Draws on vertical line on the data window.
- void `xy_to_rect` (int x1, int y1, int x2, int y2, int &rx, int &ry, int &rw, int &rh)
This function takes two points, and returns an XWin rectangle, returned via the last four parameters.
- void `draw_events_on` (Glib::RefPtr< Gdk::Drawable > drawable)
Draws events on the given drawable object.
- void `change_horz` ()
Change the scrolling offset on the x-axis, and redraw.
- void `convert_x` (int x, midipulse &tick)
This function takes screen coordinates, and gives the horizontal tick value based on the current zoom, returned via the second parameter.
- void `render_number` (Glib::RefPtr< Gdk::Pixmap > &pixmap, int x, int y, const char *const num)
Convenience function for rendering numbers.
- void `render_digits` (Glib::RefPtr< Gdk::Drawable > drawable, int digits, int x)
Renders an up to 3-digit string vertically to represent a data value.
- void `draw_events_on_pixmap` ()
Simply calls `draw_events_on()` for this object's built-in pixmap.
- void `draw_pixmap_on_window` ()
Simply queues up a draw operation.
- void `on_realize` ()
Implements the on-realization event, by calling the base-class version and then allocating the resources that could not be allocated in the constructor.
- bool `on_expose_event` (GdkEventExpose *ev)
Implements the on-expose event by calling `draw_drawable()` on the event.
- bool `on_button_press_event` (GdkEventButton *ev)
Implements a mouse button-press event.
- bool `on_button_release_event` (GdkEventButton *ev)
Implement a button-release event.
- bool `on_motion_notify_event` (GdkEventMotion *ev)
Handles a motion-notify event.
- bool `on_leave_notify_event` (GdkEventCrossing *ev)
Handles an on-leave notification event.
- bool `on_scroll_event` (GdkEventScroll *ev)
Implements the on-scroll event.
- void `on_size_allocate` (Gtk::Allocation &)
Handles a size-allocation event by updating `m_window_x` and `m_window_y`, and then updating all of the sizes of the data pane in `update_sizes()`.

Private Attributes

- [sequence](#) & [m_seq](#)
Points to the sequence whose data is being affected by this class.
- int [m_zoom](#)
Sets the zoom value for this part of the sequence editor, one pixel == m_zoom ticks, i.e.
- int [m_scroll_offset_ticks](#)
The value of the leftmost tick in the data pane.
- int [m_scroll_offset_x](#)
The value of the leftmost pixel in the data pane.
- int [m_number_w](#)
The adjusted width of a digit in a data number.
- int [m_number_h](#)
The adjusted height of all digits in a data number.
- int [m_number_offset_y](#)
A new value to make it easier to adapt the vertical number drawing of a data item's numeric value to a different font.
- midibyte [m_status](#)
Holds the status byte of the next event in the sequence, and indicates What the data window is currently editing or drawing.
- midibyte [m_cc](#)
Holds the MIDI CC byte of the next event in the sequence, and indicates What the data window is currently editing or drawing.
- GdkRectangle [m_old](#)
This rectangle is used in blanking out a data line in [draw_line_on_window\(\)](#).
- bool [m_drag_handle](#)
- bool [m_dragging](#)
This value is true if the mouse is being dragged in the data pane, which is done in order to change the height and value of each data line.

Friends

- class [lfownd](#)
- class [segevent](#)
- class [seqroll](#)

Additional Inherited Members

10.84.1 Constructor & Destructor Documentation

10.84.1.1 seqdata()

```
seq64::seqdata::seqdata (
    sequence & seq,
    perform & p,
    int zoom,
    Gtk::Adjustment & hadjust )
```

In the constructor one can only allocate colors, `get_window()` returns 0 because this pane has not yet been realized.

Parameters

<i>seq</i>	The sequence that is being displayed and edited by this data pane.
<i>p</i>	The performance object that oversees all of the sequences. This object is needed here only to access the perform::modify() function.
<i>zoom</i>	The starting zoom of this pane.
<i>hadjust</i>	The horizontal adjustment object provided by the parent class, seqedit , that created this pane.

10.84.1.2 ~seqdata()

```
virtual seq64::seqdata::~seqdata ( ) [inline], [virtual]
```

10.84.2 Member Function Documentation

10.84.2.1 reset()

```
void seq64::seqdata::reset ( )
```

Then, regardless of whether the view is realized, updates the pixmap and queues up a draw operation.

Note

If it weren't for the `is_realized()` condition, we could just call [update_sizes\(\)](#), which does all this anyway.

10.84.2.2 redraw()

```
void seq64::seqdata::redraw ( ) [inline]
```

10.84.2.3 set_zoom()

```
void seq64::seqdata::set_zoom (
    int z )
```

Called by [seqedit::set_zoom\(\)](#), which validates the zoom value.

Parameters

<i>z</i>	The desired zoom value, assumed to be validated already. See the seqedit::set_zoom() function.
----------	--

10.84.2.4 set_data_type()

```
void seq64::seqdata::set_data_type (
    midibyte status,
    midibyte control )
```

Perhaps we should check that at least one of the parameters causes a change.

Parameters

<i>status</i>	The MIDI event byte (status byte) to set.
<i>control</i>	The MIDI CC value to set.

10.84.2.5 idle_redraw()

```
int seq64::seqdata::idle_redraw ( ) [private]
```

This drawing is done only if there is no dragging in progress, to guarantee no flicker.

10.84.2.6 update_sizes()

```
void seq64::seqdata::update_sizes ( ) [private]
```

It creates a pixmap with window dimensions given by `m_window_x` and `m_window_y`.

We thought there was a potential memory leak, since `m_pixmap` is created every time the window is resized, but `valgrind` says otherwise... maybe. An awful lot of [Gtk](#) leaks!

10.84.2.7 update_pixmap()

```
void seq64::seqdata::update_pixmap ( ) [private]
```

10.84.2.8 draw_line_on_window()

```
void seq64::seqdata::draw_line_on_window ( ) [private]
```


10.84.2.9 xy_to_rect()

```
void seq64::seqdata::xy_to_rect (
    int x1,
    int y1,
    int x2,
    int y2,
    int & rx,
    int & ry,
    int & rw,
    int & rh ) [private]
```

It checks the mins/maxes, then fills in x, y, and width, height.

Parameters

	<i>x1</i>	The input x value for the first data point.
	<i>y1</i>	The input y value for the first data point.
	<i>x2</i>	The input x value for the second data point.
	<i>y2</i>	The input y value for the second data point.
out	<i>rx</i>	The output for the x value of the XWin rectangle.
out	<i>ry</i>	The output for the y value of the XWin rectangle.
out	<i>rw</i>	The output for the width value of the XWin rectangle.
out	<i>rh</i>	The output for the height of the XWin rectangle.

10.84.2.10 `draw_events_on()`

```
void seq64::seqdata::draw_events_on (
    Glib::RefPtr< Gdk::Drawable > drawable ) [private]
```

Very similar to `seqevent::draw_events_on()`. And yet it doesn't handle zooming as well, must fix!

Stazed:

For Note On there can be multiple events on the same vertical in which the selected item can be covered. For Note On the selected item needs to be drawn last so it can be seen. So, for other events the variable `num_selected_events` will be -1 for `ALL_EVENTS`. For Note On only, the variable will be the number of selected events. If 0 then only one pass is needed. If > 0 then two passes are needed, one for unselected (first), and one for selected (last). For the first pass, if any events are selected, the selection type is `EVENTS_UNSELECTED`. For the second pass, it will be set to `num_selected_events`.

We now draw the data line for selected event in dark orange, instead of black. We're not likely to adopt the Stazed convention of drawing in blue. Also, there seem to be some bugs in how the data selection works. Needs more evaluation.

Also, if we decide to draw handle on each vertical data line, it would look nicer if a circle.

Parameters

<i>drawable</i>	The given drawable object.
-----------------	----------------------------

10.84.2.11 `change_horz()`

```
void seq64::seqdata::change_horz ( ) [private]
```

Basically identical to `seqevent::change_horz()`.

10.84.2.12 convert_x()

```
void seq64::seqdata::convert_x (
    int x,
    midipulse & tick ) [inline], [private]
```

10.84.2.13 render_number()

```
void seq64::seqdata::render_number (
    Glib::RefPtr< Gdk::Pixmap > & pixmap,
    int x,
    int y,
    const char *const num ) [inline], [private]
```

Parameters

<i>pixmap</i>	The reference pointer to the GDK pixmap onto which this number will be drawing.
<i>x</i>	The x-coordinate of the position of the text.
<i>y</i>	The y-coordinate of the position of the text.
<i>num</i>	The number to be rendered. This should be a string reference, but oh well.

10.84.2.14 render_digits()

```
void seq64::seqdata::render_digits (
    Glib::RefPtr< Gdk::Drawable > drawable,
    int digits,
    int x ) [private]
```

Parameters

<i>drawable</i>	Where to draw the digits.
<i>digits</i>	xxxxxx
<i>x</i>	xxxxxx

10.84.2.15 draw_events_on_pixmap()

```
void seq64::seqdata::draw_events_on_pixmap ( ) [inline], [private]
```

10.84.2.16 draw_pixmap_on_window()

```
void seq64::seqdata::draw_pixmap_on_window ( ) [inline], [private]
```

10.84.2.17 on_realize()

```
void seq64::seqdata::on_realize ( ) [private]
```

It also connects up the [change_horz\(\)](#) function.

Note that this function creates a small pixmap for every possible y-value, where y ranges from 0 to MIDI_COUNT ← _MAX-1 = 127. It then fills each pixmap with a numeric representation of that y value, up to three digits (left-padded with spaces).

10.84.2.18 on_expose_event()

```
bool seq64::seqdata::on_expose_event (
    GdkEventExpose * ev ) [private]
```

Parameters

<i>ev</i>	Provides the expose-event.
-----------	----------------------------

Returns

Always returns true.

10.84.2.19 on_button_press_event()

```
bool seq64::seqdata::on_button_press_event (
    GdkEventButton * ev ) [private]
```

This function pushes the undo information for the sequence, sets the drop-point, resets the box that holds dirty redraw spot, and sets m_dragging to true.

Parameters

<i>ev</i>	Provides the button-press event.
-----------	----------------------------------

Returns

Always returns true.

10.84.2.20 on_button_release_event()

```
bool seq64::seqdata::on_button_release_event (
    GdkEventButton * ev ) [private]
```

Sets the current point. If m_dragging is true, then the sequence data is changed, the performance modification flag is set, and m_dragging is reset.

Parameters

<i>ev</i>	Provides the button-release event.
-----------	------------------------------------

Returns

Returns true if a modification occurred, and in that case also sets the perform modification flag.

10.84.2.21 on_motion_notify_event()

```
bool seq64::seqdata::on_motion_notify_event (
    GdkEventMotion * ev ) [private]
```

It converts the x,y of the mouse to ticks, then sets the events in the event-data-range, updates the pixmap, draws events in the window, and draws a line on the window.

Parameters

<i>ev</i>	The motion event.
-----------	-------------------

Returns

Returns true if a change in event data occurred. If true, then the perform modification flag is set.

10.84.2.22 on_leave_notify_event()

```
bool seq64::seqdata::on_leave_notify_event (
    GdkEventCrossing * ev ) [private]
```

Parameter "p0", the crossing point for the event, is unused.

10.84.2.23 on_scroll_event()

```
bool seq64::seqdata::on_scroll_event (
    GdkEventScroll * ev ) [private]
```

This scroll event only handles basic scrolling, without any modifier keys such as the Ctrl or Shift masks. If there is a note (seqroll pane) or event (sequevent pane) selected, and mouse hovers over the data area (seqdata pane), then this scrolling action will increase or decrease the value of the data item, which lengthens or shortens the line drawn.

Todo DOCUMENT the seqdata scrolling behavior in the documentation projects.

Parameters

<i>ev</i>	Provides the scroll-event.
-----------	----------------------------

Returns

Always returns true.

10.84.2.24 on_size_allocate()

```
void seq64::seqdata::on_size_allocate (
    Gtk::Allocation & r ) [private]
```

Parameters

<i>r</i>	Provides the allocation event.
----------	--------------------------------

10.84.3 Friends And Related Function Documentation**10.84.3.1 lfownd**

```
friend class lfownd [friend]
```

10.84.3.2 sequevent

```
friend class sequevent [friend]
```

10.84.3.3 seqroll

```
friend class seqroll [friend]
```

10.84.4 Field Documentation

10.84.4.1 m_seq

`sequence& seq64::seqdata::m_seq` [private]

10.84.4.2 m_zoom

`int seq64::seqdata::m_zoom` [private]

the unit is ticks/pixel.

10.84.4.3 m_scroll_offset_ticks

`int seq64::seqdata::m_scroll_offset_ticks` [private]

Adjusted in the [change_horz\(\)](#) function.

10.84.4.4 m_scroll_offset_x

`int seq64::seqdata::m_scroll_offset_x` [private]

Adjusted in the [change_horz\(\)](#) function. It is the offset ticks divided by the zoom value, i.e. the unit is pixels..

10.84.4.5 m_number_w

`int seq64::seqdata::m_number_w` [private]

By "adjusted", well this is just a minor tweak for appearances.

10.84.4.6 m_number_h

`int seq64::seqdata::m_number_h` [private]

Basically, the character height times 3. By "adjusted", well this is just a minor tweak for appearances.

10.84.4.7 m_number_offset_y

`int seq64::seqdata::m_number_offset_y` [private]

This value was hardwired as 8, for a character height of 10.

10.84.4.8 m_status

`midibyte seq64::seqdata::m_status` [private]

10.84.4.9 m_cc

```
midibyte seq64::seqdata::m_cc [private]
```

10.84.4.10 m_old

```
GdkRectangle seq64::seqdata::m_old [private]
```

10.84.4.11 m_drag_handle

```
bool seq64::seqdata::m_drag_handle [private]
```

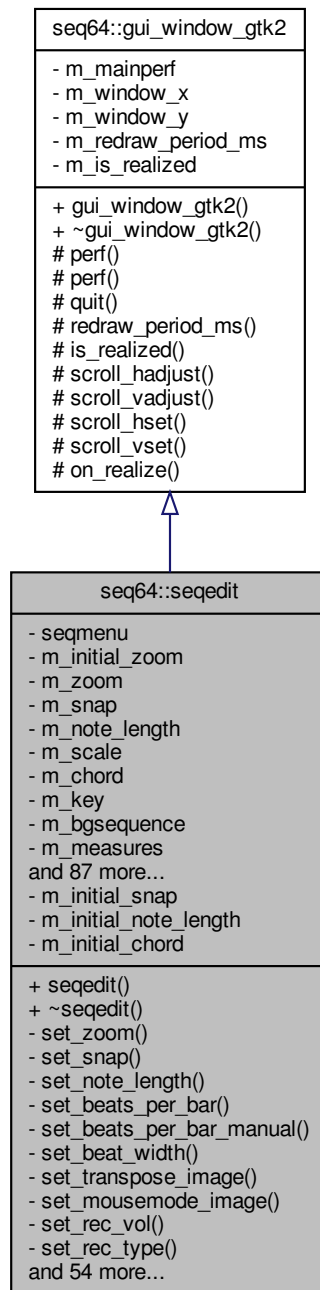
10.84.4.12 m_dragging

```
bool seq64::seqdata::m_dragging [private]
```

10.85 seq64::seqedit Class Reference

Implements the Pattern Editor, which has references to:

Inheritance diagram for seq64::seqedit:



Public Member Functions

- `seqedit` (`perform` &`perf`, `sequence` &`seq`, `int pos`, `int ppqn=SEQ64_USE_DEFAULT_PPQN`)
Principal constructor.
- virtual `~seqedit` ()
A rote destructor.

Private Member Functions

- void [set_zoom](#) (int zoom)
Selects the given zoom value.
- void [set_snap](#) (int snap)
Selects the given snap value, which is the number of ticks in a snap-sized interval.
- void [set_note_length](#) (int note_length)
Selects the given note-length value.
- void [set_beats_per_bar](#) (int bpm)
Set the bpm (beats per measure) value, using the given parameter, and some internal values passed to [apply_length\(\)](#).
- void [set_beats_per_bar_manual](#) ()
Set the bpm (beats per measure) value manually.
- void [set_beat_width](#) (int bw)
Set the bw (beat width) value, using the given parameter, and some internal values passed to [apply_length\(\)](#).
- void [set_transpose_image](#) (bool istransposable)
Changes the image used for the transpose button.
- void [set_mousemode_image](#) (bool isfruity)
Changes the image used for the mouse-mode indicator.
- void [set_rec_vol](#) (int recvol)
Passes the given parameter to [sequence::set_rec_vol\(\)](#).
- void [set_rec_type](#) (loop_record_t rectype)
- void [horizontal_adjust](#) (double step)
This function provides optimization for the [on_scroll_event\(\)](#) function.
- void [vertical_adjust](#) (double step)
This function provides optimization for the [on_scroll_event\(\)](#) function.
- void [horizontal_set](#) (double value)
Sets the exact position of a horizontal scroll-bar.
- void [vertical_set](#) (double value)
Sets the exact position of a vertical scroll-bar.
- int [get_measures](#) ()
Calculates the measures value based on the bpm (beats per measure), ppqn (parts per quarter note), and bw (beat width) values, and returns the resultant measures value.
- void [set_measures](#) (int lim)
Set the measures value, using the given parameter, and some internal values passed to [apply_length\(\)](#).
- void [set_measures_manual](#) ()
Set the measures value manually.
- void [apply_length](#) (int bpb, int bw, int measures)
Sets the sequence length based on the three given parameters.
- void [set_midi_channel](#) (int midichannel, bool user_change=false)
Selects the given MIDI channel parameter in the main sequence object, so that it will use that channel.
- void [set_midi_bus](#) (int midibus, bool user_change=false)
Selects the given MIDI buss parameter in the main sequence object, so that it will use that buss.
- void [set_scale](#) (int scale)
Selects the given scale value.
- void [set_chord](#) (int chord)
- void [set_key](#) (int note)
Selects the given key (signature) value.
- void [set_background_sequence](#) (int seq)
Draws the given background sequence on the Pattern editor so that the musician has something to see that can be played against.
- void [transpose_change_callback](#) ()

- Passes the transpose status to the sequence object.*

 - void [name_change_callback](#) ()

Set the name for the main sequence to this object's entry name.

- void [follow_change_callback](#) ()

Passes the Follow status to the seqroll object.

- void [play_change_callback](#) ()

Passes the play status to the sequence object.

- void [record_change_callback](#) ()

Passes the recording status to the sequence object.

- void [q_rec_change_callback](#) ()

Passes the quantized-recording status to the sequence object.

- void [thru_change_callback](#) ()

Passes the MIDI Thru status to the sequence object.

- void [undo_callback](#) ()

Pops an undo operation from the sequence object, and then tells the segroll, seqtime, seqdata, and sequevent objects to redraw.

- void [redo_callback](#) ()

Pops a redo operation from the sequence object, and then tell the segroll, seqtime, seqdata, and sequevent objects to redraw.

- void [update_all_windows](#) ()
- void [fill_top_bar](#) ()

This function inserts the user-interface items into the top bar or panel of the pattern editor; this bar has two rows of user interface elements.

- void [create_menus](#) ()

Creates the various menus by pushing menu elements into the menus.

- void [set_data_type](#) ([midibyte](#) status, [midibyte](#) control=0)

Sets the data type based on the given parameters.

- void [set_event_entry](#) (Gtk::Menu *menu, const std::string &text, bool present, [midibyte](#) status, [midibyte](#) control=0)

Function to create event menu entries.

- void [popup_menu](#) (Gtk::Menu *menu)

Pops up the given pop-up menu.

- void [popup_event_menu](#) ()

Populates the event-selection menu, in necessary, and then pops it up.

- void [repopulate_event_menu](#) (int buss, int channel)

Populates the event-selection menu that drops from the "Event" button in the bottom row of the Pattern editor.

- void [popup_mini_event_menu](#) ()

Populates the event-selection menu, in necessary, and then pops it up.

- void [repopulate_mini_event_menu](#) (int buss, int channel)

Populates the mini event-selection menu that drops from the mini-"Event" button in the bottom row of the Pattern editor.

- void [popup_record_menu](#) ()
- void [popup_midibus_menu](#) ()

Populates the MIDI Output buss pop-up menu.

- void [popup_sequence_menu](#) ()

Populates the "set background sequence" menu (drops from the button that has some note-bars on it at the right of the second row of the top bar).

- void [popup_tool_menu](#) ()

Sets up the pop-up menus that are brought up by pressing the Tools button, which shows a hammer image.

- void [popup_midich_menu](#) ()

Populates the MIDI Channel pop-up menu, if necessary, and then reveals the menu to the user.

- void [repopulate_midich_menu](#) (int buss)

- Populates the MIDI Channel pop-up menu.*

 - `Gtk::Image * create_menu_image (bool state=false)`

Sets the menu pixmap depending on the given state, where true is a full menu (black background), and empty menu (gray background).
 - `bool timeout ()`

Update the window after a time out, based on dirtiness and on playback progress.
 - `void do_action (int action, int var)`

Implements the actions brought forth from the Tools (hammer) button.
 - `void mouse_action (mouse_action_e action)`
 - `void start_playing ()`
 - `void stop_playing ()`
 - `void change_focus (bool set_it=true)`

Changes what perform and mainwid see as the "current sequence".
 - `void handle_close ()`

Handles closing the sequence editor.
 - `void on_realize ()`

On realization, calls the base-class version, and connects the redraw timeout signal, timed at `redraw_period_ms()`.
 - `void on_set_focus (Widget *focus)`

On receiving focus, attempt to tell mainwid that this sequence is now the current sequence.
 - `bool on_focus_in_event (GdkEventFocus *)`

Implements the on-focus event handling.
 - `bool on_focus_out_event (GdkEventFocus *)`

Implements the on-unfocus event handling.
 - `bool on_delete_event (GdkEventAny *event)`

Handles an on-delete event.
 - `bool on_scroll_event (GdkEventScroll *ev)`

Handles an on-scroll event.
 - `bool on_key_press_event (GdkEventKey *ev)`

Handles a key-press event.

Private Attributes

- friend `seqmenu`
- `const int m_initial_zoom`

Provides the initial zoom, used for restoring the original zoom using the 0 key.
- `int m_zoom`

Provides the zoom values: 1 2 3 4, and 1, 2, 4, 8, 16.
- `int m_snap`

Used in setting the snap-to value in pulses, off = 1.
- `int m_note_length`

The default length of a note to be inserted by a right-left-click operation.
- `int m_scale`

Setting for the music scale, can now be saved with the sequence.
- `int m_chord`

Setting for the current chord generation; not now saved with the sequence.
- `int m_key`

Setting for the music key, can now be saved with the sequence.
- `int m_bgsequence`

Setting for the background sequence, can now be saved with the sequence.
- `long m_measures`

- Provides the length of the sequence in measures.*

 - int [m_ppqn](#)

Holds a copy of the current PPQN for the sequence (and the entire MIDI file).
 - int [m_pp_whole](#)
 - int [m_pp_eighth](#)
 - int [m_pp_sixteenth](#)
 - [sequence](#) & [m_seq](#)

Holds a reference to the sequence that this window represents.
- Gtk::MenuBar * [m_menubar](#)

A number of user-interface objects for common.
- Gtk::Menu * [m_menu_tools](#)

The "hammer" tool button menu.
- Gtk::Menu * [m_menu_zoom](#)

Magnifying glass zoom menu.
- Gtk::Menu * [m_menu_snap](#)

Two-arrows grid-snap menu.
- Gtk::Menu * [m_menu_note_length](#)

Notes menu for note length.
- Gtk::Menu * [m_menu_length](#)

Pattern-length "bars" menu.
- Gtk::ToggleButton * [m_toggle_transpose](#)

Transpose toggle button.
- Gtk::Image * [m_image_transpose](#)

Image for transpose button.
- Gtk::Menu * [m_menu_midich](#)

MIDI channel DIN menu button.
- Gtk::Menu * [m_menu_midibus](#)

MIDI output buss menu button.
- Gtk::Menu * [m_menu_data](#)

"Event" button to select data.
- Gtk::Menu * [m_menu_minidata](#)

Mini button for actual events.
- Gtk::Menu * [m_menu_key](#)

"Music key" menu button.
- Gtk::Menu * [m_menu_scale](#)

"Music scale" menu button.
- Gtk::Menu * [m_menu_chords](#)

"Chords" menu button.
- Gtk::Menu * [m_menu_sequences](#)

"Background sequence" button.
- Gtk::Menu * [m_menu_bpm](#)

Beats/measure numerator menu.
- Gtk::Menu * [m_menu_bw](#)

Beat-width denominator menu.
- Gtk::Menu * [m_menu_rec_vol](#)

Recording level "Vol" button.
- Gtk::Menu * [m_menu_rec_type](#)

Recording type menu.
- Gtk::Adjustment * [m_vadjust](#)

Scrollbar and adjustment objects for horizontal and vertical panning.
- Gtk::Adjustment * [m_hadjust](#)

- Horizontal motion scratchpad.*

 - `Gtk::VScrollbar * m_vscroll_new`
- Main vertical scroll-bar.*

 - `Gtk::HScrollbar * m_hscroll_new`
- Main horizontal scroll-bar.*

 - `seqkeys * m_seqkeys_wid`
- Handles the piano-keys part of the pattern-editor user-interface.*

 - `seqtime * m_seqtime_wid`
- Handles the time-line (bar or measures) part of the pattern-editor user-interface.*

 - `seqdata * m_seqdata_wid`
- Handles the event-data part of the pattern-editor user-interface.*

 - `sequevent * m_sequevent_wid`
- Handles the small event part of the pattern-editor user-interface, where events can be moved and added.*

 - `seqroll * m_seqroll_wid`
- Handles the piano-roll part of the pattern-editor user-interface.*

 - `Gtk::Button * m_button_lfo`
- The LFO button in the pattern editor.*

 - `lfownd * m_lfo_wnd`
- The LFO window object used by the pattern editor.*

 - `Gtk::Table * m_table`
- More user-interface elements.*

 - `Gtk::VBox * m_vbox`
- Layout box for 3 h-boxes.*

 - `Gtk::HBox * m_hbox`
- Topmost menu/text dialog row.*

 - `Gtk::HBox * m_hbox2`
- Second row of buttons.*

 - `Gtk::Button * m_button_undo`
- Undo-edit button.*

 - `Gtk::Button * m_button_redo`
- Redo-edit button.*

 - `Gtk::Button * m_button_quantize`
- Quantize-pattern button.*

 - `Gtk::Button * m_button_tools`
- Button for the Tools menu.*

 - `Gtk::Button * m_button_sequence`
- Button for Background pattern.*

 - `Gtk::Entry * m_entry_sequence`
- Text for background pattern.*

 - `Gtk::Button * m_button_bus`
- Button for MIDI Buss menu.*

 - `Gtk::Entry * m_entry_bus`
- Text showing MIDI Buss name.*

 - `Gtk::Button * m_button_channel`
- Button for the MIDI Channel.*

 - `Gtk::Entry * m_entry_channel`
- Text for the MIDI Channel.*

 - `Gtk::Button * m_button_snap`
- Button for the Grid-snap menu.*

 - `Gtk::Entry * m_entry_snap`
- Text for selected Grid-snap.*

- Gtk::Button * [m_button_note_length](#)
Button for Note-length menu.
- Gtk::Entry * [m_entry_note_length](#)
Text showing the Note-length.
- Gtk::Button * [m_button_zoom](#)
Button for the Zoom menu.
- Gtk::Entry * [m_entry_zoom](#)
Text for the selected Zoom.
- Gtk::Button * [m_button_length](#)
Button for pattern-length.
- Gtk::Entry * [m_entry_length](#)
Text for the pattern-length.
- Gtk::Button * [m_button_key](#)
Button for the Music Key.
- Gtk::Entry * [m_entry_key](#)
Text for selected Music Key.
- Gtk::Button * [m_button_scale](#)
Button for the Music Scale.
- Gtk::Entry * [m_entry_scale](#)
Text for the Music Scale.
- Gtk::Button * [m_button_chord](#)
Button for the current Chord.
- Gtk::Entry * [m_entry_chord](#)
Text for the current Chord.
- Gtk::Tooltips * [m_tooltips](#)
Tooltip collector for dialog.
- Gtk::Button * [m_button_data](#)
Button for Event (data) menu.
- Gtk::Button * [m_button_minidata](#)
Mini button for data menu.
- Gtk::Entry * [m_entry_data](#)
Text for the selected Event.
- Gtk::Button * [m_button_bpm](#)
Button for Beats/Measure menu.
- Gtk::Entry * [m_entry_bpm](#)
Text for chosen Beats/Measure.
- Gtk::Button * [m_button_bw](#)
Button for Beat-Width menu.
- Gtk::Entry * [m_entry_bw](#)
Text for chosen Beat-Width.
- Gtk::Button * [m_button_rec_vol](#)
Button for recording volume.
- Gtk::Button * [m_button_rec_type](#)
Button for recording type.
- Gtk::ToggleButton * [m_toggle_follow](#)
Follow progress bar button.
- Gtk::ToggleButton * [m_toggle_play](#)
Pattern-to-MIDI record button.
- Gtk::ToggleButton * [m_toggle_record](#)
MIDI-port-to-pattern button.
- Gtk::ToggleButton * [m_toggle_q_rec](#)

- Quantized-record MIDI button.*
- Gtk::ToggleButton * [m_toggle_thru](#)
 - MIDI-to-pattern-MIDI button.*
- Gtk::Entry * [m_entry_seqnumber](#)
 - Number of the sequence.*
- Gtk::Entry * [m_entry_name](#)
 - Name of the sequence.*
- Gtk::Image * [m_image_mousemode](#)
 - Image for mouse mode button.*
- [midibyte m_editing_status](#)
 - Indicates what MIDI event/status the data window currently editing.*
- [midibyte m_editing_cc](#)
 - Indicates what MIDI CC value the data window currently editing.*
- [midibyte m_first_event](#)
 - Indicates the first event found in the sequence while setting up the data menu via [set_event_entry\(\)](#).*
- std::string [m_first_event_name](#)
 - Provides the string describing the first event, or "(no events)".*
- bool [m_have_focus](#)
 - Indicates that the focus has already been changed to this sequence.*

Static Private Attributes

- static int [m_initial_snap](#)
 - Static data members.*
- static int [m_initial_note_length](#)
- static int [m_initial_chord](#)

Additional Inherited Members

10.85.1 Detailed Description

- perform
- seqroll
- seqkeys
- seqdata
- seqtime
- seqevent
- sequence

This class has a metric ton of user-interface objects and other members.

10.85.2 Constructor & Destructor Documentation

10.85.2.1 seqedit()

```
seq64::seqedit::seqedit (
    perform & p,
    sequence & seq,
    int pos,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

If provided, override the scale, key, and background-sequence with the values stored in the file with the sequence, if they are set to non-default values. This is a new feature.

Todo Offload most of the work into an initialization function like options does.

Horizontal Gtk::Adjustment constructor: The initial value was 0 on a range from 0 to 1, with step and page increments of 1, and a page_size of 1. We can fix these values here, or create an h_adjustment() function similar to [eventedit::v_adjustment\(\)](#), which first gets called in [on_realize\(\)](#).

Parameters

<i>p</i>	The performance object of which the sequence is a part.
<i>seq</i>	The sequence object this window object represents.
<i>pos</i>	The sequence number (pattern slot number) for this sequence and window.
<i>ppqn</i>	The optional PPQN parameter for this sequence. Warning: not really used by the caller, need to square that!

10.85.2.2 ~seqedit()

```
seq64::seqedit::~seqedit ( ) [virtual]
```

10.85.3 Member Function Documentation

10.85.3.1 set_zoom()

```
void seq64::seqedit::set_zoom (
    int z ) [private]
```

It is passed to the seqroll, seqtime, seqdata, and sequevent objects, as well. This function doesn't check if the zoom will change, because this function might be used to initialize the zoom of the children.

The notation for zoom in the user-interface is in pixels:ticks, but I would prefer to use pulses/pixel (pulses per pixel). Oh well. Note that this value of zoom is saved to the "user" configuration file when Sequencer64 exit.

Parameters

<i>z</i>	The prospective zoom value to set. It is applied only if between the minimum and maximum allowed zoom values, inclusive. See the usr().min_zoom() and usr().max_zoom() function.
----------	--

10.85.3.2 `set_snap()`

```
void seq64::seqedit::set_snap (
    int s ) [private]
```

It is passed to the seqroll, sequevent, and sequence objects, as well.

The default initial snap is the default PPQN divided by 4, or the equivalent of a 16th note (48 ticks). The snap divisor is $192 * 4 / 48$ or 16.

Parameters

<i>s</i>	The prospective snap value to set. It is checked only to make sure it is greater than 0, to avoid a numeric exception.
----------	--

10.85.3.3 `set_note_length()`

```
void seq64::seqedit::set_note_length (
    int notelength ) [private]
```

It is passed to the seqroll object, as well.

Warning

Currently, we don't handle changes in the global PPQN after the creation of the menu. The creation of the menu hard-wires the values of note-length. To adjust for a new global PQN, we will need to store the original PPQN (`m_original_ppqn = m_ppqn`), and then adjust the notelength based on the new PPQN. For example if the new PPQN is twice as high as 192, then the notelength should double, though the text displayed in the "Note length" field should remain the same. However, we do adjust for a non-default PPQN at startup time.

Parameters

<i>notelength</i>	Provides the note length in units of MIDI pulses.
-------------------	---

10.85.3.4 `set_beats_per_bar()`

```
void seq64::seqedit::set_beats_per_bar (
    int bpb ) [private]
```

Todo Check if verification is needed at this point.

Parameters

<i>bpb</i>	Provides the BPM (beats per measure) value to set. Not beats/minute!
------------	--

10.85.3.5 set_beats_per_bar_manual()

```
void seq64::seqedit::set_beats_per_bar_manual ( ) [private]
```

For issue #77, pulled from the jean-emmanuel-manul-bpm-and-measure branch. The setting is limited to 0 to 128.

10.85.3.6 set_beat_width()

```
void seq64::seqedit::set_beat_width (
    int bw ) [private]
```

Todo Check if verification is needed at this point.

Parameters

<i>bw</i>	Provides the beat-width value to set.
-----------	---------------------------------------

10.85.3.7 set_transpose_image()

```
void seq64::seqedit::set_transpose_image (
    bool istransposable ) [private]
```

Can we leverage this variation?

```
m_toggle_transpose->add_pixlabel("info.xpm", "duh");
```

Parameters

<i>istransposable</i>	If true, set the image to the "Transpose" icon. Otherwise, set it to the "Drum" (not transposable) icon.
-----------------------	--

10.85.3.8 `set_mousemode_image()`

```
void seq64::segedit::set_mousemode_image (
    bool isfruity ) [private]
```

Parameters

<i>isfruity</i>	If true, set the image to the "Fruity" icon. Otherwise, set it to the "Tux" icon.
-----------------	---

10.85.3.9 `set_rec_vol()`

```
void seq64::segedit::set_rec_vol (
    int recvol ) [private]
```

This function also changes the button's text to match the selection, and also changes the global velocity-override setting in [user_settings](#). Note that the setting will not be saved to the "usr" configuration file unless Sequencer64 was run with the "--user-save" option.

Parameters

<i>recvol</i>	The setting to be made, obtained from the recording-volume ("Vol") menu.
---------------	--

10.85.3.10 `set_rec_type()`

```
void seq64::segedit::set_rec_type (
    loop_record_t rectype ) [private]
```

10.85.3.11 `horizontal_adjust()`

```
void seq64::segedit::horizontal_adjust (
    double step ) [inline], [private]
```

A duplicate of the one in `seqroll`.

Parameters

<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. See gui_drawingarea_gtk2::scroll_hadjust() for more information.
-------------	---

10.85.3.12 vertical_adjust()

```
void seq64::seqedit::vertical_adjust (
    double step ) [inline], [private]
```

A near-duplicate of the one in seqroll.

Parameters

<i>step</i>	Provides the step value to use for adjusting the vertical scrollbar. See gui_drawingarea_gtk2::scroll_vadjust() for more information.
-------------	---

10.85.3.13 horizontal_set()

```
void seq64::seqedit::horizontal_set (
    double value ) [inline], [private]
```

Parameters

<i>value</i>	The desired position. Mostly this is either 0.0 or 9999999.0 (an "infinite" value to select the start or end position).
--------------	---

10.85.3.14 vertical_set()

```
void seq64::seqedit::vertical_set (
    double value ) [inline], [private]
```

Parameters

<i>value</i>	The desired position. Mostly this is either 0.0 or 9999999.0 (an "infinite" value to select the start or end position).
--------------	---

10.85.3.15 get_measures()

```
int seq64::seqedit::get_measures ( ) [private]
```

Todo Create a `sequence::set_units()` function or a `sequence::get_measures()` function to forward to.

10.85.3.16 set_measures()

```
void seq64::seqedit::set_measures (
    int len ) [private]
```

Todo Check if verification is needed at this point.

Parameters

<i>len</i>	Provides the sequence length, in measures.
------------	--

10.85.3.17 set_measures_manual()

```
void seq64::seqedit::set_measures_manual ( ) [private]
```

For issue #77, pulled from the jean-emmanual-manul-bpm-and-measure branch. The setting is limited to 0 to 1024.

10.85.3.18 apply_length()

```
void seq64::seqedit::apply_length (
    int bpb,
    int bw,
    int measures ) [private]
```

There's an implicit "adjust-triggers = true" parameter used in [sequence::set_length\(\)](#). Then the seqroll, seqtime, seqdata, and seqevent objects are reset(), to cause redraw operations.

Parameters

<i>bpb</i>	Provides the beats per bar.
<i>bw</i>	Provides the beatwidth (typically 4) from the time signature.
<i>measures</i>	Provides the number of measures the sequence should cover, obtained from the user-interface.

10.85.3.19 set_midi_channel()

```
void seq64::seqedit::set_midi_channel (
    int midichannel,
    bool user_change = false ) [private]
```

Should this change set the is-modified flag? Where should validation occur?

Parameters

<i>midichannel</i>	The MIDI channel value to set.
<i>user_change</i>	True if the user made this change, and thus has potentially modified the song.

10.85.3.20 `set_midi_bus()`

```
void seq64::seqedit::set_midi_bus (
    int bus,
    bool user_change = false ) [private]
```

Should this change set the is-modified flag? Where should validation against the ALSA or JACK buss limits occur?

Also, it would be nice to be able to update this display of the MIDI bus in the field if we set it from the seqmenu.

Parameters

<i>bus</i>	The buss value to set. If this value changes the selected buss, then the MIDI channel popup menu is repopulated.
<i>user_change</i>	True if the user made this change, and thus has potentially modified the song.

10.85.3.21 `set_scale()`

```
void seq64::seqedit::set_scale (
    int scale ) [private]
```

It is passed to the seqroll and seqkeys objects, as well. As a new feature, it is also passed to the sequence, so that it can be saved as part of the sequence data.

Note that the "initial value" for this parameter is a static variable that gets set to the new value, so that opening up another sequence causes the sequence to take on the new "initial value" as well. A feature, but should it be optional? Now it is, based on the setting of `usr().global_seq_feature()`.

10.85.3.22 `set_chord()`

```
void seq64::seqedit::set_chord (
    int chord ) [private]
```

10.85.3.23 set_key()

```
void seq64::seqedit::set_key (
    int key ) [private]
```

It is passed to the seqroll and seqkeys objects, as well. As a new feature, it is also passed to the sequence, so that it can be saved as part of the sequence data.

Note that the "initial value" for this parameter is a static variable that gets set to the new value, so that opening up another sequence causes the sequence to take on the new "initial value" as well. A feature, but should it be optional? Now it is, based on the setting of [usr\(\).global_seq_feature\(\)](#).

10.85.3.24 set_background_sequence()

```
void seq64::seqedit::set_background_sequence (
    int seqnum ) [private]
```

As a new feature, it is also passed to the sequence, so that it can be saved as part of the sequence data, but only if less or equal to the maximum single-byte MIDI value, 127.

Note that the "initial value" for this parameter is a static variable that gets set to the new value, so that opening up another sequence causes the sequence to take on the new "initial value" as well. A feature, but should it be optional? Now it is, based on the setting of [usr\(\).global_seq_feature\(\)](#).

10.85.3.25 transpose_change_callback()

```
void seq64::seqedit::transpose_change_callback ( ) [private]
```

10.85.3.26 name_change_callback()

```
void seq64::seqedit::name_change_callback ( ) [private]
```

That name is the name the user has given to the sequence being edited.

10.85.3.27 follow_change_callback()

```
void seq64::seqedit::follow_change_callback ( ) [private]
```

10.85.3.28 play_change_callback()

```
void seq64::seqedit::play_change_callback ( ) [private]
```


10.85.3.29 record_change_callback()

```
void seq64::seqedit::record_change_callback ( ) [private]
```

Stazed:

Both `record_change_callback()` and `thru_change_callback()` will call `set_sequence_input()` for the same sequence. We only need to call it if it is not already set, if setting. And, we should not unset it if the `m_toggle_thru->get_active()` is true.

10.85.3.30 q_rec_change_callback()

```
void seq64::seqedit::q_rec_change_callback ( ) [private]
```

Stazed fix:

If we set Quantized recording, then also set recording, but do not unset recording if we unset Quantized recording.

This is not necessarily the most intuitive thing to do. See `midi_record.txt`.

10.85.3.31 thru_change_callback()

```
void seq64::seqedit::thru_change_callback ( ) [private]
```

Stazed:

Both `record_change_callback()` and `thru_change_callback()` will call `set_sequence_input()` for the same sequence. We only need to call it if it is not already set, if setting. And, we should not unset it if the `m_toggle_thru->get_active()` is true.

10.85.3.32 undo_callback()

```
void seq64::seqedit::undo_callback ( ) [private]
```

10.85.3.33 redo_callback()

```
void seq64::seqedit::redo_callback ( ) [private]
```

10.85.3.34 update_all_windows()

```
void seq64::seqedit::update_all_windows ( ) [private]
```

10.85.3.35 fill_top_bar()

```
void seq64::seqedit::fill_top_bar ( ) [private]
```

Note that, if a non-default title for the sequence is in force, then we immediately force the focus to the seqroll "widget", so that the space bar can be used to control playback, instead of immediately erasing the name of the sequence. The following commented radio-buttons were a visual way to select the modes of note editing (select, draw, and grow). These can easily be done with the left mouse button, keystrokes, or some other tricks, though.

10.85.3.36 create_menus()

```
void seq64::seqedit::create_menus ( ) [private]
```

The first menu is the Zoom menu, represented in the pattern/sequence editor by a button with a magnifying glass. The values are "pixels to ticks", where "ticks" are actually the "pulses" of "pulses per quarter note". We would prefer the notation "n" instead of "1:n", as in "n pulses per pixel".

Note that many of the setups here could be loops through data structures. The Snap menu is actually the Grid Snap button, which shows two arrows pointing to a central bar. This menu somewhat duplicates the same menu in perfedit.

To reduce the amount of written code, we now use a static array to initialize some of the seqedit menu entries. 0 denotes the separator. This same setup is used to set up both the snap and note menu, since they are exactly the same. Saves a *lot* of code.

This menu lets one set the key of the sequence, and is brought up by the button with the "golden key" image on it.

This button shows a down around for the bottom half of the time signature. It's tooltip is "Time signature. Length of beat." But it is called bw, or beat width, in the code.

This menu is shown when pressing the button at the bottom of the window that has "Vol" as its label. Let's show the numbers as well to help the user. And we'll have to document this change.

This menu sets the scale to show on the panel, and the button shows a "staircase" image. See the `c_music_scales` enumeration defined in the `globals` module.

This section sets up two different menus. The first is `m_menu_length`. This menu lets one set the sequence length in bars. The second menu is the `m_menu_bpm`, or BPM, which here means "beats per measure" (not "beats per minute").

10.85.3.37 set_data_type()

```
void seq64::seqedit::set_data_type (
    midibyte status,
    midibyte control = 0 ) [private]
```

This function uses the hardwired array `c_controller_names`.

Parameters

<i>status</i>	The current editing status.
<i>control</i>	The control value. However, we really need to validate it!

10.85.3.38 set_event_entry()

```
void seq64::seqedit::set_event_entry (
    Gtk::Menu * menu,
    const std::string & text,
    bool present,
    midibyte status,
    midibyte control = 0 ) [private]
```

Too damn big!

10.85.3.39 popup_menu()

```
void seq64::seqedit::popup_menu (
    Gtk::Menu * menu ) [private]
```

At construction, this function is set as the signal handler for each Gtk::Menu object, and that function is bound with the Menu object as a parameter.

10.85.3.40 popup_event_menu()

```
void seq64::seqedit::popup_event_menu ( ) [private]
```

Also see the handling of the m_button_data and m_entry_data objects.

10.85.3.41 repopulate_event_menu()

```
void seq64::seqedit::repopulate_event_menu (
    int buss,
    int channel ) [private]
```

This menu has a large number of items. They are filled in by code, but can also be loaded from sequencer64.usr.

This function first loops through all of the existing events in the sequence in order to determine what events exist in it. If any of the following events are found, their entry in the menu is marked by a filled square, rather than a hollow square:

- Note On
- Note off
- Aftertouch
- Program Change
- Channel Pressure
- Pitch Wheel
- Control Changes from 0 to 127

Parameters

<i>buss</i>	The selected bus number.
<i>channel</i>	The selected channel number.

Create the 8 sub-menus for the various ranges of controller changes, shown 16 per sub-menu.

10.85.3.42 popup_mini_event_menu()

```
void seq64::seqedit::popup_mini_event_menu ( ) [private]
```

Also see the handling of the `m_button_minidata` and `m_entry_data` objects.

10.85.3.43 repopulate_mini_event_menu()

```
void seq64::seqedit::repopulate_mini_event_menu (
    int buss,
    int channel ) [private]
```

This menu has a much smaller number of items, only the ones that actually exist in the track/pattern/loop/sequence.

Parameters

<i>buss</i>	The selected bus number.
<i>channel</i>	The selected channel number.

Create the one menu for the controller changes that actually exist in the track, if any.

10.85.3.44 popup_record_menu()

```
void seq64::seqedit::popup_record_menu ( ) [private]
```

10.85.3.45 popup_midibus_menu()

```
void seq64::seqedit::popup_midibus_menu ( ) [private]
```

The MIDI busses are obtained by getting the `mastermidibus` object, and iterating through the busses that it contains.

However, JACK counts the playback ports, such as "yoshimi:midi in", as "input" ports... the application outputs to the input ports. So we have to deal with that somehow.

10.85.3.46 popup_sequence_menu()

```
void seq64::seqedit::popup_sequence_menu ( ) [private]
```

It is populated with an "Off" menu entry, and a second "[0]" menu entry that pulls up a drop-down menu of all of the patterns/sequences that are present in the MIDI file for screen-set 0. If more screensets have active sequences, then their screen-set number appears in the screen-set section of the menu.

Now, at present, we can only save background sequence numbers that are less than 128, which means the sequences from 0 to 127, or the first four screen sets. Higher sequences can be selected, but, right now, they cannot be saved. We'll probably fix that at some point, low priority.

10.85.3.47 popup_tool_menu()

```
void seq64::seqedit::popup_tool_menu ( ) [private]
```

This button shows three sub-menus that need to be filled in by this function. All the functions accessed here seem to be implemented by the [do_action\(\)](#) function.

10.85.3.48 popup_midich_menu()

```
void seq64::seqedit::popup_midich_menu ( ) [private]
```

10.85.3.49 repopulate_midich_menu()

```
void seq64::seqedit::repopulate_midich_menu (
    int buss ) [private]
```

This action is needed at startup of the seqedit window, and when the user changes the active buss for the sequence.

When the output buss or channel are changed, we get the 16 "channels" from the new buss's definition, get the corresponding instrument, and load its name into this midich popup. Then we need to go to the instrument/channel that has been selected, and repopulate the event menu with that item's controller values/names.

Parameters

<i>buss</i>	The new value for the buss from which to get the [user-instrument-N] settings in the [user-instrument-definitions] section.
-------------	---

10.85.3.50 create_menu_image()

```
Gtk::Image * seq64::seqedit::create_menu_image (
    bool state = false ) [private]
```

10.85.3.51 timeout()

```
bool seq64::seqedit::timeout ( ) [private]
```

Note the new call to [seqroll::follow_progress\(\)](#). This allows the seqroll to pop to the next frame of events to continue to show the moving progress bar. Does this need to be an option? It only affects patterns longer than a measure or two, whatever the width of the seqroll window is. This is a new feature that is not in seq24.

What about seqtime? That doesn't change.

10.85.3.52 do_action()

```
void seq64::seqedit::do_action (
    int action,
    int var ) [private]
```

Note that the push_undo() calls push all of the current events (in [sequence::m_events](#)) onto the stack (as a single entry).

10.85.3.53 mouse_action()

```
void seq64::seqedit::mouse_action (
    mouse_action_e action ) [private]
```

10.85.3.54 start_playing()

```
void seq64::seqedit::start_playing ( ) [private]
```

10.85.3.55 stop_playing()

```
void seq64::seqedit::stop_playing ( ) [private]
```

10.85.3.56 change_focus()

```
void seq64::seqedit::change_focus (
    bool set_it = true ) [private]
```

Similar to the same function in eventedit.

Parameters

<code>set↵ _it</code>	If true (the default value), indicates we want focus, otherwise we want to give up focus.
---------------------------	---

10.85.3.57 `handle_close()`

```
void seq64::segedit::handle_close ( ) [private]
```

10.85.3.58 `on_realize()`

```
void seq64::segedit::on_realize ( ) [private]
```

10.85.3.59 `on_set_focus()`

```
void seq64::segedit::on_set_focus (
    Widget * focus ) [private]
```

Only works in certain circumstances.

10.85.3.60 `on_focus_in_event()`

```
bool seq64::segedit::on_focus_in_event (
    GdkEventFocus * ) [private]
```

10.85.3.61 `on_focus_out_event()`

```
bool seq64::segedit::on_focus_out_event (
    GdkEventFocus * ) [private]
```

10.85.3.62 on_delete_event()

```
bool seq64::seqedit::on_delete_event (
    GdkEventAny * event ) [private]
```

It tells the sequence to stop recording, tells the perform object's mastermidibus to stop processing input, and sets the sequence object's editing flag to false.

Warning

This function also calls "delete this"!

Returns

Always returns false.

10.85.3.63 on_scroll_event()

```
bool seq64::seqedit::on_scroll_event (
    GdkEventScroll * ev ) [private]
```

This handles moving the scroll wheel on a mouse or do a two-fingered scrolling action on a touchpad. If no modifier key is pressed, this moves the view up or down on the "notes" coordinate, showing different piano keys. This behavior is implemented in [seqkeys::on_scroll_event\(\)](#), and is called into play by returning false here.

If the Ctrl key is pressed, then the scrolling action causes the view to zoom in or out. This behavior is implemented here.

If the Shift key is pressed, then the scrolling action moves the view horizontally on the time-line (measures-line) of the piano roll. This behavior is implemented here.

10.85.3.64 on_key_press_event()

```
bool seq64::seqedit::on_key_press_event (
    GdkEventKey * ev ) [private]
```

A number of new keystrokes are processed, so that we can lessen the reliance on the mouse and work a little faster.

- Ctrl-W keypress. This keypress closes the sequence/pattern editor window by way of calling `on_delete_event()`. We could apply this convention to all the other windows.
- z 0 Z zoom keys. "z" zooms out, "Z" (Shift-z) zooms in, and "0" resets the zoom to the default.
- Page-Up and Page-Down. Moves up and down in the piano roll.
- Home and End. Page to the top or the bottom of the piano roll.
- Shift-Page-Up and Shift-Page-Down. Move left and right in the piano roll.
- Shift-Home and Shift-End. Page to the start or the end of the piano roll.
- Ctrl-Page-Up and Ctrl-Page-Down. Mirrors the zoom-in and zoom-out capabilities of scrolling up and down with the mouse while the Ctrl key is pressed.

The Keypad-End key is an issue on our ASUS "gaming" laptop. Whether it is seen as a "1" or an "End" key depends on an interaction between the Shift and the Num Lock key. Annoying, takes some time to get used to.

Change Note layk 2016-10-17 Issue #46. Undoing (ctrl-z) removes two instances of history. To reproduce this bug, if one makes three notes one at a time and presses ctrl-z once only the first one remains. Same goes for moving notes. This is due to this else-if statement where we call `seqroll::on_key_press_event()` making first removal. This if statement is never true and `seqroll::on_key_press_event()` is called again as `Gtk::Window::on_key_press_event()`, making another `m_seq.pop_undo()` in `seqroll`. Note that the code here was an (ill-advised) attempt to avoid the pattern title field from grabbing the initial keystrokes; better to just get used to clicking the piano roll first. Finally, fixing the undo bug also let's ctrl-page-up/page-down change the zoom. Lastly, we've removed the undo here... `seqroll` already handles both undo and redo keystrokes.

Change Note ca 2016-10-18 Issue #46. In addition to layk's fixes, we have to properly determine if we're inside the "Sequence Name" ("GtkEntry") field, as opposed to the "GtkDrawingArea" field, to avoid grabbing and using keystrokes intended for the text-entry field. We may have to rethink the whole `seqroll` vs. `seqedit` key-press process at some point, as this is a bit too tricky. Please note that the name "gtkmm__GtkEntry" likely applies only to GNU's C++ compiler, g++. This will be an issue in any port to Microsoft's C++ compiler.

Parameters

ev	Provides the keystroke event to be handled.
----	---

Returns

Returns true if we handled the keystroke here. Otherwise, returns the value of `Gtk::Window::on_key_press_event(ev)`.

10.85.4 Field Documentation

10.85.4.1 seqmenu

```
friend seq64::seqedit::seqmenu [private]
```

10.85.4.2 m_initial_snap

```
int seq64::seqedit::m_initial_snap [static], [private]
```

These items apply to all of the instances of `seqedit`, and are passed on to the following constructors:

- `seqdata`
- `seqevent`
- `seqroll`
- `seqtime`

The snap and note-length defaults would be good to write to the "user" configuration file. The scale and key would be nice to write to the proprietary section of the MIDI song. Or, even more flexibly, to each sequence, if that makes sense to do, since all tracks would generally be in the same key. Right, Charles Ives?

Note that, currently, that some of these "initial values" are modified, so that they are "contagious". That is, the next sequence to be opened in the sequence editor will adopt these values. This is a long-standing feature of `Seq24`, but strikes us as a bit surprising.

Change Note ca 2016-04-10 If we just double the PPQN, then the snap divisor becomes 32, and the snap interval is a 32nd note. We would like to keep it at a 16th note. We correct the snap ticks to the actual PPQN ratio.

10.85.4.3 m_initial_note_length

```
int seq64::seqedit::m_initial_note_length [static], [private]
```

10.85.4.4 m_initial_chord

```
int seq64::seqedit::m_initial_chord [static], [private]
```

10.85.4.5 m_initial_zoom

```
const int seq64::seqedit::m_initial_zoom [private]
```

10.85.4.6 m_zoom

```
int seq64::seqedit::m_zoom [private]
```

The value of zoom is the same as the number of pixels per tick on the piano roll.

10.85.4.7 m_snap

```
int seq64::seqedit::m_snap [private]
```

10.85.4.8 m_note_length

```
int seq64::seqedit::m_note_length [private]
```

10.85.4.9 m_scale

```
int seq64::seqedit::m_scale [private]
```

10.85.4.10 m_chord

```
int seq64::seqedit::m_chord [private]
```

10.85.4.11 m_key

```
int seq64::seqedit::m_key [private]
```

10.85.4.12 m_bgsequence

```
int seq64::seqedit::m_bgsequence [private]
```

10.85.4.13 m_measures

```
long seq64::seqedit::m_measures [private]
```

10.85.4.14 m_ppqn

```
int seq64::seqedit::m_ppqn [private]
```

10.85.4.15 m_pp_whole

```
int seq64::seqedit::m_pp_whole [private]
```

10.85.4.16 m_pp_eighth

```
int seq64::seqedit::m_pp_eighth [private]
```

10.85.4.17 m_pp_sixteenth

```
int seq64::seqedit::m_pp_sixteenth [private]
```

10.85.4.18 m_seq

```
sequence& seq64::seqedit::m_seq [private]
```

10.85.4.19 m_menubar

```
Gtk::MenuBar* seq64::seqedit::m_menubar [private]
```

Many of these are menu items, and are associated with buttons that, when pressed, bring up the menu for display and selection of its entries. The top bar with menu buttons.

10.85.4.20 m_menu_tools

```
Gtk::Menu* seq64::seqedit::m_menu_tools [private]
```

10.85.4.21 m_menu_zoom

```
Gtk::Menu* seq64::seqedit::m_menu_zoom [private]
```

10.85.4.22 m_menu_snap

```
Gtk::Menu* seq64::seqedit::m_menu_snap [private]
```

10.85.4.23 m_menu_note_length

```
Gtk::Menu* seq64::seqedit::m_menu_note_length [private]
```

10.85.4.24 m_menu_length

```
Gtk::Menu* seq64::seqedit::m_menu_length [private]
```

10.85.4.25 m_toggle_transpose

```
Gtk::ToggleButton* seq64::seqedit::m_toggle_transpose [private]
```

10.85.4.26 m_image_transpose

Gtk::Image* seq64::seqedit::m_image_transpose [private]

10.85.4.27 m_menu_midich

Gtk::Menu* seq64::seqedit::m_menu_midich [private]

10.85.4.28 m_menu_midibus

Gtk::Menu* seq64::seqedit::m_menu_midibus [private]

10.85.4.29 m_menu_data

Gtk::Menu* seq64::seqedit::m_menu_data [private]

10.85.4.30 m_menu_minidata

Gtk::Menu* seq64::seqedit::m_menu_minidata [private]

10.85.4.31 m_menu_key

Gtk::Menu* seq64::seqedit::m_menu_key [private]

10.85.4.32 m_menu_scale

Gtk::Menu* seq64::seqedit::m_menu_scale [private]

10.85.4.33 m_menu_chords

Gtk::Menu* seq64::seqedit::m_menu_chords [private]

10.85.4.34 m_menu_sequences

Gtk::Menu* seq64::seqedit::m_menu_sequences [private]

10.85.4.35 m_menu_bpm

Gtk::Menu* seq64::seqedit::m_menu_bpm [private]

10.85.4.36 m_menu_bw

Gtk::Menu* seq64::seqedit::m_menu_bw [private]

10.85.4.37 m_menu_rec_vol

Gtk::Menu* seq64::seqedit::m_menu_rec_vol [private]

10.85.4.38 m_menu_rec_type

Gtk::Menu* seq64::seqedit::m_menu_rec_type [private]

10.85.4.39 m_vadjust

Gtk::Adjustment* seq64::seqedit::m_vadjust [private]

Vertical position descriptor.

10.85.4.40 m_hadjust

Gtk::Adjustment* seq64::seqedit::m_hadjust [private]

10.85.4.41 m_vscroll_new

Gtk::VScrollbar* seq64::seqedit::m_vscroll_new [private]

10.85.4.42 m_hscroll_new

Gtk::HScrollbar* seq64::seqedit::m_hscroll_new [private]

10.85.4.43 m_seqkeys_wid

seqkeys* seq64::seqedit::m_seqkeys_wid [private]

This item draws the piano-keys at the left of the seqedit window.

10.85.4.44 m_seqtime_wid

seqtime* seq64::seqedit::m_seqtime_wid [private]

This is the location where the measure numbers and the END marker are shown.

10.85.4.45 m_seqdata_wid

seqdata* seq64::seqedit::m_seqdata_wid [private]

This is the area at the bottom of the window that shows value lines for the selected kinds of events.

10.85.4.46 m_seqevent_wid

seqevent* seq64::seqedit::m_seqevent_wid [private]

10.85.4.47 m_seqroll_wid

seqroll* seq64::seqedit::m_seqroll_wid [private]

10.85.4.48 m_button_lfo

Gtk::Button* seq64::seqedit::m_button_lfo [private]

This item will always be an optional part of the build, enabled by defining SEQ64_STAZED_LFO_SUPPORT.

10.85.4.49 m_lfo_wnd

lfownd* seq64::seqedit::m_lfo_wnd [private]

This item get the seqdata window hooked into it, and so must follow that item in the C++ initializer list.

10.85.4.50 m_table

```
Gtk::Table* seq64::seqedit::m_table [private]
```

These items provide a number of buttons and text-entry fields, as well as their layout. The layout table for editor.

10.85.4.51 m_vbox

```
Gtk::VBox* seq64::seqedit::m_vbox [private]
```

10.85.4.52 m_hbox

```
Gtk::HBox* seq64::seqedit::m_hbox [private]
```

10.85.4.53 m_hbox2

```
Gtk::HBox* seq64::seqedit::m_hbox2 [private]
```

10.85.4.54 m_button_undo

```
Gtk::Button* seq64::seqedit::m_button_undo [private]
```

10.85.4.55 m_button_redo

```
Gtk::Button* seq64::seqedit::m_button_redo [private]
```

10.85.4.56 m_button_quantize

```
Gtk::Button* seq64::seqedit::m_button_quantize [private]
```

10.85.4.57 m_button_tools

```
Gtk::Button* seq64::seqedit::m_button_tools [private]
```


10.85.4.58 m_button_sequence

Gtk::Button* seq64::seqedit::m_button_sequence [private]

10.85.4.59 m_entry_sequence

Gtk::Entry* seq64::seqedit::m_entry_sequence [private]

10.85.4.60 m_button_bus

Gtk::Button* seq64::seqedit::m_button_bus [private]

10.85.4.61 m_entry_bus

Gtk::Entry* seq64::seqedit::m_entry_bus [private]

10.85.4.62 m_button_channel

Gtk::Button* seq64::seqedit::m_button_channel [private]

10.85.4.63 m_entry_channel

Gtk::Entry* seq64::seqedit::m_entry_channel [private]

10.85.4.64 m_button_snap

Gtk::Button* seq64::seqedit::m_button_snap [private]

10.85.4.65 m_entry_snap

Gtk::Entry* seq64::seqedit::m_entry_snap [private]

10.85.4.66 m_button_note_length

Gtk::Button* seq64::seqedit::m_button_note_length [private]

10.85.4.67 m_entry_note_length

Gtk::Entry* seq64::seqedit::m_entry_note_length [private]

10.85.4.68 m_button_zoom

Gtk::Button* seq64::seqedit::m_button_zoom [private]

10.85.4.69 m_entry_zoom

Gtk::Entry* seq64::seqedit::m_entry_zoom [private]

10.85.4.70 m_button_length

Gtk::Button* seq64::seqedit::m_button_length [private]

10.85.4.71 m_entry_length

Gtk::Entry* seq64::seqedit::m_entry_length [private]

10.85.4.72 m_button_key

Gtk::Button* seq64::seqedit::m_button_key [private]

10.85.4.73 m_entry_key

Gtk::Entry* seq64::seqedit::m_entry_key [private]

10.85.4.74 m_button_scale

Gtk::Button* seq64::seqedit::m_button_scale [private]

10.85.4.75 m_entry_scale

Gtk::Entry* seq64::seqedit::m_entry_scale [private]

10.85.4.76 m_button_chord

Gtk::Button* seq64::seqedit::m_button_chord [private]

10.85.4.77 m_entry_chord

Gtk::Entry* seq64::seqedit::m_entry_chord [private]

10.85.4.78 m_tooltips

Gtk::Tooltips* seq64::seqedit::m_tooltips [private]

10.85.4.79 m_button_data

Gtk::Button* seq64::seqedit::m_button_data [private]

10.85.4.80 m_button_minidata

Gtk::Button* seq64::seqedit::m_button_minidata [private]

10.85.4.81 m_entry_data

Gtk::Entry* seq64::seqedit::m_entry_data [private]

10.85.4.82 m_button_bpm

```
Gtk::Button* seq64::seqedit::m_button_bpm [private]
```

10.85.4.83 m_entry_bpm

```
Gtk::Entry* seq64::seqedit::m_entry_bpm [private]
```

10.85.4.84 m_button_bw

```
Gtk::Button* seq64::seqedit::m_button_bw [private]
```

10.85.4.85 m_entry_bw

```
Gtk::Entry* seq64::seqedit::m_entry_bw [private]
```

10.85.4.86 m_button_rec_vol

```
Gtk::Button* seq64::seqedit::m_button_rec_vol [private]
```

10.85.4.87 m_button_rec_type

```
Gtk::Button* seq64::seqedit::m_button_rec_type [private]
```

10.85.4.88 m_toggle_follow

```
Gtk::ToggleButton* seq64::seqedit::m_toggle_follow [private]
```

10.85.4.89 m_toggle_play

```
Gtk::ToggleButton* seq64::seqedit::m_toggle_play [private]
```

10.85.4.90 m_toggle_record

Gtk::ToggleButton* seq64::seqedit::m_toggle_record [private]

10.85.4.91 m_toggle_q_rec

Gtk::ToggleButton* seq64::seqedit::m_toggle_q_rec [private]

10.85.4.92 m_toggle_thru

Gtk::ToggleButton* seq64::seqedit::m_toggle_thru [private]

10.85.4.93 m_entry_seqnumber

Gtk::Entry* seq64::seqedit::m_entry_seqnumber [private]

10.85.4.94 m_entry_name

Gtk::Entry* seq64::seqedit::m_entry_name [private]

10.85.4.95 m_image_mousemode

Gtk::Image* seq64::seqedit::m_image_mousemode [private]

10.85.4.96 m_editing_status

midibyte seq64::seqedit::m_editing_status [private]

10.85.4.97 m_editing_cc

midibyte seq64::seqedit::m_editing_cc [private]

10.85.4.98 m_first_event

```
midibyte seq64::seqedit::m_first_event [private]
```

If no events exist, the value is 0x00.

10.85.4.99 m_first_event_name

```
std::string seq64::seqedit::m_first_event_name [private]
```

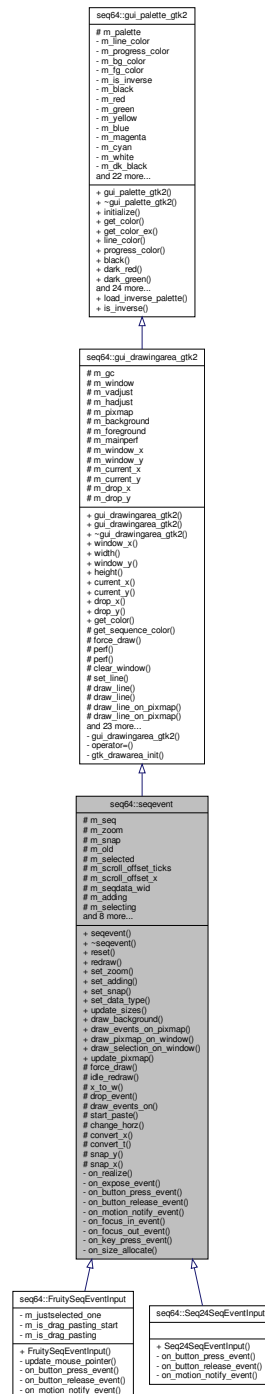
10.85.4.100 m_have_focus

```
bool seq64::seqedit::m_have_focus [private]
```

10.86 seq64::sequevent Class Reference

Implements the piano event drawing area.

Inheritance diagram for seq64::seqevent:



Public Member Functions

- [seqevent](#) ([perform](#) &p, [sequence](#) &seq, int zoom, int snap, [seqdata](#) &seqdata_wid, Gtk::Adjustment &hadjust)
Principal constructor.
- virtual [~seqevent](#) ()
Let's provide a do-nothing virtual destructor.
- void [reset](#) ()

- This function basically resets the whole widget as if it was realized again.*

 - void [redraw](#) ()

Adjusts the scrolling offset for ticks, updates the pixmap, and draws it on the window.
- void [set_zoom](#) (int zoom)

Sets zoom to the given value, and resets if the value ended up being changed.
- void [set_adding](#) (bool adding)

Changes the mouse cursor to a pencil or a left pointer in the given sequevent object, depending on the first parameter.
- void [set_snap](#) (int snap)

'Setter' function for member m_snap Simply sets the snap member.
- void [set_data_type](#) (midibyte status, midibyte control)

Sets the status to the given parameter, and the CC value to the given optional control parameter, which defaults to 0.
- void [update_sizes](#) ()

If the window is realized, this function creates a pixmap with the window dimensions, then updates the pixmap, and queues up a redraw.
- void [draw_background](#) ()

This function updates the background.
- void [draw_events_on_pixmap](#) ()

This function fills the main pixmap with events.
- void [draw_pixmap_on_window](#) ()

This function currently just queues up a draw operation for the pixmap.
- void [draw_selection_on_window](#) ()

Draw the selected events on the window.
- void [update_pixmap](#) ()

Redraws the background pixmap on the main pixmap, then puts the events on.

Protected Member Functions

- virtual void [force_draw](#) ()

Forces a draw on the current drawable area of the window.
- int [idle_redraw](#) ()

Implements redraw while idling.
- void [x_to_w](#) (int x1, int x2, int &x, int &w)

This function checks the mins / maxes.
- void [drop_event](#) (midipulse tick, bool istempo=false)

Drops (adds) an event at the given tick.
- void [draw_events_on](#) (Glib::RefPtr< Gdk::Drawable > draw)

Draws events on the given drawable object.
- void [start_paste](#) ()

Starts a paste operation.
- void [change_horz](#) ()

Changes the horizontal scrolling offset for ticks, then updates the pixmap and forces a redraw.
- void [convert_x](#) (int x, midipulse &tick)

Takes the screen x coordinate, multiplies it by the current zoom, and returns the tick value in the given parameter.
- void [convert_t](#) (midipulse tick, int &x)

Converts the given tick value to an x corrdinate, based on the zoom, and returns it via the second parameter.
- void [snap_y](#) (int &y)

This function performs a 'snap' on y.
- void [snap_x](#) (int &x)

This function performs a 'snap' on x.

Protected Attributes

- [sequence](#) & [m_seq](#)
Provides a reference to the sequence whose data is represented in this sequevent object.
- int [m_zoom](#)
Zoom setting, means that one pixel == m_zoom ticks.
- int [m_snap](#)
The grid-snap setting for the event bar grid.
- GdkRectangle [m_old](#)
Used in drawing the event selection in the thing event row.
- GdkRectangle [m_selected](#)
Used in moving and pasting the selected events in the thin event row.
- int [m_scroll_offset_ticks](#)
Provides the offset of the ticks in the event view based on where the scroll-bar has moved the view "window".
- int [m_scroll_offset_x](#)
Provides the offset of the pixels in the event view based on where the scroll-bar has moved the view "window".
- [seqdata](#) & [m_seqdata_wid](#)
The data view that parallels this event view.
- bool [m_adding](#)
True if we're adding events via the mouse.
- bool [m_selecting](#)
Used when highlighting a bunch of events.
- bool [m_moving_init](#)
Used externally by the fruityseq and seq24seq modules, to initialize the act of moving events.
- bool [m_moving](#)
Indicates that this pane is in the act of moving a selection.
- bool [m_growing](#)
Used externally by the fruityseq and seq24seq modules, when growing the event duration.
- bool [m_painting](#)
Used externally by the fruityseq and seq24seq modules, in painting the selected events.
- bool [m_paste](#)
Indicates that we've selected some events and are in paste mode.
- int [m_move_snap_offset_x](#)
Used externally by the fruityseq and seq24seq modules, in snapping.
- midibyte [m_status](#)
Indicates what is the data window currently editing.
- midibyte [m_cc](#)
Indicates what is the data window currently editing.

Private Member Functions

- virtual void [on_realize](#) ()
Implements the on-realize callback.
- virtual bool [on_expose_event](#) (GdkEventExpose *ev)
Implements the on-expose event callback.
- virtual bool [on_button_press_event](#) (GdkEventButton *ev)
Implements the on-button-press event callback.
- virtual bool [on_button_release_event](#) (GdkEventButton *ev)
Implements the on-button-release event callback.
- virtual bool [on_motion_notify_event](#) (GdkEventMotion *ev)

Implements the on-motion-notify event callback.

- virtual bool [on_focus_in_event](#) (GdkEventFocus *)
Responds to a focus event by setting the HAS_FOCUS flag.
- virtual bool [on_focus_out_event](#) (GdkEventFocus *)
Responds to a unfocus event by resetting the HAS_FOCUS flag.
- virtual bool [on_key_press_event](#) (GdkEventKey *p0)
Implements the key-press event callback function.
- virtual void [on_size_allocate](#) (Gtk::Allocation &)
Implements the on-size-allocate event callback.

Additional Inherited Members

10.86.1 Constructor & Destructor Documentation

10.86.1.1 `seqevent()`

```
seq64::seqevent::seqevent (
    perform & p,
    sequence & seq,
    int zoom,
    int snap,
    seqdata & seqdata_wid,
    Gtk::Adjustment & hadjust )
```

Parameters

<i>p</i>	The "parent" perform object controlling all of the sequences.
<i>seq</i>	The current sequence operated on by this object.
<i>zoom</i>	The initial zoom value.
<i>snap</i>	The initial snap value.
<i>seqdata_wid</i>	The data pane that this event pane is associated with.
<i>hadjust</i>	The horizontal scroll-bar.

10.86.1.2 `~seqevent()`

```
virtual seq64::seqevent::~~seqevent ( ) [inline], [virtual]
```

10.86.2 Member Function Documentation

10.86.2.1 reset()

```
void seq64::segevent::reset ( )
```

Basically identical to [seqtime::reset\(\)](#).

10.86.2.2 redraw()

```
void seq64::segevent::redraw ( )
```

Somewhat similar to [seqroll::redraw\(\)](#).

10.86.2.3 set_zoom()

```
void seq64::segevent::set_zoom (
    int z )
```

Parameters

<i>z</i>	The desired zoom value, assumed to be validated already. See the seqedit::set_zoom() function.
----------	--

10.86.2.4 set_adding()

```
void seq64::segevent::set_adding (
    bool adding )
```

Modifies `m_adding` as well.

Parameters

<i>adding</i>	The value to set <code>m_adding</code> to, and if true, sets the mouse cursor to a pencil icon, otherwise sets it to a standard mouse-pointer icon.
---------------	---

10.86.2.5 set_snap()

```
void seq64::segevent::set_snap (
    int snap ) [inline]
```

The parameter is not validated.

10.86.2.6 set_data_type()

```
void seq64::segevent::set_data_type (
    midibyte status,
    midibyte control )
```

Then redraws.

Parameters

<i>status</i>	The status/event byte to set. For example, EVENT_NOTE_ON and EVENT_NOTE off. This byte should have the channel nybble cleared.
<i>control</i>	The MIDI CC byte to set.

10.86.2.7 update_sizes()

```
void seq64::segevent::update_sizes ( )
```

This ends up filling the background with dotted lines, etc.

10.86.2.8 draw_background()

```
void seq64::segevent::draw_background ( )
```

It sets the foreground to white, draws the rectangle, in order to clear the pixmap. The build-time option SEQ64↔_SOLID_PIANOROLL_GRID causes solid lines to be drawn, in gray, instead of dotted black lines, for a smoother look.

Also, as a trial option, if the current data type is EVENT_NOTE_ON, EVENT_NOTE_OFF, and EVENT_AFTER↔TOUCH, we draw the background in light grey to remind the user that there are issues in copying or moving these events around (unlinked) by themselves.

10.86.2.9 draw_events_on_pixmap()

```
void seq64::segevent::draw_events_on_pixmap ( )
```

10.86.2.10 draw_pixmap_on_window()

```
void seq64::segevent::draw_pixmap_on_window ( )
```

Old comments:

```
It then tells event to do the same. We changed something on this
window, and chances are we need to update the event widget as well and
update our velocity window.
```

10.86.2.11 draw_selection_on_window()

```
void seq64::segevent::draw_selection_on_window ( )
```

10.86.2.12 update_pixmap()

```
void seq64::segevent::update_pixmap ( )
```

10.86.2.13 force_draw()

```
void seq64::segevent::force_draw ( ) [protected], [virtual]
```

Reimplemented from [seq64::gui_drawingarea_gtk2](#).

10.86.2.14 idle_redraw()

```
int seq64::segevent::idle_redraw ( ) [protected]
```

Who calls this routine? Probably the default timer routine, but not sure.

Returns

Always returns true.

10.86.2.15 x_to_w()

```
void seq64::segevent::x_to_w (
    int x1,
    int x2,
    int & x,
    int & w ) [protected]
```

Then it fills in x and the width.

Parameters

	<i>x1</i>	The "left" x value.
	<i>x2</i>	The "right" x value.
out	<i>x</i>	The destination for the converted x value.
out	<i>w</i>	The destination for the converted width value.

10.86.2.16 drop_event()

```
void seq64::segevent::drop_event (
    midipulse tick,
    bool istempo = false ) [protected]
```

It sets the first byte properly for after-touch, program-change, channel-pressure, and pitch-wheel. The type of event is determined by `m_status`.

Parameters

<i>tick</i>	The destination time (division, pulse, tick) for the event to be dropped at.
<i>istempo</i>	Indicates if the event is a tempo event. If so, we add a fake tempo event of BPM = 120. Defaults to false.

10.86.2.17 draw_events_on()

```
void seq64::segevent::draw_events_on (
    Glib::RefPtr< Gdk::Drawable > drawable ) [protected]
```

Very similar to [seqdata::draw_events_on\(\)](#).

This function exercises the new version of `get_next_event()`, `get_next_event_ex()`, which allows (and forces) the caller to provide the event iterator.

Parameters

<i>drawable</i>	The given drawable object.
-----------------	----------------------------

10.86.2.18 start_paste()

```
void seq64::segevent::start_paste ( ) [protected]
```

It gets the clipboard box that selected elements are in, makes a coordinate conversion, and then, sets the `m_↔` selected rectangle to hold the (x,y,w,h) of the selected events.

10.86.2.19 change_horz()

```
void seq64::segevent::change_horz ( ) [protected]
```

Very similar to [seqroll::change_horz\(\)](#). Basically identical to [seqdata::change_horz\(\)](#).

10.86.2.20 convert_x()

```
void seq64::segevent::convert_x (
    int x,
    midipulse & tick ) [inline], [protected]
```

Why not just return it normally?

Parameters

	<i>x</i>	The x (pixel) value to convert.
out	<i>tick</i>	The destination for the converted x value.

10.86.2.21 convert_t()

```
void seq64::segevent::convert_t (
    midipulse tick,
    int & x ) [inline], [protected]
```

Why not just return it normally?

Parameters

	<i>tick</i>	The tick (pulse) value to convert.
out	<i>x</i>	The destination for the converted tick value.

10.86.2.22 snap_y()

```
void seq64::segevent::snap_y (
    int & y ) [inline], [protected]
```

Parameters

out	<i>y</i>	The return parameter for the conversion. Why not just return the value?
-----	----------	---

10.86.2.23 snap_x()

```
void seq64::segevent::snap_x (
    int & x ) [protected]
```

- snap = number pulses to snap to

- `m_zoom` = number of pulses per pixel
- Therefore `snap / m_zoom` = number of pixels to snap to.

Parameters

<code>out</code>	<code>x</code>	The output destination for the snapped x value.
------------------	----------------	---

10.86.2.24 `on_realize()`

```
void seq64::segevent::on_realize ( ) [private], [virtual]
```

It calls the base-class version, and then allocates additional resource not allocated in the constructor. Finally, it connects up the `change_horz` function.

10.86.2.25 `on_expose_event()`

```
bool seq64::segevent::on_expose_event (
    GdkEventExpose * e ) [private], [virtual]
```

Parameters

<code>e</code>	The expose event.
----------------	-------------------

10.86.2.26 `on_button_press_event()`

```
bool seq64::segevent::on_button_press_event (
    GdkEventButton * ev ) [private], [virtual]
```

It distinguishes between the Seq24 and Fruity varieties of mouse interaction.

Odd. In the legacy code, each case fell through to the next case to the "default" case! We will assume for now that this is incorrect.

Note that returning "true" from a Gtkmm event-handler stops the propagation of the event to higher-level widgets. The Fruity and Seq24 event handlers return true, always. In the legacy code, though, the fall-through code caused false to be returned, always. Not sure what effect this had. Added some fixes, but then commented them out until better testing can be done.

Parameters

<code>ev</code>	The button event.
-----------------	-------------------

Returns

Returns true if the button-press was handled. Not sure the return code is meaningful.

Needs update. `m_seq.unselect(); ???????`

Reimplemented in [seq64::FruitySeqEventInput](#), and [seq64::Seq24SeqEventInput](#).

10.86.2.27 on_button_release_event()

```
bool seq64::segevent::on_button_release_event (
    GdkEventButton * ev ) [private], [virtual]
```

It distinguishes between the Seq24 and Fruity varieties of mouse interaction.

Odd. The fruity case fell through to the Seq24 case. We will assume for now that this is correct. Added some fixes, but then commented them out until better testing can be done.

Parameters

<i>ev</i>	The button event.
-----------	-------------------

Returns

Returns true if the button-press was handled.

Reimplemented in [seq64::FruitySeqEventInput](#), and [seq64::Seq24SeqEventInput](#).

10.86.2.28 on_motion_notify_event()

```
bool seq64::segevent::on_motion_notify_event (
    GdkEventMotion * ev ) [private], [virtual]
```

It distinguishes between the Seq24 and Fruity varieties of mouse interaction.

Odd. The fruity case fell through to the Seq24 case. We will assume for now that this is correct. Added some fixes, but then commented them out until better testing can be done.

Parameters

<i>ev</i>	The motion event.
-----------	-------------------

Returns

Returns true if the motion-event was handled.

Reimplemented in [seq64::FruitySeqEventInput](#), and [seq64::Seq24SeqEventInput](#).

10.86.2.29 on_focus_in_event()

```
bool seq64::segevent::on_focus_in_event (
    GdkEventFocus * ) [private], [virtual]
```

Parameter "ev" is the focus event, unused.

Returns

Always returns false.

10.86.2.30 on_focus_out_event()

```
bool seq64::segevent::on_focus_out_event (
    GdkEventFocus * ) [private], [virtual]
```

Parameter "ev" is the focus event, unused.

Returns

Always returns false.

10.86.2.31 on_key_press_event()

```
bool seq64::segevent::on_key_press_event (
    GdkEventKey * ev ) [private], [virtual]
```

It handles deleting a selection via the Backspace or Delete keys, cut via Ctrl-X, copy via Ctrl-C, paste via Ctrl-V, and undo via Ctrl-Z. Would be nice to provide redo functionality via Ctrl-Y. :-)

Parameters

ev	The key-press event.
----	----------------------

Returns

Returns true if an event was handled. Only some of the handled events also cause the perform modification flag to be set as a side-effect.

10.86.2.32 on_size_allocate()

```
void seq64::segevent::on_size_allocate (
    Gtk::Allocation & a ) [private], [virtual]
```

The m_window_x and m_window_y values are set to the allocation width and height, respectively.

Parameters

<i>a</i>	The allocation to be processed.
----------	---------------------------------

10.86.3 Field Documentation

10.86.3.1 m_seq

[sequence&](#) seq64::segevent::m_seq [protected]

10.86.3.2 m_zoom

int seq64::segevent::m_zoom [protected]

10.86.3.3 m_snap

int seq64::segevent::m_snap [protected]

Same meaning as for the piano roll. This value is the denominator of the note size used for the snap.

10.86.3.4 m_old

GdkRectangle seq64::segevent::m_old [protected]

10.86.3.5 m_selected

GdkRectangle seq64::segevent::m_selected [protected]

10.86.3.6 m_scroll_offset_ticks

```
int seq64::segevent::m_scroll_offset_ticks [protected]
```

10.86.3.7 m_scroll_offset_x

```
int seq64::segevent::m_scroll_offset_x [protected]
```

Set to m_scroll_offset_ticks divided by m_zoom.

10.86.3.8 m_seqdata_wid

```
seqdata& seq64::segevent::m_seqdata_wid [protected]
```

10.86.3.9 m_adding

```
bool seq64::segevent::m_adding [protected]
```

10.86.3.10 m_selecting

```
bool seq64::segevent::m_selecting [protected]
```

10.86.3.11 m_moving_init

```
bool seq64::segevent::m_moving_init [protected]
```

10.86.3.12 m_moving

```
bool seq64::segevent::m_moving [protected]
```

WARNING: This operation seems to have a bug. It makes the events very very long. This bug exists in Seq24.

10.86.3.13 m_growing

```
bool seq64::segevent::m_growing [protected]
```

Does growing work in this view? Need to do some better testing.

10.86.3.14 m_painting

```
bool seq64::segevent::m_painting [protected]
```

10.86.3.15 m_paste

```
bool seq64::segevent::m_paste [protected]
```

10.86.3.16 m_move_snap_offset_x

```
int seq64::segevent::m_move_snap_offset_x [protected]
```

10.86.3.17 m_status

```
midibyte seq64::segevent::m_status [protected]
```

The current status/event byte.

10.86.3.18 m_cc

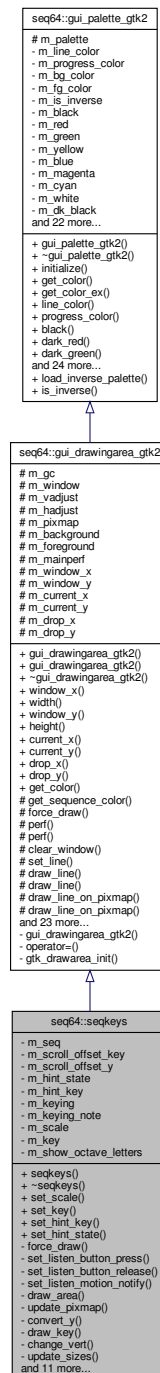
```
midibyte seq64::segevent::m_cc [protected]
```

The current MIDI CC value.

10.87 seq64::seqkeys Class Reference

This class implements the left side piano of the pattern/sequence editor.

Inheritance diagram for seq64::seqkeys:



Public Member Functions

- `seqkeys` (`sequence &seq`, `perform &p`, `Gtk::Adjustment &vadjust`)
Principal constructor.
- virtual `~seqkeys` ()
Let's provide a do-nothing virtual destructor.
- void `set_scale` (int scale)

- Sets the musical scale, then resets.*
 - void [set_key](#) (int key)
- Sets the musical key, then resets.*
 - void [set_hint_key](#) (int key)
- Sets a key to grey so that it can serve as a scale hint.*
 - void [set_hint_state](#) (bool state)
- Sets the hint state to the given value.*

Private Member Functions

- virtual void [force_draw](#) ()*Forces a draw operation on the whole window.*
- void [set_listen_button_press](#) (GdkEventButton *ev)*Sneaky accessors for the seqroll friend.*
- void [set_listen_button_release](#) (GdkEventButton *ev)
- void [set_listen_motion_notify](#) (GdkEventMotion *ev)
- void [draw_area](#) ()*Draws the updated pixmap on the drawable area of the window where the keys' location is hardwired.*
- void [update_pixmap](#) ()*Updates the pixmaps to prepare it for the next draw operation.*
- void [convert_y](#) (int y, int ¬e)*Takes the screen y coordinate, and returns the note value in the second parameter.*
- void [draw_key](#) (int key, bool state)*Draws the given key according to the given state.*
- void [change_vert](#) ()*Changes the y offset of the scrolling, and the forces a draw.*
- void [update_sizes](#) ()
- void [reset](#) ()*Resetting the keys view updates the pixmap and queues up a draw operation.*
- bool [is_black_key](#) (int key) const*Detects a black key.*
- void [on_realize](#) ()*Implements the on-realize event.*
- bool [on_expose_event](#) (GdkEventExpose *ev)*Implements the on-expose event, by drawing on the window.*
- bool [on_button_press_event](#) (GdkEventButton *ev)*Implements the on-button-press event callback.*
- bool [on_button_release_event](#) (GdkEventButton *ev)*Implements the on-button-release event callback.*
- bool [on_motion_notify_event](#) (GdkEventMotion *p0)*Implements the on-motion-notify event handler.*
- bool [on_enter_notify_event](#) (GdkEventCrossing *p0)*Implements the on-enter notification event handler.*
- bool [on_leave_notify_event](#) (GdkEventCrossing *p0)*Implements the on-leave notification event handler.*
- bool [on_scroll_event](#) (GdkEventScroll *ev)*Implements the on-scroll-event notification event handler.*
- void [on_size_allocate](#) (Gtk::Allocation &)*Implements the on-size-allocation notification event handler.*

Private Attributes

- [sequence](#) & [m_seq](#)
The sequence object that the keys pane will be using.
- int [m_scroll_offset_key](#)
Provides the value of the current top key in the keys pane.
- int [m_scroll_offset_y](#)
Provides the value of the current top key in the keys pane in units of relative pixels.
- bool [m_hint_state](#)
Indicates if a piano key is set to indicate where on the pitch scale the mouse cursor is sitting.
- int [m_hint_key](#)
Indicates the current y-value of the mouse pointer in units of key value.
- bool [m_keying](#)
Set to true while the left mouse button is being pressed.
- int [m_keying_note](#)
The note to be played when selected in the seqkeys pane.
- int [m_scale](#)
This member holds the scale value for the musical scale for the current edit of the sequence.
- int [m_key](#)
This member holds the key value for the musical key for the current edit of the sequence.
- bool [m_show_octave_letters](#)
The default value is to show the octave letters on the vertical virtual keyboard.

Friends

- class [seqroll](#)
- class [FruitySeqRollInput](#)

Additional Inherited Members

10.87.1 Detailed Description

Note the friends of this class, [seqroll](#) and [FruitySeqRollInput](#). Where is [Seq24SeqRollInput](#)? Gone. It has been folded back into [seqroll](#).

10.87.2 Constructor & Destructor Documentation

10.87.2.1 [seqkeys\(\)](#)

```
seq64::seqkeys::seqkeys (
    sequence & seq,
    perform & p,
    Gtk::Adjustment & vadjust )
```


Parameters

<i>seq</i>	Provides the sequence object to which this seqkeys pane is associated.
<i>p</i>	Provides the performance object to which this seqkeys pane (and all sequences) are associated.
<i>vadjust</i>	The range object for the vertical scrollbar linked to the position in the seqkeys pane.

10.87.2.2 ~seqkeys()

```
virtual seq64::seqkeys::~seqkeys ( ) [inline], [virtual]
```

10.87.3 Member Function Documentation

10.87.3.1 set_scale()

```
void seq64::seqkeys::set_scale (
    int scale )
```

This function is called by the seqedit class.

Parameters

<i>scale</i>	The musical scale value to be set.
--------------	------------------------------------

10.87.3.2 set_key()

```
void seq64::seqkeys::set_key (
    int key )
```

Parameters

<i>key</i>	The musical key value to be set.
------------	----------------------------------

10.87.3.3 set_hint_key()

```
void seq64::seqkeys::set_hint_key (
    int key )
```

If `m_hint_state` is true, the key is drawn (again).

Parameters

<i>key</i>	The key value to set the hint-key to.
------------	---------------------------------------

10.87.3.4 set_hint_state()

```
void seq64::seqkeys::set_hint_state (
    bool state )
```

Parameters

<i>state</i>	Provides the value for hinting, where true == on, false == off.
--------------	---

10.87.3.5 force_draw()

```
void seq64::seqkeys::force_draw ( ) [private], [virtual]
```

Unlike most other overridden versions of [force_draw\(\)](#), this one does not call the base-class version.

Reimplemented from [seq64::gui_drawingarea_gtk2](#).

10.87.3.6 set_listen_button_press()

```
void seq64::seqkeys::set_listen_button_press (
    GdkEventButton * ev ) [inline], [private]
```

From the stazed code.

Parameters

<i>ev</i>	The event to be forwarded from the seqroll.
-----------	---

10.87.3.7 set_listen_button_release()

```
void seq64::seqkeys::set_listen_button_release (
    GdkEventButton * ev ) [inline], [private]
```

10.87.3.8 set_listen_motion_notify()

```
void seq64::seqkeys::set_listen_motion_notify (
    GdkEventMotion * ev ) [inline], [private]
```

10.87.3.9 draw_area()

```
void seq64::seqkeys::draw_area ( ) [private]
```

10.87.3.10 update_pixmap()

```
void seq64::seqkeys::update_pixmap ( ) [private]
```

This function draws the keys, which range from 0 to 127 (SEQ64_MIDI_COUNT_MAX - 1 = c_num_keys - 1). Every octave, a key letter and number (e.g. "C4") is shown. The letter is adjusted to match the current scale (e.g. "C#4").

We want to support an option to show the key number rather than the note letter/number combination, and perhaps to toggle between them. The current difficulty is that the fonts used are just a little too high to fit within the vertical limits of each key. We really don't want to change the vertical size at this time, so we just print every other note value.

Also note that this algorithm draws from the top down, so we have to account for that.

10.87.3.11 convert_y()

```
void seq64::seqkeys::convert_y (
    int y,
    int & note ) [private]
```

Parameters

	<i>y</i>	The y (vertical) screen coordinate to convert.
out	<i>note</i>	The destination for the note calculation. This would be better as a return value.

10.87.3.12 draw_key()

```
void seq64::seqkeys::draw_key (
    int key,
    bool state ) [private]
```

It accounts for the black keys and the white keys, and for the highlighting of the active key.

Parameters

<i>key</i>	The key to be drawn.
<i>state</i>	How the key is to be drawn, where false == normal, true == grayed. A key is greyed when the mouse cursor is at the same vertical location on the piano as the key.

10.87.3.13 change_vert()

```
void seq64::seqkeys::change_vert ( ) [private]
```

Weird, in seq24 and here, the following was used, completely by accident! We fixed it, but must beware!

```
m_scroll_offset_y = m_scroll_offset_key * c_key_y, // comma operator!!!  
force_draw();
```

10.87.3.14 update_sizes()

```
void seq64::seqkeys::update_sizes ( ) [private]
```

10.87.3.15 reset()

```
void seq64::seqkeys::reset ( ) [private]
```

10.87.3.16 is_black_key()

```
bool seq64::seqkeys::is_black_key (  
    int key ) const [inline], [private]
```

Parameters

<i>key</i>	The key to analyze.
------------	---------------------

Returns

Returns true if the key is black (value 1, 3, 6, 8, or 10).

10.87.3.17 on_realize()

```
void seq64::seqkeys::on_realize ( ) [private]
```

Call the base-class version and then allocates resources that could not be allocated in the constructor. It connects the [change_vert\(\)](#) function and then calls it.

10.87.3.18 on_expose_event()

```
bool seq64::seqkeys::on_expose_event (
    GdkEventExpose * ev ) [private]
```

Parameters

<i>ev</i>	The expose-event object.
-----------	--------------------------

10.87.3.19 on_button_press_event()

```
bool seq64::seqkeys::on_button_press_event (
    GdkEventButton * ev ) [private]
```

It handles the left and right buttons. The left button, pressed on the piano keyboard, causes m_keying to be set to true, and the given note to play. The right button toggles the note display between letter/number and MIDI note number.

Parameters

<i>ev</i>	The mouse-button event to use.
-----------	--------------------------------

Returns

Always returns true.

10.87.3.20 on_button_release_event()

```
bool seq64::seqkeys::on_button_release_event (
    GdkEventButton * ev ) [private]
```

It currently handles only the left button, and only if m_keying is true.

This function is used after pressing on one of the keys on the left-side piano keyboard, to make it play, and turns off the playing of the note.

Parameters

<i>ev</i>	The button-event.
-----------	-------------------

Returns

Always returns true.

10.87.3.21 on_motion_notify_event()

```
bool seq64::seqkeys::on_motion_notify_event (
    GdkEventMotion * p0 ) [private]
```

This allows rolling down the keyboard, playing the notes one-by-one.

Parameters

<i>p0</i>	The motion event.
-----------	-------------------

Returns

Always returns false.

10.87.3.22 on_enter_notify_event()

```
bool seq64::seqkeys::on_enter_notify_event (
    GdkEventCrossing * p0 ) [private]
```

This greys the current key.

10.87.3.23 on_leave_notify_event()

```
bool seq64::seqkeys::on_leave_notify_event (
    GdkEventCrossing * p0 ) [private]
```

This un-greys the current key and stops playing the note.

10.87.3.24 on_scroll_event()

```
bool seq64::seqkeys::on_scroll_event (
    GdkEventScroll * ev ) [private]
```

Note that there is no usage of the modifier keys (e.g. Shift or Ctrl). Compare this function to [seqedit::on_scroll_event\(\)](#).

Parameters

<i>ev</i>	Provides the direction of the scroll event.
-----------	---

Returns

Always returns true.

10.87.3.25 on_size_allocate()

```
void seq64::seqkeys::on_size_allocate (
    Gtk::Allocation & all ) [private]
```

Parameters

<i>all</i>	Provies the allocation and its width and height.
------------	--

10.87.4 Friends And Related Function Documentation

10.87.4.1 seqroll

```
friend class seqroll [friend]
```

10.87.4.2 FruitySeqRollInput

```
friend class FruitySeqRollInput [friend]
```

10.87.5 Field Documentation

10.87.5.1 m_seq

```
sequence& seq64::seqkeys::m_seq [private]
```

10.87.5.2 m_scroll_offset_key

```
int seq64::seqkeys::m_scroll_offset_key [private]
```

Modified in [change_vert\(\)](#).

10.87.5.3 m_scroll_offset_y

```
int seq64::seqkeys::m_scroll_offset_y [private]
```

Modified in [change_vert\(\)](#).

10.87.5.4 m_hint_state

```
bool seq64::seqkeys::m_hint_state [private]
```

10.87.5.5 m_hint_key

```
int seq64::seqkeys::m_hint_key [private]
```

10.87.5.6 m_keying

```
bool seq64::seqkeys::m_keying [private]
```

Used in playing the sound for each note as it is clicked in the seqkeys pane.

10.87.5.7 m_keying_note

```
int seq64::seqkeys::m_keying_note [private]
```

10.87.5.8 m_scale

```
int seq64::seqkeys::m_scale [private]
```

10.87.5.9 m_key

```
int seq64::seqkeys::m_key [private]
```


10.87.5.10 m_show_octave_letters

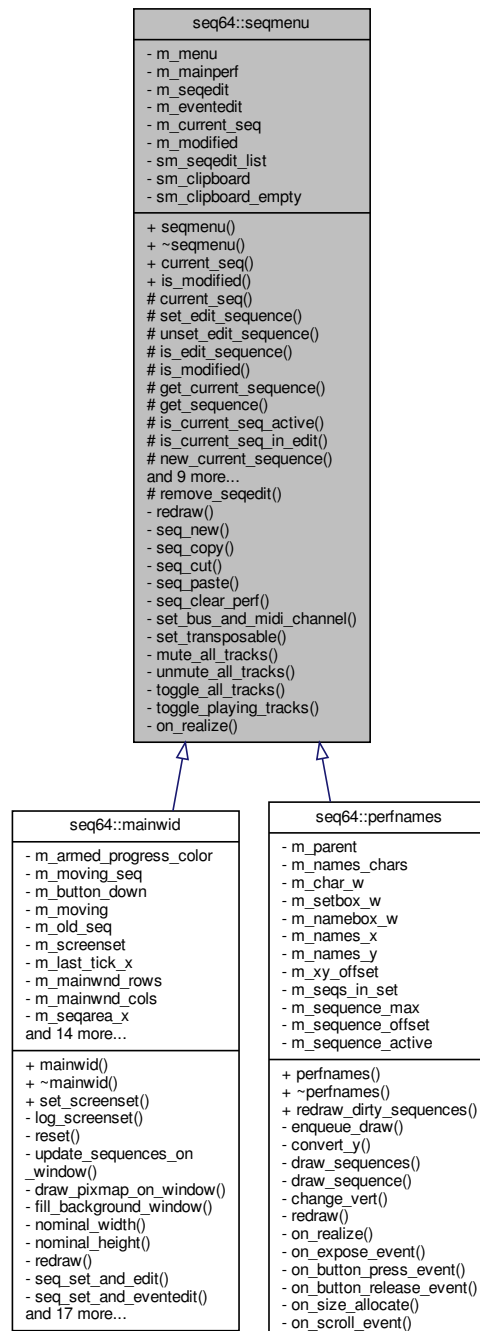
```
bool seq64::seqkeys::m_show_octave_letters [private]
```

If false, then the MIDI key numbers are shown instead. This is a new feature of Sequencer64.

10.88 seq64::seqmenu Class Reference

This class handles the right-click menu of the sequence slots in the pattern window.

Inheritance diagram for seq64::seqmenu:



Public Member Functions

- [seqmenu](#) ([perform](#) &a_p)
Principal constructor.
- virtual [~seqmenu](#) ()
Provides a rote base-class destructor.
- int [current_seq](#) () const

'Getter' function for member `m_current_seq` We're changing the name, so that "seq" indicates an integer by (an imperfect) convention.

- bool `is_modified` () const

'Getter' function for member `m_modified`

Protected Member Functions

- void `current_seq` (int seq)

'Setter' function for member `m_current_seq`

- void `set_edit_sequence` (int seqnum)

'Setter' function for member `m_edit_sequence` Pass in -1 to disable the edit-sequence number.

- void `unset_edit_sequence` (int seqnum)

'Setter' function for member `m_edit_sequence` Disable the edit-sequence number if it matches the parameter.

- bool `is_edit_sequence` (int seqnum) const

'Getter' function for member `m_edit_sequence` Tests the parameter against `m_edit_sequence`.

- void `is_modified` (bool flag)

'Setter' function for member `m_modified`

- `sequence * get_current_sequence` () const

'Getter' function for member `m_mainperf.get_sequence(current_seq())` This call is used many, many times, and well worth wrapping.

- `sequence * get_sequence` (int seqnum) const

Forwards the get-sequence call to the perform object.

- bool `is_current_seq_active` () const

Forwards the is-sequence-active check to the perform object.

- bool `is_current_seq_in_edit` () const

Forwards the is-sequence-in-edit check to the perform object.

- void `new_current_sequence` ()

Forwards the new-current-sequence call to the perform object.

- void `new_sequence` (int seqnum)

Forwards the new-sequence call to the perform object.

- void `delete_current_sequence` ()

Forwards the delete-sequence call to the perform object.

- void `toggle_current_sequence` ()

Forwards the sequence-playing-toggle call to the perform object.

- void `popup_menu` ()

This function sets up the pattern menu entries.

- void `seq_edit` ()

This menu callback launches the sequence-editor (pattern editor) window.

- void `seq_event_edit` ()

This menu callback launches the new event editor window.

- `seqedit * create_seqedit` (`sequence &s`)

A wrapper function so that we can not only create a new `seqedit` object, but have some management over it.

- virtual void `seq_set_and_edit` (int seqnum)

Sets the current sequence and then acts as if the user had clicked on its slot.

- virtual void `seq_set_and_eventedit` (int seqnum)

Sets the current sequence and then acts as if the user had right-clicked on its slot and selected "Event Edit".

Static Protected Member Functions

- static void `remove_seqedit` (`sequence &s`)

A wrapper function to make sure the `seqedit` object is removed from the list when it goes away.

Private Types

- typedef std::map< int, [seqedit](#) * > [SeqeditMap](#)
An easy type definition for a map of seqedit pointers keyed by the sequence number.
- typedef std::pair< int, [seqedit](#) * > [SeqeditPair](#)
A pair to make an entry to add to the seqedit map.
- typedef std::map< int, [seqedit](#) * >::iterator [iterator](#)
An iterator for the seqedit map.
- typedef std::map< int, [seqedit](#) * >::const_iterator [const_iterator](#)
A const iterator for the seqedit map.

Private Member Functions

- virtual void [redraw](#) (int a_sequence)=0
- void [seq_new](#) ()
This function sets the new sequence into the perform object, a bit prematurely, though.
- void [seq_copy](#) ()
Copies the selected (current) sequence to the clipboard sequence.
- void [seq_cut](#) ()
Deletes the selected (current) sequence and copies it to the clipboard sequence, if it is not in edit mode.
- void [seq_paste](#) ()
Pastes the sequence clipboard into the current sequence, if the current sequence slot is not active.
- void [seq_clear_perf](#) ()
If the current sequence is active, this function pushes a trigger undo in the main perform object, clears its sequence triggers for the current sequence, and sets the dirty flag of the sequence.
- void [set_bus_and_midi_channel](#) (int a_bus, int a_ch)
Sets up the bus, MIDI channel, and dirtiness flag of the current sequence in the main perform object, as per the give parameters.
- void [set_transposable](#) (bool flag)
Sets the "is-transposable" flag of the current sequence.
- void [mute_all_tracks](#) ()
Mutes all tracks in the main perform object.
- void [unmute_all_tracks](#) ()
Unmutes all tracks in the main perform object.
- void [toggle_all_tracks](#) ()
Toggles the mute-status of all tracks in the main perform object.
- void [toggle_playing_tracks](#) ()
Toggles the mute-status of only the playing tracks in the main perform object.
- void [on_realize](#) ()

Private Attributes

- Gtk::Menu * [m_menu](#)
The menu to pop up when the right-click action is used either on a mainwid pattern slot or on a perfedit pattern name.
- [perform](#) & [m_mainperf](#)
Provides a reference to the central (non-UI) object involved in managing a song and performance.
- [seqedit](#) * [m_seqedit](#)
Points to the latest seqedit object, if created.
- [eventedit](#) * [m_eventedit](#)
Points to the latest eventedit object, if created.
- int [m_current_seq](#)
References the current sequence by sequence number.
- bool [m_modified](#)
Indicates if a sequence has been created.

Static Private Attributes

- static [SeqeditMap sm_seqedit_list](#)
Holds a list of the currently open seqedit objects, stored as pointers keyed by the sequence number.
- static [sequence sm_clipboard](#)
Holds a copy of data concerning a sequence, which can then be pasted into another pattern slot.
- static bool [sm_clipboard_empty](#)
Indicates if the common clipboard is empty.

Friends

- class [mainwnd](#)
- class [seqedit](#)

10.88.1 Detailed Description

It is an abstract base class.

10.88.2 Member Typedef Documentation

10.88.2.1 SeqeditMap

```
typedef std::map<int, seqedit *> seq64::seqmenu::SeqeditMap [private]
```

10.88.2.2 SeqeditPair

```
typedef std::pair<int, seqedit *> seq64::seqmenu::SeqeditPair [private]
```

10.88.2.3 iterator

```
typedef std::map<int, seqedit *>::iterator seq64::seqmenu::iterator [private]
```

10.88.2.4 const_iterator

```
typedef std::map<int, seqedit *>::const_iterator seq64::seqmenu::const\_iterator [private]
```

10.88.3 Constructor & Destructor Documentation

10.88.3.1 seqmenu()

```
seq64::seqmenu::seqmenu (  
    perform & p )
```

Apart from filling in some of the members, this function initializes the clipboard, so that we don't get a crash on a paste with no previous copy.

Parameters

<i>p</i>	The main performance object representing the whole MIDI song.
----------	---

10.88.3.2 `~seqmenu()`

```
seq64::seqmenu::~~seqmenu ( ) [virtual]
```

A rote destructor.

This is necessary in an abstraction base class.

If we determine that we need to delete the `m_seqedit` pointer, we can do it here. But that is not likely, because we can have many new `seqedit` objects in play, because we can edit many at once.

10.88.4 Member Function Documentation**10.88.4.1** `current_seq()` [1/2]

```
int seq64::seqmenu::current_seq ( ) const [inline]
```

10.88.4.2 `is_modified()` [1/2]

```
bool seq64::seqmenu::is_modified ( ) const [inline]
```

10.88.4.3 `current_seq()` [2/2]

```
void seq64::seqmenu::current_seq (
    int seq ) [inline], [protected]
```

10.88.4.4 `set_edit_sequence()`

```
void seq64::seqmenu::set_edit_sequence (
    int seqnum ) [inline], [protected]
```

Now a pass-along to the perform object.

10.88.4.5 unset_edit_sequence()

```
void seq64::seqmenu::unset_edit_sequence (
    int seqnum ) [inline], [protected]
```

10.88.4.6 is_edit_sequence()

```
bool seq64::seqmenu::is_edit_sequence (
    int seqnum ) const [inline], [protected]
```

Returns true if that member is not -1, and the parameter matches it. Now a pass-along to the perform object.

10.88.4.7 is_modified() [2/2]

```
void seq64::seqmenu::is_modified (
    bool flag ) [inline], [protected]
```

10.88.4.8 get_current_sequence()

```
sequence* seq64::seqmenu::get_current_sequence ( ) const [inline], [protected]
```

10.88.4.9 get_sequence()

```
sequence* seq64::seqmenu::get_sequence (
    int seqnum ) const [inline], [protected]
```

10.88.4.10 is_current_seq_active()

```
bool seq64::seqmenu::is_current_seq_active ( ) const [inline], [protected]
```

10.88.4.11 is_current_seq_in_edit()

```
bool seq64::seqmenu::is_current_seq_in_edit ( ) const [inline], [protected]
```

10.88.4.12 new_current_sequence()

```
void seq64::seqmenu::new_current_sequence ( ) [inline], [protected]
```

10.88.4.13 new_sequence()

```
void seq64::seqmenu::new_sequence (
    int seqnum ) [inline], [protected]
```

10.88.4.14 delete_current_sequence()

```
void seq64::seqmenu::delete_current_sequence ( ) [inline], [protected]
```

10.88.4.15 toggle_current_sequence()

```
void seq64::seqmenu::toggle_current_sequence ( ) [inline], [protected]
```

10.88.4.16 popup_menu()

```
void seq64::seqmenu::popup_menu ( ) [protected]
```

It also sets up the pattern popup menu entries that are used in mainwid. Note that, for the selected sequence, the "Edit" and "Event Edit" menu entries are not included if a pattern editor or event editor is already running. The new event editor seems to create far-reaching problems that we do not yet understand, so it is now possible to disable it at build time. We have mitigated most of those problems by not allowing both a [seq_edit\(\)](#) and a [seq_event_edit\(\)](#) at the same time.

10.88.4.17 seq_edit()

```
void seq64::seqmenu::seq_edit ( ) [protected]
```

If it is already open for that sequence, this function just raises it.

Note that the `m_seqedit` member to which we save the new pointer is currently there just to avoid a compiler warning.

Also, if a new sequences is created, we set the `m_modified` flag to true, even though the sequence might later be deleted. Too much modification to keep track of!

An oddity is that calling `show_all()` here does not work unless the [seqedit\(\)](#) constructor makes its `show_all()` call.

10.88.4.18 seq_event_edit()

```
void seq64::seqmenu::seq_event_edit ( ) [protected]
```

If it is already open for that sequence, this function just raises it.

Note that the `m_eventedit` member to which we save the new pointer is currently there just to avoid a compiler warning.

This menu entry is available only if the selected sequence is active. That is, if the sequence has already been created.

An oddity is that we need the `show_all()` call here in order to see the dialog. A situation different from that for `seqedit`! However, now it doesn't seem to be needed, and we have put it back into the `eventedit` constructor.

10.88.4.19 create_seqedit()

```
seqedit * seq64::seqmenu::create_seqedit (
    sequence & s ) [protected]
```

We don't bother checking here if the insert succeeded. If it doesn't, all bets are off.

Parameters

s	Provides the sequence for which the <code>seqedit</code> will be created. The <code>perform</code> object and the <code>current_seq()</code> value are implicit parameters. This object can obviously be modified by the sequence editor, so cannot be constant.
---	--

Returns

Returns the pointer to the new `seqedit` object.

10.88.4.20 remove_seqedit()

```
void seq64::seqmenu::remove_seqedit (
    sequence & s ) [static], [protected]
```

Called by `seqedit::on_delete_event()`.

10.88.4.21 seq_set_and_edit()

```
void seq64::seqmenu::seq_set_and_edit (
    int seqnum ) [protected], [virtual]
```

How do we account for the current screenset? It might not matter if the mute/unmute keystrokes were designed to work only with the current screenset.

Change Note ca 2016-11-01 We would like to be able to right-click on a given pattern slot in `mainwid`, and figure out if it has a `seqedit` window open, so that we can update that window. So we need to add that `seqedit` window to a map of `seqedit`s, keyed by the slot number. Then we can look up that slot and see if it has a `seqedit` window open. If the `seqedit` window closes, that window needs to remove itself from the map. This won't be needed for the event editor, which has no functionality from `seqmenu`.

Parameters

<i>seqnum</i>	The number of the sequence to edit.
---------------	-------------------------------------

Reimplemented in [seq64::mainwid](#).

10.88.4.22 `seq_set_and_eventedit()`

```
void seq64::seqmenu::seq_set_and_eventedit (
    int seqnum ) [protected], [virtual]
```

Parameters

<i>seqnum</i>	The number of the sequence to event-edit.
---------------	---

Reimplemented in [seq64::mainwid](#).

10.88.4.23 `redraw()`

```
virtual void seq64::seqmenu::redraw (
    int a_sequence ) [private], [pure virtual]
```

Implemented in [seq64::mainwid](#), and [seq64::perfnames](#).

10.88.4.24 `seq_new()`

```
void seq64::seqmenu::seq_new ( ) [private]
```

For one thing, if [current_seq\(\)](#) is either a -1 or is greater than the maximum allowed sequence number, [perform↵::is_active\(\)](#) will return false, and we have no idea whether the sequence is not active or the sequence number is just invalid. So we need to check the pointer we got before trying to use it.

10.88.4.25 `seq_copy()`

```
void seq64::seqmenu::seq_copy ( ) [private]
```

We use a more appropriate function than operator `=()` here: [sequence::partial_assign\(\)](#).

Todo Can be offloaded to a `perform` member function that accepts a sequence clipboard non-const reference parameter.

10.88.4.26 seq_cut()

```
void seq64::seqmenu::seq_cut ( ) [private]
```

Todo A lot of [seq_cut\(\)](#) can be offloaded to a (new) perform member function that takes a sequence clipboard non-const reference parameter.

10.88.4.27 seq_paste()

```
void seq64::seqmenu::seq_paste ( ) [private]
```

Then it sets the dirty flag for the destination sequence.

Todo All of [seq_paste\(\)](#) can be offloaded to a (new) perform member function with a const clipboard reference parameter.

10.88.4.28 seq_clear_perf()

```
void seq64::seqmenu::seq_clear_perf ( ) [private]
```

10.88.4.29 set_bus_and_midi_channel()

```
void seq64::seqmenu::set_bus_and_midi_channel (
    int bus,
    int ch ) [private]
```

Parameters

<i>bus</i>	The MIDI buss number to set (bus vs buss? You decide.)
<i>ch</i>	The MIDI channel number to set.

10.88.4.30 set_transposable()

```
void seq64::seqmenu::set_transposable (
    bool flag ) [private]
```

Parameters

<i>flag</i>	The value to use to set the flag.
-------------	-----------------------------------

10.88.4.31 mute_all_tracks()

```
void seq64::seqmenu::mute_all_tracks ( ) [inline], [private]
```

10.88.4.32 unmute_all_tracks()

```
void seq64::seqmenu::unmute_all_tracks ( ) [inline], [private]
```

10.88.4.33 toggle_all_tracks()

```
void seq64::seqmenu::toggle_all_tracks ( ) [inline], [private]
```

10.88.4.34 toggle_playing_tracks()

```
void seq64::seqmenu::toggle_playing_tracks ( ) [inline], [private]
```

Note that the perform object will do this action only in Live mode.

10.88.4.35 on_realize()

```
void seq64::seqmenu::on_realize ( ) [private]
```

10.88.5 Friends And Related Function Documentation

10.88.5.1 mainwnd

```
friend class mainwnd [friend]
```

10.88.5.2 seqedit

```
friend class seqedit [friend]
```

10.88.6 Field Documentation

10.88.6.1 sm_seqedit_list

```
seqmenu::SeqeditMap seq64::seqmenu::sm_seqedit_list [static], [private]
```

The single map of seqedit objects, for seqedit updates and management.

We can use this map to look up patterns that we want to change from the right-click seqmenu, and modify the seqedit affected if it is found in the list.

Currently selectable by the USE_SEQEDIT_MACRO until we can make it foolproof.

10.88.6.2 sm_clipboard

```
sequence seq64::seqmenu::sm_clipboard [static], [private]
```

Current saved sequence data.

10.88.6.3 sm_clipboard_empty

```
bool seq64::seqmenu::sm_clipboard_empty [static], [private]
```

Indicates if sequence data is present or not.

10.88.6.4 m_menu

```
Gtk::Menu* seq64::seqmenu::m_menu [private]
```

10.88.6.5 m_mainperf

```
perform& seq64::seqmenu::m_mainperf [private]
```

10.88.6.6 m_seqedit

```
seqedit* seq64::seqmenu::m_seqedit [private]
```

Change Note Added by Chris on 2015-08-02 based on compiler warnings and a comment warning in the [seq_edit\(\)](#) function. We'll save the result of that function here, and will let valgrind tell us later if Gtkmm takes care of it.

10.88.6.7 m_eventedit

```
eventedit* seq64::seqmenu::m_eventedit [private]
```

10.88.6.8 m_current_seq

```
int seq64::seqmenu::m_current_seq [private]
```

10.88.6.9 m_modified

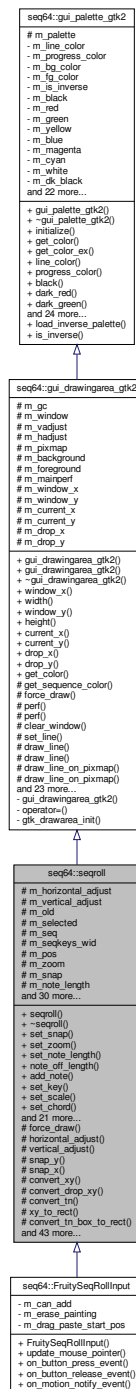
```
bool seq64::seqmenu::m_modified [private]
```

Todo We need to make sure that the perform object is in control of the modification flag.

10.89 seq64::seqroll Class Reference

Implements the piano roll section of the pattern editor.

Inheritance diagram for seq64::seqroll:



Public Member Functions

- [seqroll](#) ([perform](#) &[perf](#), [sequence](#) &seq, int zoom, int snap, [seqkeys](#) &seqkeys_wid, int pos, Gtk::Adjustment &hadjust, Gtk::Adjustment &vadjust)

Principal constructor.

- virtual [~seqroll](#) ()

Provides a destructor to delete allocated objects.

- void [set_snap](#) (int snap)
Sets the snap to the given value, and then resets the view.
- void [set_zoom](#) (int zoom)
Sets the zoom to the given value, and then resets the view.
- void [set_note_length](#) (int note_length)
'Setter' function for member `m_note_length`
- int [note_off_length](#) () const
'Getter' function for member `m_note_length`, adjusted for the `note_off_margin`.
- bool [add_note](#) (midipulse tick, int note, bool paint=true)
Convenience wrapper for [sequence::add_note\(\)](#).
- void [set_key](#) (int key)
Sets the music key to the given value, and then resets the view.
- void [set_scale](#) (int scale)
Sets the music scale to the given value, and then resets the view.
- void [set_chord](#) (int chord)
Sets the current chord to the given value.
- void [set_data_type](#) (midibyte status, midibyte control)
Sets the status to the given parameter, and the CC value to the given optional control parameter, which defaults to 0.
- void [set_background_sequence](#) (bool state, int seq)
This function sets the given sequence onto the piano roll of the pattern editor, so that the musician can have another pattern to play against.
- void [update_pixmap](#) ()
This function draws the background pixmap on the main pixmap, and then draws the events on it.
- void [update_sizes](#) ()
Update the sizes of items based on zoom, PPQN, BPM, BW (beat width) and more.
- void [update_background](#) ()
Updates the background of this window.
- void [draw_background_on_pixmap](#) ()
Draws the main pixmap.
- void [draw_events_on_pixmap](#) ()
Fills the main pixmap with events.
- void [draw_selection_on_window](#) ()
Draws the current selection on the main window.
- void [draw_progress_on_window](#) ()
Draw a progress line on the window.
- void [reset](#) ()
This function basically resets the whole widget as if it were realized again.
- void [update_and_draw](#) (int force=false)
Wraps up some common code.
- void [redraw](#) ()
Redraws unless `m_ignore_redraw` is true.
- void [redraw_events](#) ()
Redraws events unless `m_ignore_redraw` is true.
- void [start_paste](#) ()
Starts a paste operation.
- void [complete_paste](#) ()
- void [complete_paste](#) (int x, int y)
Completes a paste operation.
- void [follow_progress](#) ()
Checks the position of the tick, and, if it is in a different piano-roll "page" than the last page, moves the page to the next page.

- void [set_expanded_recording](#) (bool expand)
'Getter' function for member m_expanded_recording
- bool [get_expanded_record](#) ()
'Setter' function for member m_expanded_recording
- void [set_progress_follow](#) (bool follow)
'Getter' function for member m_progress_follow
- bool [get_progress_follow](#) ()
'Setter' function for member m_progress_follow

Protected Member Functions

- virtual void [force_draw](#) ()
Set the pixmap into the window and then draws the selection on it.
- void [horizontal_adjust](#) (double step)
This function provides optimization for the [on_scroll_event\(\)](#) function.
- void [vertical_adjust](#) (double step)
This function provides optimization for the [on_scroll_event\(\)](#) function.
- void [snap_y](#) (int &y)
Snaps the y value to the piano-key "height".
- void [snap_x](#) (int &x)
Performs a 'snap' operation on the x coordinate.
- void [convert_xy](#) (int x, int y, [midipulse](#) &tick, int ¬e)
- void [convert_drop_xy](#) ([midipulse](#) &tick, int ¬e)
Convenience function that calls [convert_xy\(\)](#) for the drop and y values.
- void [convert_tn](#) ([midipulse](#) tick, int note, int &x, int &y)
This function takes the given note and tick, and returns the screen coordinates via the pointer parameters.
- void [xy_to_rect](#) (int x1, int y1, int x2, int y2, int &x, int &y, int &w, int &h)
- void [convert_tn_box_to_rect](#) ([midipulse](#) tick_s, [midipulse](#) tick_f, int note_h, int note_l, int &x, int &y, int &w, int &h)
Converts a tick/note box to an x/y rectangle.
- void [convert_tn_box_to_rect](#) ([midipulse](#) tick_s, [midipulse](#) tick_f, int note_h, int note_l, [rect](#) &r)
Converts a tick/note box to an x/y rectangle.
- void [convert_sel_box_to_rect](#) ([midipulse](#) tick_s, [midipulse](#) tick_f, int note_h, int note_l)
A convenience function wrapping a common call to [convert_tn_box_to_rect\(\)](#).
- void [get_selected_box](#) ([midipulse](#) &tick_s, int ¬e_h, [midipulse](#) &tick_f, int ¬e_l)
A convenience function wrapping a common call to [m_seq.get_selected_box\(\)](#) and [convert_tn_box_to_rect\(\)](#).
- void [draw_events_on](#) (Glib::RefPtr< Gdk::Drawable > draw)
Draws events on the given drawable area.
- int [idle_redraw](#) ()
Draw the events on the main window and on the pixmap.
- int [idle_progress](#) ()
- void [change_horz](#) ()
Change the horizontal scrolling offset and redraw.
- void [change_vert](#) ()
Change the vertical scrolling offset and redraw.
- void [move_selection_box](#) (int dx, int dy)
Function to allow motion of the selection box via the arrow keys.
- void [move_selected_notes](#) (int dx, int dy)
Proposed new function to encapsulate the movement of selections even more fully.
- void [grow_selected_notes](#) (int dx)

- Proposed new function to encapsulate the movement of selections even more fully.*

 - void `set_adding` (bool `adding`)

Changes the mouse cursor pixmap according to whether a note is being added or not.
- void `update_mouse_pointer` (bool `adding`=false)

Updates the mouse pointer, implementing a context-sensitive mouse.
- bool `button_press_initial` (GdkEventButton *ev, int &norm_x, int &snapped_x, int &snapped_y)
- void `align_selection` (midipulse &tick_s, int ¬e_h, midipulse &tick_f, int ¬e_l, int snapped_x)

Get the box that selected elements are in.
- bool `button_press` (GdkEventButton *ev)

Implements the on-button-press event handling for the Seq24 style of mouse interaction.
- bool `button_release` (GdkEventButton *ev)

Implements the on-button-release event handling for the Seq24 style of mouse interaction.
- bool `motion_notify` (GdkEventMotion *ev)

Seq24-style on-motion mouse interaction.
- void `clear_flags` ()

Clears all the mouse-action flags.
- void `set_scroll_x` ()

Sets the horizontal scroll value according to the current value of the horizontal scroll-bar.
- void `set_scroll_y` ()

Sets the vertical scroll value according to the current value of the vertical scroll-bar.
- int `scroll_offset_x` (int x) const

Useful x calculation.
- int `scroll_offset_y` (int y) const

Useful y calculation.
- void `set_current_offset_x_y` (int x, int y)

Useful x and y calculation.
- bool `adding` () const

'Getter' function for member m_adding
- bool `selecting` () const

'Getter' function for member m_selecting
- bool `growing` () const

'Getter' function for member m_growing
- bool `normal_action` () const

Indicates if we're drag-pasting, selecting, moving, growing, or pasting.
- bool `select_action` () const

Indicates if we're selecting, moving, growing, or pasting.
- bool `drop_action` () const

Indicates if we're moving or pasting.
- bool `moving` () const

'Getter' function for member m_moving
- virtual void `on_realize` ()

Implements the on-realize event handling.
- virtual bool `on_expose_event` (GdkEventExpose *ev)

Implements the on-expose event handling.
- virtual bool `on_button_press_event` (GdkEventButton *ev)

Implements the on-button-press event handling.
- virtual bool `on_button_release_event` (GdkEventButton *ev)

Implements the on-button-release event handling.
- virtual bool `on_motion_notify_event` (GdkEventMotion *ev)

Implements the on-motion-notify event handling.
- virtual bool `on_focus_in_event` (GdkEventFocus *)

- Implements the on-focus event handling.*
- virtual bool [on_focus_out_event](#) (GdkEventFocus *)
Implements the on-unfocus event handling.
- virtual bool [on_key_press_event](#) (GdkEventKey *ev)
Implements the on-key-press event handling.
- virtual bool [on_scroll_event](#) (GdkEventScroll *a_ev)
Implements the on-scroll event handling.
- virtual void [on_size_allocate](#) (Gtk::Allocation &)
Implements the on-size-allocate event handling.
- virtual bool [on_leave_notify_event](#) (GdkEventCrossing *p0)
Implements the on-leave-notify event handling.
- virtual bool [on_enter_notify_event](#) (GdkEventCrossing *p0)
Implements the on-enter-notify event handling.

Protected Attributes

- Gtk::Adjustment & [m_horizontal_adjust](#)
For accessing [on_key_press_event\(\)](#).
- Gtk::Adjustment & [m_vertical_adjust](#)
We need direct access to the vertical scrollbar if we want to be able to make it follow PageUp and PageDown.
- [rect m_old](#)
The previous selection rectangle, used for undrawing it.
- [rect m_selected](#)
Used in moving and pasting notes.
- [sequence](#) & [m_seq](#)
Provides a reference to the sequence represented by piano roll.
- [seqkeys](#) & [m_seqkeys_wid](#)
Holds a reference to the seqkeys pane that is associated with the seqroll piano roll.
- int [m_pos](#)
A position value.
- int [m_zoom](#)
Zoom setting, means that one pixel == [m_zoom](#) ticks.
- int [m_snap](#)
The grid-snap setting for the piano roll grid.
- int [m_note_length](#)
Holds the note length in force for this sequence.
- int [m_scale](#)
Indicates the musical scale in force for this sequence.
- int [m_chord](#)
Indicates the current chord in force for this sequence for inserting notes.
- int [m_key](#)
Indicates the musical key in force for this sequence.
- bool [m_adding](#)
Set when in note-adding mode.
- bool [m_selecting](#)
Set when highlighting a bunch of events.
- bool [m_moving](#)
Set when moving a bunch of events.
- bool [m_moving_init](#)
Indicates the beginning of moving some events.

- bool [m_growing](#)
Indicates that the notes are to be extended or reduced in length.
- bool [m_painting](#)
Indicates the painting of events.
- bool [m_paste](#)
Indicates that we are in the process of pasting notes.
- bool [m_is_drag_pasting](#)
Indicates the drag-pasting of events.
- bool [m_is_drag_pasting_start](#)
Indicates the drag-pasting of events.
- bool [m_justselected_one](#)
Indicates the selection of one event.
- int [m_move_delta_x](#)
Tells where the dragging started, the x value.
- int [m_move_delta_y](#)
Tells where the dragging started, the y value.
- int [m_move_snap_offset_x](#)
This item is used in the fruityseqroll module.
- int [m_progress_x](#)
Provides the location of the progress bar.
- int [m_scroll_offset_ticks](#)
The horizontal value of the scroll window in units of ticks/pulses/divisions.
- int [m_scroll_offset_key](#)
The vertical offset of the scroll window in units of MIDI notes/keys.
- int [m_scroll_offset_x](#)
The horizontal value of the scroll window in units of pixels.
- int [m_scroll_offset_y](#)
The vertical value of the scroll window in units of pixels.
- int [m_scroll_page](#)
Provides the current scroll page in which the progress bar resides.
- bool [m_progress_follow](#)
Progress bar follow state.
- bool [m_transport_follow](#)
Indicates if we are going to follow the transport in the GUI.
- bool [m_trans_button_press](#)
TBD.
- int [m_background_sequence](#)
Holds the value of the musical background sequence that is shown in cyan (formerly grey) on the background of the piano roll.
- bool [m_drawing_background_seq](#)
Set to true if the drawing of the background sequence is to be done.
- bool [m_expanded_recording](#)
Provides an option for expanding the number of measures while recording.
- midibyte [m_status](#)
The current status/event selected in the seqedit.
- midibyte [m_cc](#)
The current MIDI control value selected in the seqedit.

Additional Inherited Members

10.89.1 Constructor & Destructor Documentation

10.89.1.1 seqroll()

```
seq64::seqroll::seqroll (
    perform & p,
    sequence & seq,
    int zoom,
    int snap,
    seqkeys & seqkeys_wid,
    int pos,
    Gtk::Adjustment & hadjust,
    Gtk::Adjustment & vadjust )
```

Parameters

<i>p</i>	The performance object that helps control this piano roll. Note that we can get the perform object from the sequence, and save a parameter. Low priority change.
<i>seq</i>	The sequence object represented by this piano roll.
<i>zoom</i>	The initial zoom of this piano roll.
<i>snap</i>	The initial grid snap of this piano roll.
<i>seqkeys_wid</i>	A reference to the piano keys window that is shown to the left of this piano roll.
<i>pos</i>	A position parameter. See the description of seqroll::m_pos . This is actually the sequence number, and is currently unused. However, we're sure we can find a use for it sometime.
<i>hadjust</i>	Represents the horizontal scrollbar of this window. It is actually created by the "parent" seqedit object.
<i>vadjust</i>	Represents the vertical scrollbar of this window. It is actually created by the "parent" seqedit object.

10.89.1.2 ~seqroll()

```
seq64::seqroll::~~seqroll ( ) [virtual]
```

The only thing to delete here is the clipboard. Except it is never used, so is commented out.

10.89.2 Member Function Documentation

10.89.2.1 set_snap()

```
void seq64::seqroll::set_snap (
    int snap ) [inline]
```

Parameters

<i>snap</i>	Provides the sname value to set.
-------------	----------------------------------

10.89.2.2 `set_zoom()`

```
void seq64::seqroll::set_zoom (
    int zoom )
```

Parameters

<i>zoom</i>	The desired zoom value, assumed to be validated already. See the seqedit::set_zoom() function.
-------------	--

10.89.2.3 `set_note_length()`

```
void seq64::seqroll::set_note_length (
    int note_length ) [inline]
```

10.89.2.4 `note_off_length()`

```
int seq64::seqroll::note_off_length ( ) const [inline]
```

10.89.2.5 `add_note()`

```
bool seq64::seqroll::add_note (
    midipulse tick,
    int note,
    bool paint = true )
```

The length parameters is obtained from the [note_off_length\(\)](#) function. This sets the note length at a little less than the snap value.

Parameters

<i>tick</i>	The time destination of the new note, in pulses.
<i>note</i>	The pitch destination of the new note.
<i>paint</i>	If true, repaint to be left with just the inserted event. The default is true. The value of false is useful in inserting a number of events and saving the repainting until last. It is a bit tricky, as the default paint value for sequence::add_note() is false.

10.89.2.6 set_key()

```
void seq64::seqroll::set_key (
    int key )
```

Parameters

<i>key</i>	The desired key value.
------------	------------------------

10.89.2.7 set_scale()

```
void seq64::seqroll::set_scale (
    int scale )
```

Parameters

<i>scale</i>	The desired scale value.
--------------	--------------------------

10.89.2.8 set_chord()

```
void seq64::seqroll::set_chord (
    int chord )
```

Parameters

<i>chord</i>	The desired chord value.
--------------	--------------------------

10.89.2.9 set_data_type()

```
void seq64::seqroll::set_data_type (
    midibyte status,
    midibyte control ) [inline]
```

Unlike the same function in sequevent, this version does not redraw. Used by seqedit.

10.89.2.10 set_background_sequence()

```
void seq64::seqroll::set_background_sequence (
    bool state,
    int seq )
```

The state parameter sets the boolean m_drawing_background_seq.

Parameters

<i>state</i>	If true, the background sequence will be drawn.
<i>seq</i>	Provides the sequence number, which is checked against the SEQ64_IS_LEGAL_SEQUENCE() macro before being used. This macro allows the value SEQ64_SEQUENCE_LIMIT, which disables the background sequence.

10.89.2.11 update_pixmap()

```
void seq64::seqroll::update_pixmap ( )
```

10.89.2.12 update_sizes()

```
void seq64::seqroll::update_sizes ( )
```

It brings the scrollbar back to the beginning, resets the upper limit to the number of ticks in the sequence, sets the page-size based on the window size and the zoom factor.

The horizontal step increment is 1 semiquaver (1/16) note per zoom level. The horizontal page increment is currently always one bar. We may want to make that larger for scrolling after the progress bar.

The maximum value set for the scrollbar brings it to the last "page" of the piano roll.

The vertical size are also adjusted. More on the story later.

10.89.2.13 update_background()

```
void seq64::seqroll::update_background ( )
```

The first thing done is to clear the background, painting it white.

10.89.2.14 draw_background_on_pixmap()

```
void seq64::seqroll::draw_background_on_pixmap ( )
```

10.89.2.15 draw_events_on_pixmap()

```
void seq64::seqroll::draw_events_on_pixmap ( )
```

Just calls [draw_events_on\(\)](#).

10.89.2.16 draw_selection_on_window()

```
void seq64::seqroll::draw_selection_on_window ( )
```

Note the parameters of [draw_drawable\(\)](#), which we need to be sure of to draw thicker boxes.

- x and y position of rectangle to draw
- x and y position in drawable where rectangle should be drawn
- width and height of rectangle to draw

A final parameter of false draws an unfilled rectangle. Orange makes it a little more clear that we're pasting, I think. We also want to try to thicken the lines somehow.

10.89.2.17 draw_progress_on_window()

```
void seq64::seqroll::draw_progress_on_window ( )
```

This is done by first blanking out the line with the background, which contains white space and grey lines, using the [draw_drawable](#) function. Remember that we wrap the [draw_drawable\(\)](#) function so it's parameters are xsrc, ysrc, xdest, ydest, width, and height.

Note that the progress-bar position is based on the [sequence::get_last_tick\(\)](#) value, the current zoom, and the current scroll-offset x value.

Finally, we had an issue with the selection box flickering, which seems to be solved satisfactorily by not drawing it if a select action is in force. Hopefully no one needs to select notes on the fly and see the progress bar moving at the same time! Another tactic is to draw progress only when the performance is running. This has the benefit/drawback that the progress bar is left where it stops. Consider an enumeration of options: normal, when-not-selecting, and when-running.

10.89.2.18 reset()

```
void seq64::seqroll::reset ( )
```

It's almost identical to the [change_horz\(\)](#) function, just calling [update_sizes\(\)](#) before [update_and_draw\(\)](#).

10.89.2.19 update_and_draw()

```
void seq64::seqroll::update_and_draw (
    int force = false )
```

Parameters

<i>force</i>	If true, force an immediate draw, otherwise just queue up a draw. This value defaults to false.
--------------	---

10.89.2.20 redraw()

```
void seq64::seqroll::redraw ( )
```

Somewhat similar to [sequevent::redraw\(\)](#). Actually, we don't seem to need to ignore redraw when making settings in the seqedit constructor, so this member no longer exists.

10.89.2.21 redraw_events()

```
void seq64::seqroll::redraw_events ( )
```

Actually, that member is not needed and no longer exists.

10.89.2.22 start_paste()

```
void seq64::seqroll::start_paste ( )
```

10.89.2.23 complete_paste() [1/2]

```
void seq64::seqroll::complete_paste ( ) [inline]
```

10.89.2.24 complete_paste() [2/2]

```
void seq64::seqroll::complete_paste (
    int x,
    int y )
```

10.89.2.25 follow_progress()

```
void seq64::seqroll::follow_progress ( )
```

We don't want to do any of this if the length of the sequence fits in the window, but for now it doesn't hurt; the progress bar just never meets the criterion for moving to the next page.

- Todo**
- If playback is disabled (such as by a trigger), then do not update the page;
 - When it comes back, make sure we're on the correct page;
 - When it stops, put the window back to the beginning, even if the beginning is not defined as "0".

10.89.2.26 set_expanded_recording()

```
void seq64::seqroll::set_expanded_recording (
    bool expand ) [inline]
```

10.89.2.27 get_expanded_record()

```
bool seq64::seqroll::get_expanded_record ( ) [inline]
```

10.89.2.28 set_progress_follow()

```
void seq64::seqroll::set_progress_follow (
    bool follow ) [inline]
```

10.89.2.29 get_progress_follow()

```
bool seq64::seqroll::get_progress_follow ( ) [inline]
```

10.89.2.30 force_draw()

```
void seq64::seqroll::force_draw ( ) [protected], [virtual]
```

Reimplemented from [seq64::gui_drawingarea_gtk2](#).

10.89.2.31 horizontal_adjust()

```
void seq64::seqroll::horizontal_adjust (
    double step ) [inline], [protected]
```

A duplicate of the one in `seqedit`.

Parameters

<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. See gui_drawingarea_gtk2::scroll_hadjust() for more information.
-------------	---

10.89.2.32 vertical_adjust()

```
void seq64::seqroll::vertical_adjust (
    double step ) [inline], [protected]
```

A duplicate of the one in `seqedit`.

Parameters

step	Provides the step value to use for adjusting the vertical scrollbar. See gui_drawingarea_gtk2::scroll_vadjust() for more information.
------	---

10.89.2.33 snap_y()

```
void seq64::seqroll::snap_y (
    int & y ) [inline], [protected]
```

Parameters

out	y	The y-value to be snapped.
-----	---	----------------------------

10.89.2.34 snap_x()

```
void seq64::seqroll::snap_x (
    int & x ) [protected]
```

This function is similar to [snap_y\(\)](#), but it calculates a modulo value from the snap and zoom settings.

- m_snap = number pulses to snap to
- m_zoom = number of pulses per pixel

Therefore, m_snap / m_zoom = number pixels to snap to.

Parameters

out	x	Provides the x-value to be snapped and returned. A return value would be better.
-----	---	--

10.89.2.35 convert_xy()

```
void seq64::seqroll::convert_xy (
    int x,
    int y,
    midipulse & tick,
    int & note ) [protected]
```

10.89.2.36 convert_drop_xy()

```
void seq64::seqroll::convert_drop_xy (
    midipulse & tick,
    int & note ) [inline], [protected]
```

Parameters

<i>tick</i>	The horizontal location of the drop.
<i>note</i>	The vertical location of the drop.

10.89.2.37 convert_tn()

```
void seq64::seqroll::convert_tn (
    midipulse tick,
    int note,
    int & x,
    int & y ) [protected]
```

This function is the "inverse" of [convert_xy\(\)](#).

Parameters

	<i>tick</i>	Provides the horizontal value in MIDI pulses.
	<i>note</i>	Provides the vertical value, a note value.
out	<i>x</i>	Provides the destination x value of the coordinate.
out	<i>y</i>	Provides the destination y value of the coordinate.

10.89.2.38 xy_to_rect()

```
void seq64::seqroll::xy_to_rect (
    int x1,
    int y1,
    int x2,
    int y2,
    int & x,
    int & y,
    int & w,
    int & h ) [protected]
```

10.89.2.39 convert_tn_box_to_rect() [1/2]

```
void seq64::seqroll::convert_tn_box_to_rect (
    midipulse tick_s,
```

```

midipulse tick_f,
int note_h,
int note_l,
int & x,
int & y,
int & w,
int & h ) [protected]

```

We should refactor this function to use the utility class seqroll::rect as the destination for the conversion.

Parameters

	<i>tick_s</i>	The starting tick of the rectangle.
	<i>tick_f</i>	The finishing tick of the rectangle.
	<i>note</i> ↔ <i>_h</i>	The high note of the rectangle.
	<i>note</i> ↔ <i>_l</i>	The low note of the rectangle.
out	<i>x</i>	The destination for the x value in pixels.
out	<i>y</i>	The destination for the y value in pixels.
out	<i>w</i>	The destination for the rectangle width in pixels.
out	<i>h</i>	The destination for the rectangle height value in pixels.

10.89.2.40 convert_tn_box_to_rect() [2/2]

```

void seq64::seqroll::convert_tn_box_to_rect (
    midipulse tick_s,
    midipulse tick_f,
    int note_h,
    int note_l,
    rect & r ) [protected]

```

We should refactor this function to use the utility class seqroll::rect as the destination for the conversion.

Parameters

	<i>tick_s</i>	The starting tick of the rectangle.
	<i>tick_f</i>	The finishing tick of the rectangle.
	<i>note</i> ↔ <i>_h</i>	The high note of the rectangle.
	<i>note</i> ↔ <i>_l</i>	The low note of the rectangle.
out	<i>r</i>	The destination rectangle for the values in pixels.

10.89.2.41 convert_sel_box_to_rect()

```

void seq64::seqroll::convert_sel_box_to_rect (
    midipulse tick_s,

```

```

midipulse tick_f,
int note_h,
int note_l ) [protected]

```

We made a version taking a reference to an [seq64::rect](#).

Parameters

<i>tick_s</i>	The starting tick of the rectangle.
<i>tick_f</i>	The finishing tick of the rectangle.
<i>note</i> ↔ <i>_h</i>	The high note of the rectangle.
<i>note</i> ↔ <i>_l</i>	The low note of the rectangle.

10.89.2.42 get_selected_box()

```

void seq64::seqroll::get_selected_box (
    midipulse & tick_s,
    int & note_h,
    midipulse & tick_f,
    int & note_l ) [protected]

```

Parameters

out	<i>tick_s</i>	The starting tick of the rectangle.
out	<i>tick_f</i>	The finishing tick of the rectangle.
out	<i>note</i> ↔ <i>_h</i>	The high note of the rectangle.
out	<i>note</i> ↔ <i>_l</i>	The low note of the rectangle.

10.89.2.43 draw_events_on()

```

void seq64::seqroll::draw_events_on (
    Glib::RefPtr< Gdk::Drawable > draw ) [protected]

```

"Method 0" draws the background sequence, if active. "Method 1" draws the sequence itself.

Parameters

<i>draw</i>	The "drawable" area to draw on.
-------------	---------------------------------

10.89.2.44 idle_redraw()

```
int seq64::seqroll::idle_redraw ( ) [protected]
```

10.89.2.45 idle_progress()

```
int seq64::seqroll::idle_progress ( ) [protected]
```

10.89.2.46 change_horz()

```
void seq64::seqroll::change_horz ( ) [protected]
```

Roughly similar to [seqevent::change_horz\(\)](#).

10.89.2.47 change_vert()

```
void seq64::seqroll::change_vert ( ) [protected]
```

10.89.2.48 move_selection_box()

```
void seq64::seqroll::move_selection_box (
    int dx,
    int dy ) [protected]
```

We now let the Enter key to finish pasting and deselect the moved notes. With the mouse, selecting all notes, copying them, and moving the selection box, pasting can be completed with either a left-click or the Enter key.

We have a weird problem on our main system where the selection box is very flickery. But it works fine on another system. A Gtk-2 issue? Now it seems to work fine, after an update. No, it seems to work well in sequences that have non-note events amongst the note events.

Parameters

<i>dx</i>	The amount to move the selection box. Values are -1, 0, or 1. -1 is left one snap, 0 is no movement horizontally, and 1 is right one snap.
<i>dy</i>	The amount to move the selection box. Values are -1, 0, or 1. -1 is up one snap, 0 is no movement vertically, and 1 is down one snap.

10.89.2.49 move_selected_notes()

```
void seq64::seqroll::move_selected_notes (
    int dx,
    int dy ) [protected]
```

Works with the four arrow keys.

Note that the movement vertically is different for the selection box versus the notes. While the movement values are -1, 0, or 1, the differences are as follows:

- Selection box vertical movement:
 - -1 is up one note snap.
 - 0 is no vertical movement.
 - +1 is down one note snap.
- Note vertical movement:
 - -1 is down one note.
 - 0 is no note vertical movement.
 - +1 is up one note.

Parameters

<i>dx</i>	The amount to move the selection box or the selection horizontally. Values are -1 (left one time snap), 0 (no movement), and +1 (right one snap). Obviously values other than +-1 can be used for larger movement, but the GUI doesn't yet support that ... we could implement movement by "pages" some day.
<i>dy</i>	The amount to move the selection box or the selection vertically. See the notes above.

10.89.2.50 grow_selected_notes()

```
void seq64::seqroll::grow_selected_notes (
    int dx ) [protected]
```

Parameters

<i>dx</i>	The amount to grow the selection horizontally. Values are -1 (left one time snap), 0 (no stretching), and +1 (right one snap). Obviously values other than +-1 can be used for larger stretching, but the GUI doesn't yet support that.
-----------	---

10.89.2.51 set_adding()

```
void seq64::seqroll::set_adding (
    bool isadding ) [protected]
```

What calls this? It is actually a right click. Not present in the "fruity" implementation. Now moved to the normal seqroll class.

Parameters

<i>adding</i>	True if adding a note.
---------------	------------------------

10.89.2.52 update_mouse_pointer()

```
void seq64::seqroll::update_mouse_pointer (
    bool isadding = false ) [protected]
```

Moved here from the "fruity" seqroll class.

10.89.2.53 button_press_initial()

```
bool seq64::seqroll::button_press_initial (
    GdkEventButton * ev,
    int & norm_x,
    int & snapped_x,
    int & snapped_y ) [protected]
```

10.89.2.54 align_selection()

```
void seq64::seqroll::align_selection (
    midipulse & tick_s,
    int & note_h,
    midipulse & tick_f,
    int & note_l,
    int snapped_x ) [protected]
```

Save offset that we get from the snap above. Align selection for drawing. Could be used in XRollInput::on_button↔_press_event().

10.89.2.55 button_press()

```
bool seq64::seqroll::button_press (
    GdkEventButton * ev ) [protected]
```

This function now uses the needs_update flag to determine if the perform object should modify().

Parameters

<i>ev</i>	Provides the button-press event to process.
-----------	---

Returns

Returns the value of needs_update. It used to return only true.

10.89.2.56 button_release()

```
bool seq64::seggroll::button_release (
    GdkEventButton * ev ) [protected]
```

This function now uses the needs_update flag to determine if the perform object should modify().

Parameters

ev	Provides the button-release event to process.
----	---

Returns

Returns the value of needs_update. It used to return only true.

If in moving mode, adjust for snap and convert deltas into screen coordinates. Since delta_note was from delta_y, it will be flipped (delta_y[0] = note[127], etc.), so we have to adjust.

A left/middle click converts deltas into screen coordinates, then pushes the undo state. Shift causes a "stretch selected" which currently acts like a "move selected" operation. BUG? Otherwise, Ctrl indirectly allows a "grow selected" operation.

Minor new feature. If the Super (Mod4, Windows) key is pressed when release, keep the adding state in force. One can then use the unadorned left-click key to add notes. Right click to reset the adding mode. This feature is enabled only if allowed by the settings (but is true by default). See the same code in perfrollinput.cpp.

10.89.2.57 motion_notify()

```
bool seq64::seggroll::motion_notify (
    GdkEventMotion * ev ) [protected]
```

We could allow move painting for chords, but that would take some tricky code to move all of the notes of the chord. And allowing painting here currently affects only one note after the chord itself is created.

Parameters

ev	Provides the button-release event to process.
----	---

Returns

Returns true if the event was processed.

10.89.2.58 clear_flags()

```
void seq64::seqroll::clear_flags ( ) [inline], [protected]
```

10.89.2.59 set_scroll_x()

```
void seq64::seqroll::set_scroll_x ( ) [protected]
```

10.89.2.60 set_scroll_y()

```
void seq64::seqroll::set_scroll_y ( ) [protected]
```

10.89.2.61 scroll_offset_x()

```
int seq64::seqroll::scroll_offset_x (
    int x ) const [inline], [protected]
```

Offsets the x value by the x origin of the current page.

Parameters

<i>x</i>	The x value to offset.
----------	------------------------

10.89.2.62 scroll_offset_y()

```
int seq64::seqroll::scroll_offset_y (
    int y ) const [inline], [protected]
```

Offsets the y value by the y origin of the current page.

Parameters

<i>y</i>	The y value to offset.
----------	------------------------

10.89.2.63 set_current_offset_x_y()

```
void seq64::seqroll::set_current_offset_x_y (
```

```
int x,  
int y ) [inline], [protected]
```

Offsets the current x and y values by the x and y origin of the current page.

Parameters

x	The y value to offset.
y	The y value to offset.

10.89.2.64 adding()

```
bool seq64::seqroll::adding ( ) const [inline], [protected]
```

10.89.2.65 selecting()

```
bool seq64::seqroll::selecting ( ) const [inline], [protected]
```

10.89.2.66 growing()

```
bool seq64::seqroll::growing ( ) const [inline], [protected]
```

10.89.2.67 normal_action()

```
bool seq64::seqroll::normal_action ( ) const [inline], [protected]
```

Returns

Returns true if one of those five flags are set.

10.89.2.68 select_action()

```
bool seq64::seqroll::select_action ( ) const [inline], [protected]
```

Returns

Returns true if one of those four flags are set.

10.89.2.69 drop_action()

```
bool seq64::seqroll::drop_action ( ) const [inline], [protected]
```

Returns

Returns true if one of those two flags are set.

10.89.2.70 moving()

```
bool seq64::seqroll::moving ( ) const [inline], [protected]
```

10.89.2.71 on_realize()

```
void seq64::seqroll::on_realize ( ) [protected], [virtual]
```

10.89.2.72 on_expose_event()

```
bool seq64::seqroll::on_expose_event (
    GdkEventExpose * ev ) [protected], [virtual]
```

Parameters

<i>ev</i>	The expose event to process.
-----------	------------------------------

Returns

Always returns true.

10.89.2.73 on_button_press_event()

```
bool seq64::seqroll::on_button_press_event (
    GdkEventButton * ev ) [protected], [virtual]
```

Parameters

<i>ev</i>	The expose event to process.
-----------	------------------------------

Returns

Returns the result of the Seq24 interaction or the Fruity interaction, that is, the return value of Seq24SeqRollInput::on_button_press_event() or FruitySeqRollInput::on_button_press_event().

Reimplemented in [seq64::FruitySeqRollInput](#).

10.89.2.74 on_button_release_event()

```
bool seq64::seqroll::on_button_release_event (
    GdkEventButton * ev ) [protected], [virtual]
```

This function checks the "rc" interaction-method option, and calls the forwarding function for the seq24 or the fruity interaction method. Might be a good case to prefer inheritance and not try to support changing the interaction method without a restart of Sequencer64.

Parameters

<i>ev</i>	The button release event to process.
-----------	--------------------------------------

Returns

Returns the return value of Seq24SeqRollInput::on_button_release_event() or FruitySeqRollInput::on_button_release_event().

Reimplemented in [seq64::FruitySeqRollInput](#).

10.89.2.75 on_motion_notify_event()

```
bool seq64::seqroll::on_motion_notify_event (
    GdkEventMotion * ev ) [protected], [virtual]
```

Parameters

<i>ev</i>	The motion-notification event to process.
-----------	---

Returns

Returns the return value of Seq24SeqRollInput::on_motion_notify_event() or FruitySeqRollInput::on_motion_notify_event().

Reimplemented in [seq64::FruitySeqRollInput](#).

10.89.2.76 on_focus_in_event()

```
bool seq64::seqroll::on_focus_in_event (
    GdkEventFocus * ) [protected], [virtual]
```

Sets the GDK HAS_FOCUS flag. Parameter "ev" is the event-focus event, not used.

Returns

Always returns false.

10.89.2.77 on_focus_out_event()

```
bool seq64::seqroll::on_focus_out_event (
    GdkEventFocus * ) [protected], [virtual]
```

Resets the GDK HAS_FOCUS flag. Parameter "ev" is the event-focus event, not used.

Returns

Always returns false.

10.89.2.78 on_key_press_event()

```
bool seq64::seqroll::on_key_press_event (
    GdkEventKey * ev ) [protected], [virtual]
```

The start/end key may be the same key (i.e. SPACEBAR). Allow toggling when the same key is mapped to both triggers (i.e. SPACEBAR).

Concerning the usage of the arrow keys in this function: This code is reached, but has no visible effect. Why? I think they were meant to move the point for playback. We may have a bug with our new handling of triggers (unlikely), or maybe these depend upon the proper playback mode. In any case, the old functionality is preserved. However, if there are notes selected, then these keys support selection movement.

Since the Up and Down arrow keys are used for movement, we'd have to check selection status before trying to use them to move up and down in the piano roll, in smaller steps than the new Page-Up and Page-Down key support.

Parameters

ev	The key-press event to process.
----	---------------------------------

Returns

Returns true if the key-press was handled.

I think we should be able to move and remove notes while playing, which is already supported using the mouse.

```
if (! perf\(\).is_playing)
```

10.89.2.79 `on_scroll_event()`

```
bool seq64::seqroll::on_scroll_event (
    GdkEventScroll * ev ) [protected], [virtual]
```

This scroll event only handles basic scrolling without any modifier keys such as the Ctrl or Shift masks. The seqedit class handles that fun stuff.

Note that this function seems to duplicate the functionality of [seqkeys::on_scroll_event\(\)](#). Do we really need both? Which one do we need?

Parameters

<i>ev</i>	The scroll event to process.
-----------	------------------------------

Returns

Returns true if the scroll event was handled.

10.89.2.80 `on_size_allocate()`

```
void seq64::seqroll::on_size_allocate (
    Gtk::Allocation & a ) [protected], [virtual]
```

Calls the base-class version of this function and sets `m_window_x` and `m_window_y` to the width and height of the allocation parameter. Then calls [update_sizes\(\)](#).

Parameters

<i>a</i>	The GDK allocation event object.
----------	----------------------------------

10.89.2.81 `on_leave_notify_event()`

```
bool seq64::seqroll::on_leave_notify_event (
    GdkEventCrossing * p0 ) [protected], [virtual]
```

Calls `m_seqkeys_wid.set_hint_state(false)`. Parameter "ev" is the event-crossing event, not used.

Returns

Always returns false.

10.89.2.82 on_enter_notify_event()

```
bool seq64::seqroll::on_enter_notify_event (
    GdkEventCrossing * p0 ) [protected], [virtual]
```

Calls `m_seqkeys_wid.set_hint_state(true)`. Parameter "ev" is the event-crossing event, not used.

Returns

Always returns false.

10.89.3 Field Documentation

10.89.3.1 m_horizontal_adjust

```
Gtk::Adjustment& seq64::seqroll::m_horizontal_adjust [protected]
```

It would be good to be able to avoid this access!

Change Note layk 2016-10-17 Issue #46. No need for this declaration now, due to the fix in seqedit.

friend class seqedit; We need direct access to the horizontal scrollbar if we want to be able to make it follow the progress bar.

10.89.3.2 m_vertical_adjust

```
Gtk::Adjustment& seq64::seqroll::m_vertical_adjust [protected]
```

10.89.3.3 m_old

```
rect seq64::seqroll::m_old [protected]
```

10.89.3.4 m_selected

```
rect seq64::seqroll::m_selected [protected]
```

10.89.3.5 m_seq

```
sequence& seq64::seqroll::m_seq [protected]
```

10.89.3.6 m_seqkeys_wid

```
seqkeys& seq64::seqroll::m_seqkeys_wid [protected]
```

10.89.3.7 m_pos

```
int seq64::seqroll::m_pos [protected]
```

Need to clarify what exactly this member is used for.

10.89.3.8 m_zoom

```
int seq64::seqroll::m_zoom [protected]
```

10.89.3.9 m_snap

```
int seq64::seqroll::m_snap [protected]
```

Same meaning as for the event-bar grid. This value is the denominator of the note size used for the snap.

10.89.3.10 m_note_length

```
int seq64::seqroll::m_note_length [protected]
```

Used in the seq24seqroll module only.

10.89.3.11 m_scale

```
int seq64::seqroll::m_scale [protected]
```

10.89.3.12 m_chord

```
int seq64::seqroll::m_chord [protected]
```

10.89.3.13 m_key

```
int seq64::seqroll::m_key [protected]
```

10.89.3.14 m_adding

```
bool seq64::seqroll::m_adding [protected]
```

This flag was moved from both the fruity and the seq24 seqroll classes.

10.89.3.15 m_selecting

```
bool seq64::seqroll::m_selecting [protected]
```

10.89.3.16 m_moving

```
bool seq64::seqroll::m_moving [protected]
```

10.89.3.17 m_moving_init

```
bool seq64::seqroll::m_moving_init [protected]
```

Used in the fruity and seq24 mouse-handling modules.

10.89.3.18 m_growing

```
bool seq64::seqroll::m_growing [protected]
```

10.89.3.19 m_painting

```
bool seq64::seqroll::m_painting [protected]
```

Used in the fruity and seq24 mouse-handling modules.

10.89.3.20 m_paste

```
bool seq64::seqroll::m_paste [protected]
```

10.89.3.21 m_is_drag_pasting

```
bool seq64::seqroll::m_is_drag_pasting [protected]
```

Used in the fruity mouse-handling module.

10.89.3.22 m_is_drag_pasting_start

```
bool seq64::seqroll::m_is_drag_pasting_start [protected]
```

Used in the fruity mouse-handling module.

10.89.3.23 m_justselected_one

```
bool seq64::seqroll::m_justselected_one [protected]
```

Used in the fruity mouse-handling module.

10.89.3.24 m_move_delta_x

```
int seq64::seqroll::m_move_delta_x [protected]
```

10.89.3.25 m_move_delta_y

```
int seq64::seqroll::m_move_delta_y [protected]
```

10.89.3.26 m_move_snap_offset_x

```
int seq64::seqroll::m_move_snap_offset_x [protected]
```

10.89.3.27 m_progress_x

```
int seq64::seqroll::m_progress_x [protected]
```

10.89.3.28 m_scroll_offset_ticks

```
int seq64::seqroll::m_scroll_offset_ticks [protected]
```

10.89.3.29 m_scroll_offset_key

```
int seq64::seqroll::m_scroll_offset_key [protected]
```

10.89.3.30 m_scroll_offset_x

```
int seq64::seqroll::m_scroll_offset_x [protected]
```

10.89.3.31 m_scroll_offset_y

```
int seq64::seqroll::m_scroll_offset_y [protected]
```

10.89.3.32 m_scroll_page

```
int seq64::seqroll::m_scroll_page [protected]
```

10.89.3.33 m_progress_follow

```
bool seq64::seqroll::m_progress_follow [protected]
```

10.89.3.34 m_transport_follow

```
bool seq64::seqroll::m_transport_follow [protected]
```

Progress follow?

10.89.3.35 m_trans_button_press

```
bool seq64::seqroll::m_trans_button_press [protected]
```

10.89.3.36 m_background_sequence

```
int seq64::seqroll::m_background_sequence [protected]
```

10.89.3.37 m_drawing_background_seq

```
bool seq64::seqroll::m_drawing_background_seq [protected]
```

10.89.3.38 m_expanded_recording

```
bool seq64::seqroll::m_expanded_recording [protected]
```

In essence, the "infinite" track we've wanted, thanks to Stazed and his Seq32 project. Defaults to false.

10.89.3.39 m_status

```
midibyte seq64::seqroll::m_status [protected]
```

Not used in seqroll at present.

10.89.3.40 m_cc

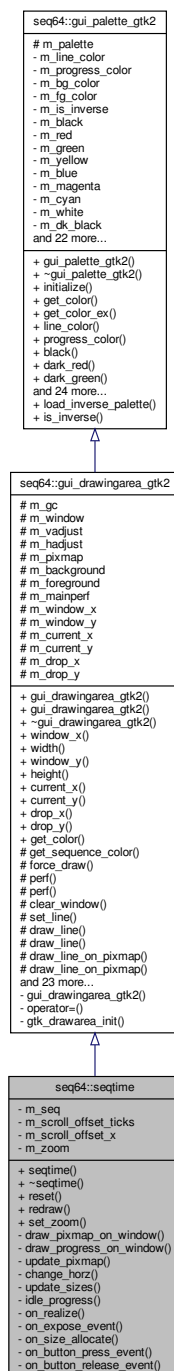
```
midibyte seq64::seqroll::m_cc [protected]
```

Not used in seqroll at present.

10.90 seq64::seqtime Class Reference

This class implements the piano time, whatever that is.

Inheritance diagram for seq64::seqtime:



Public Member Functions

- [seqtime](#) ([sequence](#) &seq, [perform](#) &p, int zoom, Gtk::Adjustment &hadjust)
Principal constructor.
- virtual [~seqtime](#) ()
Let's provide a do-nothing virtual destructor.
- void [reset](#) ()

Sets the scroll offset tick and x values, updates the sizes and the pixmap, and resets the window.

- void [redraw](#) ()

Very similar to the [reset\(\)](#) function, except it doesn't update the sizes.

- void [set_zoom](#) (int zoom)

Sets the zoom to the given value and resets the window.

Private Member Functions

- void [draw_pixmap_on_window](#) ()

Draws the pixmap on the window.

- void [draw_progress_on_window](#) ()

- void [update_pixmap](#) ()

Updates the pixmap.

- void [change_horz](#) ()

Changes the scrolling horizontal offset, updates the pixmap, and forces a redraw.

- void [update_sizes](#) ()

Updates the pixmap to a new size and queues up a draw operation.

- bool [idle_progress](#) ()

Simply returns true.

- void [on_realize](#) ()

Called when the window is drawn.

- bool [on_expose_event](#) (GdkEventExpose *a_ev)

Implements the on-expose event handler.

- void [on_size_allocate](#) (Gtk::Allocation &)

Implements the on-size-allocate event handler.

- bool [on_button_press_event](#) (GdkEventButton *)

Implements the on-button-press event handler.

- bool [on_button_release_event](#) (GdkEventButton *)

Implements the on-button-release event handler.

Private Attributes

- [sequence](#) & [m_seq](#)
- int [m_scroll_offset_ticks](#)
- int [m_scroll_offset_x](#)
- int [m_zoom](#)

one pixel == m_zoom ticks

Additional Inherited Members

10.90.1 Constructor & Destructor Documentation

10.90.1.1 seqtime()

```
seq64::seqtime::seqtime (
    sequence & seq,
    perform & p,
    int zoom,
    Gtk::Adjustment & hadjust )
```

In the constructor you can only allocate colors; `get_window()` returns 0 because the window is not yet realized>

10.90.1.2 ~seqtime()

```
virtual seq64::seqtime::~~seqtime ( ) [inline], [virtual]
```

10.90.2 Member Function Documentation

10.90.2.1 reset()

```
void seq64::seqtime::reset ( )
```

Basically identical to [sequevent::reset\(\)](#).

10.90.2.2 redraw()

```
void seq64::seqtime::redraw ( )
```

10.90.2.3 set_zoom()

```
void seq64::seqtime::set_zoom (
    int zoom )
```

Parameters

<i>zoom</i>	The desired zoom value, assumed to be validated already. See the seqedit::set_zoom() function.
-------------	--

10.90.2.4 draw_pixmap_on_window()

```
void seq64::seqtime::draw_pixmap_on_window ( ) [private]
```

10.90.2.5 draw_progress_on_window()

```
void seq64::seqtime::draw_progress_on_window ( ) [private]
```

10.90.2.6 update_pixmap()

```
void seq64::seqtime::update_pixmap ( ) [private]
```

When the zoom is at 32 ticks per pixel, there is a thick bar for every measure, and a measure number and major time division every 4 measures.at the default PPQN of 192.

A major line is a line that has a measure number in the timeline. The number of measures in a major line is 1 for zooms from 1:1 to 1:8; 2 for zoom 1:16; 4 for zoom 1:32; 8 for zoom 1:64 (new); and 16 for zoom 1:128. Zooms 1:64 and above look good only for high PPQN values.

We calculate the measure length in 32nd notes. This value is, of course, 32, when the time signature is 4/4. Then calculate measures/line. "measures_per_major" is more like "measures per major line". With a higher zoom than 32, this calculation yields a floating-point exception if m_zoom

32, so we rearrange the calculation and hope that it still works out the

same for smaller values.

Stazed:

At 32, a bar every measure.

zoom	32	16	8	4	1
ml					
1	128				
2	64				
4	32	16	8		
8	16m	8	4	2	1
16	8m	4	2	1	1
32	4m	2	1	1	1
64	2m	1	1	1	1
128	1m	1	1	1	1

Todo Sizing needs to be controlled by font parameters. Instead of 19 or 20, estimate the width of 3 letters. Instead of 9 pixels down, use the height of the seqtime and the height of a character.

10.90.2.7 change_horz()

```
void seq64::seqtime::change_horz ( ) [private]
```

10.90.2.8 update_sizes()

```
void seq64::seqtime::update_sizes ( ) [private]
```

10.90.2.9 idle_progress()

```
bool seq64::seqtime::idle_progress ( ) [inline], [private]
```

10.90.2.10 on_realize()

```
void seq64::seqtime::on_realize ( ) [private]
```

Call the base-class version of this function first. Then addition resources are allocated.

10.90.2.11 on_expose_event()

```
bool seq64::seqtime::on_expose_event (
    GdkEventExpose * a_ev ) [private]
```

10.90.2.12 on_size_allocate()

```
void seq64::seqtime::on_size_allocate (
    Gtk::Allocation & a ) [private]
```

10.90.2.13 on_button_press_event()

```
bool seq64::seqtime::on_button_press_event (
    GdkEventButton * ) [inline], [private]
```

Simply returns false.

10.90.2.14 on_button_release_event()

```
bool seq64::seqtime::on_button_release_event (
    GdkEventButton * ) [inline], [private]
```

Simply returns false.

10.90.3 Field Documentation

10.90.3.1 m_seq

```
sequence& seq64::seqtime::m_seq [private]
```

10.90.3.2 m_scroll_offset_ticks

```
int seq64::seqtime::m_scroll_offset_ticks [private]
```

10.90.3.3 m_scroll_offset_x

```
int seq64::seqtime::m_scroll_offset_x [private]
```

10.90.3.4 m_zoom

```
int seq64::seqtime::m_zoom [private]
```

10.91 seq64::sequence Class Reference

The sequence class is firstly a receptable for a single track of MIDI data read from a MIDI file or edited into a pattern.

Public Types

- enum `select_action_t` {
 `e_select`,
 `e_select_one`,
 `e_is_selected`,
 `e_would_select`,
 `e_deselect`,
 `e_toggle_selection`,
 `e_remove_one`,
 `e_select_onset`,
 `e_is_selected_onset` }

This enumeration is used in selecting events and note.

Public Member Functions

- [sequence](#) (int ppqn=SEQ64_USE_DEFAULT_PPQN)
Principal constructor.
- [~sequence](#) ()
A rote destructor.
- void [partial_assign](#) (const [sequence](#) &rhs)
A cut-down version of principal assignment operator.
- [event_list](#) & [events](#) ()
'Getter' function for member m_events Non-const version.
- const [event_list](#) & [events](#) () const
'Getter' function for member m_events Const version.
- bool [any_selected_notes](#) () const
'Getter' function for member m_events.any_selected_notes()
- const [triggers::List](#) & [triggerlist](#) () const
'Getter' function for member m_triggers This is the const version.
- [triggers::List](#) & [triggerlist](#) ()
'Getter' function for member m_triggers
- int [trigger_count](#) () const
Gets the trigger count, useful for exporting a sequence.
- int [selected_trigger_count](#) () const
Gets the number of selected triggers.
- void [set_trigger_paste_tick](#) (midipulse tick)
Sets the tick for pasting.
- [midipulse](#) [get_trigger_paste_tick](#) () const
Gets the tick for pasting.
- std::string [seq_number](#) () const
'Getter' function for member m_seq_number as a string
- int [number](#) () const
'Getter' function for member m_seq_number
- void [number](#) (int seqnum)
'Setter' function for member m_seq_number This setter will set the sequence number only if it has not already been set.
- int [color](#) () const
'Getter' function for member m_seq_color
- void [color](#) (int c)
'Setter' function for member m_seq_color Provides the color-palette value to set.
- [edit_mode_t](#) [edit_mode](#) () const
'Getter' function for member m_seq_edit_mode A feature adapted from Kepler34.
- void [edit_mode](#) ([edit_mode_t](#) mode)
'Setter' function for member m_seq_edit_mode A feature adapted from Kepler34.
- void [modify](#) ()
A convenience function that we have to put here so that the m_parent pointer can be used without an additional include-file in the sequence.hpp module.
- int [event_count](#) () const
Returns the number of events stored in m_events.
- void [set_hold_undo](#) (bool hold)
Modifies the undo-hold container.
- int [get_hold_undo](#) () const
'Getter' function for member m_events_undo_hold.count()
- void [set_have_undo](#) ()

- 'Setter' function for member `m_have_undo`
- bool `have_undo` () const
 - 'Getter' function for member `m_have_undo`
- void `set_have_redo` ()
 - 'Setter' function for member `m_have_redo` No reliable way to "unmodify" the performance here.
- bool `have_redo` () const
 - 'Getter' function for member `m_have_redo`
- void `push_undo` (bool hold=false)
 - Pushes the event-list into the undo-list or the upcoming undo-hold-list.
- void `pop_undo` ()
 - If there are items on the undo list, this function pushes the event-list into the redo-list, puts the top of the undo-list into the event-list, pops from the undo-list, calls `verify_and_link()`, and then calls `unselect()`.
- void `pop_redo` ()
 - If there are items on the redo list, this function pushes the event-list into the undo-list, puts the top of the redo-list into the event-list, pops from the redo-list, calls `verify_and_link()`, and then calls `unselect()`.
- void `push_trigger_undo` ()
 - Calls `triggers::push_undo()` with locking.
- void `pop_trigger_undo` ()
 - Calls `triggers::pop_undo()` with locking.
- void `pop_trigger_redo` ()
 - Calls `triggers::pop_redo()` with locking.
- void `set_name` (const std::string &name="")
 - Sets the sequence name member, `m_name`.
- int `calculate_measures` () const
 - Calculates the number of measures in the sequence based on the unit-measure and the current length, in pulses, of the sequence.
- int `get_ppqn` () const
 - 'Getter' function for member `m_ppqn` Provided as a convenience for the `editable_events` class.
- void `set_beats_per_bar` (int beatspermeasure)
 - 'Setter' function for member `m_time_beats_per_measure`
- int `get_beats_per_bar` () const
 - 'Getter' function for member `m_time_beats_per_measure`
- void `set_beat_width` (int beatwidth)
 - 'Setter' function for member `m_time_beat_width`
- int `get_beat_width` () const
 - 'Getter' function for member `m_time_beat_width`
- `midipulse_measures_to_ticks` (int measures=1) const
 - A convenience function for calculating the number of ticks in the given number of measures.
- void `clocks_per_metronome` (int cpm)
 - 'Setter' function for member `m_clocks_per_metronome`
- int `clocks_per_metronome` () const
 - 'Getter' function for member `m_clocks_per_metronome`
- void `set_32nds_per_quarter` (int tpq)
 - 'Setter' function for member `m_32nds_per_quarter`
- int `get_32nds_per_quarter` () const
 - 'Getter' function for member `m_32nds_per_quarter`
- void `us_per_quarter_note` (long upqn)
 - 'Setter' function for member `m_us_per_quarter_note`
- long `us_per_quarter_note` () const
 - 'Getter' function for member `m_us_per_quarter_note`
- void `set_rec_vol` (int rec_vol)

- 'Setter' function for member `m_rec_vol` If this velocity is greater than zero, then `m_note_on_velocity` will be set as well.
- void `set_song_mute` (bool mute)

'Setter' function for member `m_song_mute` This function also calls `set_dirty_mp()` to make sure that the perfnames panel is updated to show the new mute status of the sequence.
 - void `toggle_song_mute` ()

'Setter' function for member `m_song_mute` This function toogles the song muting status.
 - bool `get_song_mute` () const

'Getter' function for member `m_song_mute`
 - void `apply_song_transpose` ()

Applies the transpose value held by the master MIDI buss object, if non-zero, and if the sequence is set to be transposable.
 - void `set_transposable` (bool flag)

'Setter' function for member `m_transposable` Changing this flag modifies the sequence and performance.
 - bool `get_transposable` () const

'Getter' function for member `m_transposable`
 - std::string `title` () const

Gets the title of the pattern, to show in the pattern slot.
 - const std::string & `name` () const

'Getter' function for member `m_name`
 - bool `is_default_name` () const

Tests the name for being changed.
 - void `set_editing` (bool edit)

'Setter' function for member `m_editing`
 - bool `get_editing` () const

'Getter' function for member `m_editing`
 - void `set_raise` (bool edit)

'Setter' function for member `m_raise`
 - bool `get_raise` (void) const

'Getter' function for member `m_raise`
 - void `set_length` (midipulse len=0, bool adjust_triggers=true, bool verify=true)

Sets the length (`m_length`) and adjusts triggers for it, if desired.
 - void `set_num_measures` (int measures)

Kepler34.
 - int `get_num_measures` ()

Kepler34.
 - void `apply_length` (int bpb, int ppqn, int bw, int measures=1)

Sets the sequence length based on the three given parameters.
 - int `extend` (midipulse len)

Extends the length of the sequence.
 - void `apply_length` (int meas=1)

An overload that gets its values from this sequence object.
 - midipulse `get_length` () const

'Getter' function for member `m_length`
 - midipulse `get_last_tick` () const

Returns the last tick played, and is used by the editor's idle function.
 - void `set_last_tick` (midipulse tick)

'Setter' function for member `m_last_tick` This function used to be called "`set_orig_tick()`", now renamed to match up with `get_last_tick()`.
 - midipulse `mod_last_tick` ()

Some MIDI file errors and other things can lead to an `m_length` of 0, which causes arithmetic errors when `m_last_tick` is modded against it.

- void [set_playing](#) (bool)
 - Sets the playing state of this sequence.*
- bool [get_playing](#) () const
 - 'Getter' function for member m_playing*
- void [toggle_playing](#) ()
- void [toggle_playing](#) (midipulse tick, bool resumenoteons)
 - Toggles the playing status of this sequence.*
- void [toggle_queued](#) ()
 - 'Setter' function for member m_queued and m_queued_tick Toggles the queued flag and sets the dirty-mp flag.*
- bool [get_queued](#) () const
 - 'Getter' function for member m_queued*
- midipulse [get_queued_tick](#) () const
 - 'Getter' function for member m_queued_tick*
- bool [check_queued_tick](#) (midipulse tick) const
 - Helper function for perform.*
- void [set_recording](#) (bool record_active)
 - 'Setter' function for member m_recording and m_notes_on*
- void [set_quantized_recording](#) (bool qr)
 - 'Setter' function for member m_quantized_rec*
- void [set_input_recording](#) (bool record_active, bool toggle=false)
 - Like [perform::set_sequence_input\(\)](#), but it uses the internal recording status directly, rather than getting it from seqedit.*
- bool [get_recording](#) () const
 - 'Getter' function for member m_recording*
- bool [recording_next_measure](#) () const
 - Makes a calculation for expanded recording, used in seqedit.*
- midipulse [get_snap_tick](#) () const
 - 'Getter' function for member m_snap_tick*
- void [set_snap_tick](#) (int st)
 - 'Setter' function for member m_snap_tick*
- bool [get_quantized_rec](#) () const
 - 'Getter' function for member m_quantized_rec*
- void [set_thru](#) (bool thru_active)
 - 'Setter' function for member m_thru*
- void [set_input_thru](#) (bool thru_active, bool toggle=false)
 - Like [perform::set_sequence_input\(\)](#), but it uses the internal thru status directly, rather than getting it from seqedit.*
- bool [get_thru](#) () const
 - 'Getter' function for member m_thru*
- void [off_one_shot](#) ()
 - Sets the dirty flag, sets m_one_shot to false, and m_off_from_snap to true.*
- void [song_recording_start](#) (midipulse tick, bool snap)
 - Starts the growing of the sequence for Song recording.*
- void [song_recording_stop](#) (midipulse tick)
 - Stops the growing of the sequence for Song recording.*
- midipulse [one_shot_tick](#) () const
 - 'Getter' function for member m_one_shot_tick*
- bool [song_recording](#) () const
 - 'Getter' function for member m_song_recording*
- bool [one_shot](#) () const
 - 'Getter' function for member m_one_shot*
- bool [off_from_snap](#) () const
 - 'Getter' function for member m_off_from_snap*

- bool [song_playback_block](#) () const
'Getter' function for member m_song_playback_block
- bool [song_recording_snap](#) () const
'Getter' function for member m_song_recording_snap
- [midipulse song_record_tick](#) () const
'Getter' function for member m_song_recording_tic
- void [resume_note_ons](#) ([midipulse](#) tick)
If the Note-On event is after the Note-Off event, the pattern wraps around, so that we play it now to resume.
- void [toggle_one_shot](#) ()
Toggles the m_one_shot flag, sets m_off_from_snap to true, and adjusts m_one_shot_tick according to m_last_tick and m_length.
- bool [is_dirty_main](#) ()
Returns the value of the dirty main flag, and sets that flag to false (i.e.
- bool [is_dirty_edit](#) ()
Returns the value of the dirty edit flag, and sets that flag to false.
- bool [is_dirty_perf](#) ()
Returns the value of the dirty performance flag, and sets that flag to false.
- bool [is_dirty_names](#) ()
Returns the value of the dirty names (heh heh) flag, and sets that flag to false.
- void [set_dirty_mp](#) ()
Sets the dirty flags for names, main, and performance.
- void [set_dirty](#) ()
Call [set_dirty_mp\(\)](#) and then sets the dirty flag for editing.
- [midibyte get_midi_channel](#) () const
'Getter' function for member m_midi_channel
- bool [is_smf_0](#) () const
Returns true if this sequence is an SMF 0 sequence.
- void [set_midi_channel](#) ([midibyte](#) ch, bool user_change=false)
Sets the m_midi_channel number, which is the output channel for this sequence.
- void [print](#) () const
Prints a list of the currently-held events.
- void [print_triggers](#) () const
Prints a list of the currently-held triggers.
- void [play](#) ([midipulse](#) tick, bool playback_mode, bool resume=false)
The [play\(\)](#) function dumps notes starting from the given tick, and it pre-buffers ahead.
- void [play_queue](#) ([midipulse](#) tick, bool playbackmode, bool resume)
Why don't we see this in kepler34? We do, in the MidiPerformance::play() function.
- bool [add_note](#) ([midipulse](#) tick, [midipulse](#) len, int note, bool paint=false, int velocity=SEQ64_PRESERVE_V↵ELOCITY)
Adds a note of a given length and note value, at a given tick location.
- bool [add_chord](#) (int chord, [midipulse](#) tick, [midipulse](#) len, int note)
Adds a chord of a given length and note value, at a given tick location.
- bool [add_event](#) (const [event](#) &er)
Adds an event to the internal event list in a sorted manner.
- bool [add_event](#) ([midipulse](#) tick, [midibyte](#) status, [midibyte](#) d0, [midibyte](#) d1, bool paint=false)
Adds a event of a given status value and data values, at a given tick location.
- bool [append_event](#) (const [event](#) &er)
An alternative to [add_event\(\)](#) that does not sort the events, even if the event list is implemented by an std::list.
- void [sort_events](#) ()
Calls [event_list::sort\(\)](#).
- void [add_trigger](#) ([midipulse](#) tick, [midipulse](#) len, [midipulse](#) offset=0, bool [adjust_offset](#)=true)

- Adds a trigger.*

 - void `split_trigger` (`midipulse` tick)
- Splits a trigger.*

 - void `half_split_trigger` (`midipulse` tick)
- Half-splits a trigger.*

 - void `exact_split_trigger` (`midipulse` tick)
- Exact-splits a trigger.*

 - void `grow_trigger` (`midipulse` tick_from, `midipulse` tick_to, `midipulse` len)
- Grows a trigger.*

 - void `delete_trigger` (`midipulse` tick)
- Deletes a trigger, that brackets the given tick, from the trigger-list.*

 - bool `get_trigger_state` (`midipulse` tick) const
- Checks the list of triggers against the given tick.*

 - bool `select_trigger` (`midipulse` tick)
- Checks the list of triggers against the given tick.*

 - `triggers::List` `get_triggers` () const
- Returns a copy of the triggers for this sequence.*

 - bool `unselect_trigger` (`midipulse` tick)
- Unselects the desired trigger.*

 - bool `unselect_triggers` ()
- Unselects all triggers.*

 - bool `intersect_triggers` (`midipulse` pos, `midipulse` &start, `midipulse` &end)
- This function examines each trigger in the trigger list.*

 - bool `intersect_triggers` (`midipulse` pos)
- This function examines each note in the event list.*

 - bool `intersect_notes` (`midipulse` position, int position_note, `midipulse` &start, `midipulse` &ender, int ¬e)
- This function examines each non-note event in the event list.*

 - bool `intersect_events` (`midipulse` posstart, `midipulse` posend, `midibyte` status, `midipulse` &start)
- This function examines each non-note event in the event list.*

 - void `delete_selected_triggers` ()
- Deletes the first selected trigger that is found.*

 - void `cut_selected_trigger` ()
- Copies and deletes the first selected trigger that is found.*

 - void `copy_selected_trigger` ()
- First, this function clears any unpasted middle-click tick setting.*

 - void `paste_trigger` (`midipulse` paste_tick=SEQ64_NO_PASTE_TRIGGER)
- If there is a copied trigger, then this function grabs it from the trigger clipboard and adds it.*

 - bool `move_triggers` (`midipulse` tick, bool `adjust_offset`, `triggers::grow_edit_t` which=`triggers::GROW_MOVE`)
- Moves selected triggers as per the given parameters.*

 - `midipulse` `selected_trigger_start` ()
- Gets the last-selected trigger's start tick.*

 - `midipulse` `selected_trigger_end` ()
- Gets the selected trigger's end tick.*

 - `midipulse` `get_max_trigger` () const
- Get the ending value of the last trigger in the trigger-list.*

 - void `move_triggers` (`midipulse` start_tick, `midipulse` distance, bool direction)
- Moves triggers in the trigger-list.*

 - void `copy_triggers` (`midipulse` start_tick, `midipulse` distance)
- Copies triggers to another location.*

 - void `clear_triggers` ()
- Clears the whole list of triggers.*

 - `midipulse` `get_trigger_offset` () const

- 'Getter' function for member m_trigger_offset*
- void [set_midi_bus](#) (char mb, bool user_change=false)
 - Sets the MIDI buss/port number to dump MIDI data to.*
- char [get_midi_bus](#) () const
 - 'Getter' function for member m_bus*
- void [set_master_midi_bus](#) (mastermidibus *mmb)
 - 'Setter' function for member m_master_bus Do we need to call [set_dirty_mp\(\)](#) here? It doesn't affect any user-interface elements.*
- int [select_note_events](#) (midipulse tick_s, int note_h, midipulse tick_f, int note_l, [select_action_t](#) action)
 - Selects events in range provided: tick start, note high, tick end, and note low.*
- int [select_events](#) (midipulse tick_s, midipulse tick_f, midibyte status, midibyte cc, [select_action_t](#) action)
 - Select all events in the given range, and returns the number selected.*
- int [select_events](#) (midibyte status, midibyte cc, bool inverse=false)
 - Select all events with the given status, and returns the number selected.*
- int [select_events](#) (midipulse tick_s, midipulse tick_f, midibyte status)
 - Select all events with the given status, and returns the number selected.*
- int [select_event_handle](#) (midipulse tick_s, midipulse tick_f, midibyte status, midibyte cc, int data_s)
 - Use selected note ons if any.*
- int [select_linked](#) (long tick_s, long tick_f, midibyte status)
 - Used with sequevent when selecting Note On or Note Off, this function will select the opposite linked event.*
- int [select_even_or_odd_notes](#) (int note_len, bool even)
 - Selects every other note.*
- void [select_all_notes](#) (bool inverse=false)
 - New convenience function.*
- int [get_num_selected_notes](#) () const
 - Counts the selected notes in the event list.*
- int [get_num_selected_events](#) (midibyte status, midibyte cc) const
 - Counts the selected events, with the given status, in the event list.*
- void [select_all](#) ()
 - Selects all events, unconditionally.*
- void [copy_selected](#) ()
 - Copies the selected events.*
- void [cut_selected](#) (bool copyevents=true)
 - Cuts the selected events.*
- void [paste_selected](#) (midipulse tick, int note)
 - Pastes the selected notes (and only note events) at the given tick and the given note value.*
- void [get_selected_box](#) (midipulse &tick_s, int ¬e_h, midipulse &tick_f, int ¬e_l)
 - Returns the 'box' of the selected items.*
- void [get_onsets_selected_box](#) (midipulse &tick_s, int ¬e_h, midipulse &tick_f, int ¬e_l)
 - Returns the 'box' of the selected items for only Note On values.*
- void [get_clipboard_box](#) (midipulse &tick_s, int ¬e_h, midipulse &tick_f, int ¬e_l)
 - Returns the 'box' of the clipboard items.*
- midipulse [adjust_timestamp](#) (midipulse t, bool isnoteoff)
 - A new function to consolidate the adjustment of timestamps in a pattern.*
- midipulse [trim_timestamp](#) (midipulse t)
 - A new function to consolidate the adjustment of timestamps in a pattern.*
- midipulse [clip_timestamp](#) (midipulse ontime, midipulse offtime)
 - A new function to consolidate the growth/shrinkage of timestamps in a pattern.*
- void [move_selected_notes](#) (midipulse deltatick, int deltanote)
 - Removes and adds selected notes in position.*
- bool [stream_event](#) (event &ev)
 - Streams the given event.*

- bool `change_event_data_range` (midipulse tick_s, midipulse tick_f, midibyte status, midibyte cc, int d_s, int d_f)
Changes the event data range.
- void `change_event_data_lfo` (double value, double range, double speed, double phase, wave_type_t wave, midibyte status, midibyte cc)
Modifies data events according to the parameters active in the LFO window (lfownd).
- void `increment_selected` (midibyte status, midibyte)
Increments events the match the given status and control values.
- void `decrement_selected` (midibyte status, midibyte)
Decrements events the match the given status and control values.
- void `grow_selected` (midipulse deltatick)
The original description was "Moves note off event." But this also gets called when simply selecting a second note via a ctrl-left-click, even in seq24.
- void `stretch_selected` (midipulse deltatick)
Performs a stretch operation on the selected events.
- bool `mark_selected` ()
Marks the selected events.
- void `remove_selected` ()
Removes selected events.
- bool `remove_marked` ()
Removes marked events.
- void `unpaint_all` ()
Unpaints all events in the event-list.
- void `unselect` ()
Deselects all events, unconditionally.
- void `verify_and_link` ()
This function verifies state: all note-ons have a note-off, and it links note-offs with their note-ons.
- void `link_new` ()
Links a new event.
- void `link_tempos` ()
A new function to re-link the tempo events added by the user.
- void `zero_markers` ()
Resets everything to zero.
- void `play_note_on` (int note)
Plays a note from the piano roll on the main bus on the master MIDI buss.
- void `play_note_off` (int note)
Turns off a note from the piano roll on the main bus on the master MIDI buss.
- void `off_playing_notes` ()
Sends a note-off event for all active notes.
- void `stop` (bool song_mode=false)
Provides a helper function simplify and speed up perform :: reset_sequences().
- void `pause` (bool song_mode=false)
A pause version of stop().
- void `inc_draw_marker` ()
This increments the draw marker.
- void `reset_draw_marker` ()
This refreshes the draw marker to the first event.
- void `reset_draw_trigger_marker` ()
Sets the draw-trigger iterator to the beginning of the trigger list.
- void `reset_ex_iterator` (event_list::const_iterator &evi)
Reset the caller's iterator.

- `draw_type_t get_next_note_event` (midipulse &tick_s, midipulse &tick_f, int ¬e, bool &selected, int &velocity)
Each call to seqdata() fills the passed references with a events elements, and returns true.
- `bool get_minmax_note_events` (int &lowest, int &highest)
A new function provided so that we can find the minimum and maximum notes with only one (not two) traversal of the event list.
- `bool get_next_event` (midibyte &status, midibyte &cc)
Get the next event in the event list.
- `bool get_next_event_ex` (midibyte status, midibyte cc, event_list::const_iterator &ev, int evtype=EVENTS_↵ALL)
New version in progress.
- `bool get_next_trigger` (midipulse &tick_on, midipulse &tick_off, bool &selected, midipulse &tick_offset)
Get the next trigger in the trigger list, and set the parameters based on that trigger.
- `void quantize_events` (midibyte status, midibyte cc, midipulse snap_tick, int divide, bool linked=false)
Grabs the specified events, puts them into a list, quantizes them against the snap ticks, and merges them in to the event container.
- `void push_quantize` (midibyte status, midibyte cc, midipulse snap_tick, int divide, bool linked=false)
A new convenience function.
- `void transpose_notes` (int steps, int scale)
Transposes notes by the given steps, in accordance with the given scale.
- `void shift_notes` (midipulse ticks)
We need to look into this function.
- `void multiply_pattern` (double multiplier)
- `midibyte musical_key` () const
'Getter' function for member m_musical_key
- `void musical_key` (int key)
'Setter' function for member m_musical_key
- `midibyte musical_scale` () const
'Getter' function for member m_musical_scale
- `void musical_scale` (int scale)
'Setter' function for member m_musical_scale
- `int background_sequence` () const
'Getter' function for member m_background_sequence
- `void background_sequence` (int bs)
'Setter' function for member m_background_sequence Only partial validation at present, we do not want the upper limit to be hard-wired at this time.
- `void show_events` () const
A member function to dump a summary of events stored in the event-list of a sequence.
- `void copy_events` (const event_list &newevents)
Copies an external container of events into the current container, effectively replacing all of its events.
- `midipulse note_off_margin` () const
'Getter' function for member m_note_length
- `void set_unit_measure` () const
Calculates and sets $u = 4BP/W$, where u is $m_unit_measure$, B is the beats/bar, P is the PPQN, and W is the beat-width.
- `midipulse get_unit_measure` () const
'Getter' function for member m_unit_measure
- `bool channel_match` () const
'Getter' function for member m_channel_match The master bus needs to know if the match feature is truly in force, otherwise it must pass the incoming events to all recording sequences.
- `void set_overwrite_rec` (bool ovwr)
'Setter' function for member m_overwrite_recording

- bool [get_overwrite_rec](#) ()
'Getter' function for member m_overwrite_recording
- void [set_loop_reset](#) (bool reset)
'Setter' function for member m_loop_reset
- bool [get_loop_reset](#) ()
'Getter' function for member m_loop_reset
- [midipulse handle_size](#) ([midipulse](#) start, [midipulse](#) finish)
Actually, useful mainly for the user-interface, this function calculates the size of the left and right handles of a note.

Static Public Member Functions

- static const std::string & [default_name](#) ()
'Getter' function for member sm_default_name

Private Types

- typedef std::stack< [event_list](#) > [EventStack](#)
Provides a stack of event-lists for use with the undo and redo facility.

Private Member Functions

- [sequence](#) & [operator=](#) (const [sequence](#) &rhs)
- bool [event_in_range](#) (const [event](#) &e, [midibyte](#) status, [midipulse](#) tick_s, [midipulse](#) tick_f) const
A convenience function used a couple of times.
- void [set_parent](#) ([perform](#) *p)
'Setter' function for member m_parent Sets the "parent" of this sequence, so that it can get some extra information about the performance.
- void [put_event_on_bus](#) ([event](#) &ev)
Takes an event that this sequence is holding, and places it on the MIDI buss.
- void [reset_loop](#) ()
- void [set_trigger_offset](#) ([midipulse](#) trigger_offset)
Sets m_trigger_offset and wraps it to m_length.
- void [adjust_trigger_offsets_to_length](#) ([midipulse](#) newlen)
Adjusts trigger offsets to the length specified for all triggers, and undo triggers.
- [midipulse](#) [adjust_offset](#) ([midipulse](#) offset)
- void [remove](#) ([event_list::iterator](#) i)
A helper function, which does not lock/unlock, so it is unsafe to call without supplying an iterator from the event-list.
- void [remove](#) ([event](#) &e)
A helper function, which does not lock/unlock, so it is unsafe to call without supplying an iterator from the event-list.
- void [remove_all](#) ()
Clears all events from the event container.
- bool [channels_match](#) (const [event](#) &e) const
Checks to see if the event's channel matches the sequence's nominal channel.
- void [one_shot](#) (bool f)
'Setter' function for member m_one_shot
- void [off_from_snap](#) (bool f)
'Setter' function for member m_off_from_snap
- void [song_playback_block](#) (bool f)
'Setter' function for member m_song_playback_block

- void [song_recording](#) (bool f)
'Setter' function for member m_song_recording
- void [song_recording_snap](#) (bool f)
'Getter' function for member m_song_recording_snap
- void [song_record_tick](#) (midipulse t)
'Getter' function for member m_song_record_tick

Private Attributes

- [perform](#) * [m_parent](#)
For pause support, we need a way for the sequence to find out if JACK transport is active.
- [event_list](#) [m_events](#)
This list holds the current pattern/sequence events.
- [triggers](#) [m_triggers](#)
Holds the list of triggers associated with the sequence, used in the performance/song editor.
- [event_list](#) [m_events_undo_hold](#)
Provides a list of event actions to undo for the Stazed LFO and seqdata support.
- bool [m_have_undo](#)
A stazed flag indicating that we have some undo information.
- bool [m_have_redo](#)
A stazed flag indicating that we have some redo information.
- [EventStack](#) [m_events_undo](#)
Provides a list of event actions to undo.
- [EventStack](#) [m_events_redo](#)
Provides a list of event actions to redo.
- [event_list::iterator](#) [m_iterator_draw](#)
An iterator for drawing events.
- bool [m_channel_match](#)
A new feature for recording, based on a "stazed" feature.
- [midibyte](#) [m_midi_channel](#)
Contains the proper MIDI channel for this sequence.
- [midibyte](#) [m_bus](#)
Contains the proper MIDI bus number for this sequence.
- bool [m_song_mute](#)
Provides a flag for the song playback mode muting.
- bool [m_transposable](#)
Indicate if the sequence is transposable or not.
- short [m_notes_on](#)
Provides a member to hold the polyphonic step-edit note counter.
- [mastermidibus](#) * [m_master_bus](#)
Provides the master MIDI buss which handles the output of the sequence to the proper buss and MIDI channel.
- short [m_playing_notes](#) [SEQ64_MIDI_NOTES_MAX]
Provides a "map" for Note On events.
- bool [m_was_playing](#)
Indicates if the sequence was playing.
- bool [m_playing](#)
True if sequence playback currently is in progress for this sequence.
- bool [m_recording](#)
True if sequence recording currently is in progress for this sequence.
- bool [m_quantized_rec](#)

- True if recording in quantized mode.*

 - bool `m_thru`

True if recording in MIDI-through mode.
- bool `m_queued`

True if the events are queued.
- bool `m_one_shot`

A member from the Kepler34 project to indicate we are in one-shot mode for triggering.
- midipulse `m_one_shot_tick`

A Member from the Kepler34 project, set in sequence :: `toggle_one_shot()` to `m_last_tick` adjusted to the length of the sequence.
- bool `m_off_from_snap`

Indicates if we have turned off from a snap operation.
- bool `m_song_playback_block`

Used to temporarily block Song Mode events while recording new ones.
- bool `m_song_recording`

Used to keep on blocking Song Mode events while recording new ones.
- bool `m_song_recording_snap`

This value indicates that the following feature is active: the number of tick to snap recorded improvisations.
- midipulse `m_song_record_tick`

Saves the tick from when we started recording live song data.
- bool `m_overwrite_recording`

Indicates if overwrite recording more is in force.
- bool `m_loop_reset`

Indicates if the play marker has gone to the beginning of the sequence upon looping.
- midipulse `m_unit_measure`

Hold the current unit for a measure.
- bool `m_dirty_main`

These flags indicate that the content of the sequence has changed due to recording, editing, performance management, or even (?) a name change.
- bool `m_dirty_edit`

Provides the main is-edited flag.
- bool `m_dirty_perf`

Provides performance dirty flagflag.
- bool `m_dirty_names`

Provides the names dirtiness flag.
- bool `m_editing`

Indicates that the sequence is currently being edited.
- bool `m_raise`

Used in seqmenu and seqedit.
- std::string `m_name`

Provides the name/title for the sequence.
- midipulse `m_last_tick`

These members manage where we are in the playing of this sequence, including triggering.
- midipulse `m_queued_tick`

Provides the tick for queuing.
- midipulse `m_trigger_offset`

Provides the trigger offset.
- const int `m_maxbeats`

This constant provides the scaling used to calculate the time position in ticks (pulses), based also on the PPQN value.
- unsigned short `m_ppqn`

Holds the PPQN value for this sequence, so that we don't have to rely on a global constant value.

- short [m_seq_number](#)
A new member so that the sequence number is carried along with the sequence.
- [colorbyte m_seq_color](#)
Reserved for a potential feature from the Kepler34 project.
- [edit_mode_t m_seq_edit_mode](#)
A feature adapted from Kepler34.
- [midipulse m_length](#)
Holds the length of the sequence in pulses (ticks).
- [midipulse m_snap_tick](#)
The size of snap in units of pulses (ticks).
- unsigned short [m_time_beats_per_measure](#)
Provides the number of beats per bar used in this sequence.
- unsigned short [m_time_beat_width](#)
Provides with width of a beat.
- int [m_clocks_per_metronome](#)
Augments the beats/bar and beat-width with the additional values included in a Time Signature meta event.
- int [m_32nds_per_quarter](#)
Augments the beats/bar and beat-width with the additional values included in a Time Signature meta event.
- long [m_us_per_quarter_note](#)
Augments the beats/bar and beat-width with the additional values included in a Tempo meta event.
- short [m_rec_vol](#)
The volume to be used when recording.
- short [m_note_on_velocity](#)
The Note On velocity used.
- short [m_note_off_velocity](#)
The Note Off velocity used.
- [midibyte m_musical_key](#)
Holds a copy of the musical key for this sequence, which we now support writing to this sequence.
- [midibyte m_musical_scale](#)
Holds a copy of the musical scale for this sequence, which we now support writing to this sequence.
- short [m_background_sequence](#)
Holds a copy of the background sequence number for this sequence, which we now support writing to this sequence.
- [mutex m_mutex](#)
Provides locking for the sequence.
- const [midipulse m_note_off_margin](#)
Provides the number of ticks to shave off of the end of painted notes.

Static Private Attributes

- static [event_list m_events_clipboard](#)
A static clipboard for holding pattern/sequence events.
- static const std::string [sm_default_name](#)
Provides the default name/title for the sequence.

Friends

- class [perform](#)
- class [triggers](#)

10.91.1 Detailed Description

More members than you can shake a stick at.

10.91.2 Member Typedef Documentation

10.91.2.1 EventStack

```
typedef std::stack<event_list> seq64::sequence::EventStack [private]
```

10.91.3 Member Enumeration Documentation

10.91.3.1 select_action_t

```
enum seq64::sequence::select_action_t
```

Se the [select_note_events\(\)](#) and [select_events\(\)](#) functions.

Enumerator

e_select	Selection in progress.
e_select_one	To select a single event.
e_is_selected	The events are selected.
e_would_select	The events would be selected.
e_deselect	To deselect event under the cursor.
e_toggle_selection	Toggle selection under cursor.
e_remove_one	To remove one note under the cursor.
e_select_onset	Kepler34, To select a single onset.
e_is_selected_onset	New, from Kepler34, onsets selected.

10.91.4 Constructor & Destructor Documentation

10.91.4.1 sequence()

```
seq64::sequence::sequence (
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

Parameters

<i>ppqn</i>	Provides the PPQN parameter to perhaps alter the default PPQN value of this sequence.
-------------	---

10.91.4.2 ~sequence()

```
seq64::sequence::~~sequence ( )
```

10.91.5 Member Function Documentation

10.91.5.1 operator=()

```
sequence& seq64::sequence::operator= (
    const sequence & rhs ) [private]
```

10.91.5.2 partial_assign()

```
void seq64::sequence::partial_assign (
    const sequence & rhs )
```

We're replacing that incomplete seq24 function (many members are not assigned) with the more accurately-named [partial_assign\(\)](#) function. It did not assign them all, so we created this [partial_assign\(\)](#) function to do this work, and replaced [operator =\(\)](#) with this function in client code.

Threadsafe

Parameters

<i>rhs</i>	Provides the source of the new member values.
------------	---

10.91.5.3 events() [1/2]

```
event_list& seq64::sequence::events ( ) [inline]
```

10.91.5.4 events() [2/2]

```
const event_list& seq64::sequence::events ( ) const [inline]
```

10.91.5.5 any_selected_notes()

```
bool seq64::sequence::any_selected_notes ( ) const [inline]
```

10.91.5.6 triggerlist() [1/2]

```
const triggers::List& seq64::sequence::triggerlist ( ) const [inline]
```

10.91.5.7 triggerlist() [2/2]

```
triggers::List& seq64::sequence::triggerlist ( ) [inline]
```

10.91.5.8 trigger_count()

```
int seq64::sequence::trigger_count ( ) const [inline]
```

10.91.5.9 selected_trigger_count()

```
int seq64::sequence::selected_trigger_count ( ) const [inline]
```

That is, selected in the perfrroll.

10.91.5.10 set_trigger_paste_tick()

```
void seq64::sequence::set_trigger_paste_tick (
    midipulse tick ) [inline]
```

Parameters

<i>tick</i>	Provides the pulse value to set.
-------------	----------------------------------

10.91.5.11 get_trigger_paste_tick()

```
midipulse seq64::sequence::get_trigger_paste_tick ( ) const [inline]
```

Returns

Returns the current pulse value.

10.91.5.12 seq_number()

```
std::string seq64::sequence::seq_number ( ) const [inline]
```

10.91.5.13 number() [1/2]

```
int seq64::sequence::number ( ) const [inline]
```

10.91.5.14 number() [2/2]

```
void seq64::sequence::number (
    int seqnum ) [inline]
```

10.91.5.15 color() [1/2]

```
int seq64::sequence::color ( ) const [inline]
```

10.91.5.16 color() [2/2]

```
void seq64::sequence::color (
    int c ) [inline]
```

Parameters

c	Provides the index into the color-palette. The only rules here are that -1 represents no color or a default color, and values of zero and above (to an unknown limit) represent a legal palette color.
----------	--

10.91.5.17 edit_mode() [1/2]

```
edit_mode_t seq64::sequence::edit_mode ( ) const [inline]
```

10.91.5.18 edit_mode() [2/2]

```
void seq64::sequence::edit_mode (
    edit_mode_t mode ) [inline]
```

10.91.5.19 modify()

```
void seq64::sequence::modify ( )
```

One minor issue is how can we unmodify the performance? We'd need to keep a count/stack of modifications over all sequences in the performance. Probably not practical, in general. We will probably keep track of the modification of the buss (port) and channel numbers, as per GitHub Issue #47.

10.91.5.20 event_count()

```
int seq64::sequence::event_count ( ) const
```

Note that only playable events are counted in a sequence. If a sequence class function provides a mutex, call `m_events.count()` instead.

Threadsafe

Returns

Returns `m_events.count()`.

10.91.5.21 set_hold_undo()

```
void seq64::sequence::set_hold_undo (
    bool hold )
```

Parameters

<i>hold</i>	If true, then the events in the <code>m_events</code> container are added to the <code>m_events_undo_hold</code> container. Otherwise, that container is cleared.
-------------	---

10.91.5.22 get_hold_undo()

```
int seq64::sequence::get_hold_undo ( ) const [inline]
```

10.91.5.23 set_have_undo()

```
void seq64::sequence::set_have_undo ( ) [inline]
```

10.91.5.24 have_undo()

```
bool seq64::sequence::have_undo ( ) const [inline]
```

10.91.5.25 set_have_redo()

```
void seq64::sequence::set_have_redo ( ) [inline]
```

10.91.5.26 have_redo()

```
bool seq64::sequence::have_redo ( ) const [inline]
```

10.91.5.27 push_undo()

```
void seq64::sequence::push_undo (
    bool hold = false )
```

Threadsafe

Parameters

<i>hold</i>	A new parameter for the stazed undo/redo support, not yet used. If true, then the events go into the undo-hold-list.
-------------	--

10.91.5.28 pop_undo()

```
void seq64::sequence::pop_undo ( )
```

We would like to be able to set perform's modify flag to false here, but other sequences might still be in a modified state. We could add a modify flag to sequence, and falsify that flag here. Something to think about.

Threadsafe

10.91.5.29 pop_redo()

```
void seq64::sequence::pop_redo ( )
```

Threadsafe

10.91.5.30 push_trigger_undo()

```
void seq64::sequence::push_trigger_undo ( )
```

Threadsafe

10.91.5.31 pop_trigger_undo()

```
void seq64::sequence::pop_trigger_undo ( )
```

Threadsafe

10.91.5.32 pop_trigger_redo()

```
void seq64::sequence::pop_trigger_redo ( )
```

Threadsafe

10.91.5.33 set_name()

```
void seq64::sequence::set_name (
    const std::string & name = "" )
```

This is the name shown in the top of a mainwid pattern slot.

We now try to include the length of the sequences in measures at the end of the name, and limit the length of the entire string. As noted in the printing of sequence::get_name() in mainwid, this length is 13 characters.

10.91.5.34 calculate_measures()

```
int seq64::sequence::calculate_measures ( ) const
```

Used by seqedit. If m_unit_measure hasn't been calculated yet, it is calculated here.

Change Note ca 2017-08-15 Fixed issue #106, where the measure count of a pattern kept incrementing when edited.

Returns

Returns the sequence length divided by the measure length, roughly. m_unit_measure is 0. The lowest valid measure is 1.

10.91.5.35 get_ppqn()

```
int seq64::sequence::get_ppqn ( ) const [inline]
```

10.91.5.36 set_beats_per_bar()

```
void seq64::sequence::set_beats_per_bar (
    int beatspermeasure )
```

Threadsafe

Parameters

<i>beatspermeasure</i>	The new setting of the beats-per-bar value.
------------------------	---

10.91.5.37 get_beats_per_bar()

```
int seq64::sequence::get_beats_per_bar ( ) const [inline]
```

10.91.5.38 set_beat_width()

```
void seq64::sequence::set_beat_width (
    int beatwidth )
```

Threadsafe

Parameters

<i>beatwidth</i>	The new setting of the beat width value.
------------------	--

10.91.5.39 `get_beat_width()`

```
int seq64::sequence::get_beat_width ( ) const [inline]
```

*Threadsafe*10.91.5.40 `measures_to_ticks()`

```
midipulse seq64::sequence::measures_to_ticks (
    int measures = 1 ) const [inline]
```

10.91.5.41 `clocks_per_metronome()` [1/2]

```
void seq64::sequence::clocks_per_metronome (
    int cpm ) [inline]
```

10.91.5.42 `clocks_per_metronome()` [2/2]

```
int seq64::sequence::clocks_per_metronome ( ) const [inline]
```

10.91.5.43 `set_32nds_per_quarter()`

```
void seq64::sequence::set_32nds_per_quarter (
    int tpq ) [inline]
```

10.91.5.44 `get_32nds_per_quarter()`

```
int seq64::sequence::get_32nds_per_quarter ( ) const [inline]
```

10.91.5.45 us_per_quarter_note() [1/2]

```
void seq64::sequence::us_per_quarter_note (
    long upqn ) [inline]
```

10.91.5.46 us_per_quarter_note() [2/2]

```
long seq64::sequence::us_per_quarter_note ( ) const [inline]
```

10.91.5.47 set_rec_vol()

```
void seq64::sequence::set_rec_vol (
    int recvol )
```

*Threadsafe***Parameters**

<i>recvol</i>	The new setting of the recording volume setting. It is used only if it ranges from 0 to SEQ64_MAX_NOTE_ON_VELOCITY, or is set to SEQ64_PRESERVE_VELOCITY.
---------------	---

10.91.5.48 set_song_mute()

```
void seq64::sequence::set_song_mute (
    bool mute ) [inline]
```

10.91.5.49 toggle_song_mute()

```
void seq64::sequence::toggle_song_mute ( ) [inline]
```

10.91.5.50 get_song_mute()

```
bool seq64::sequence::get_song_mute ( ) const [inline]
```

10.91.5.51 `apply_song_transpose()`

```
void seq64::sequence::apply_song_transpose ( )
```

10.91.5.52 `set_transposable()`

```
void seq64::sequence::set_transposable (
    bool flag )
```

Note that when a sequence is being read from a MIDI file, it will not yet have a parent, so we have to check for that before setting the perform modify flag.

10.91.5.53 `get_transposable()`

```
bool seq64::sequence::get_transposable ( ) const [inline]
```

10.91.5.54 `title()`

```
std::string seq64::sequence::title ( ) const
```

This function differs from `name`, which just returns the value of `m_name`. Here, we also include the length of the sequences in measures at the end of the name, and limit the length of the entire string. As noted in the printing of `sequence::get_name()` in `mainwid`, this length is 13 characters.

Returns

Returns the name of the sequence, with the length in measures of the pattern wedged in at the end, if non-zero.

10.91.5.55 `name()`

```
const std::string& seq64::sequence::name ( ) const [inline]
```

10.91.5.56 `is_default_name()`

```
bool seq64::sequence::is_default_name ( ) const [inline]
```

10.91.5.57 default_name()

```
static const std::string& seq64::sequence::default_name ( ) [inline], [static]
```

10.91.5.58 set_editing()

```
void seq64::sequence::set_editing (
    bool edit ) [inline]
```

10.91.5.59 get_editing()

```
bool seq64::sequence::get_editing ( ) const [inline]
```

10.91.5.60 set_raise()

```
void seq64::sequence::set_raise (
    bool edit ) [inline]
```

10.91.5.61 get_raise()

```
bool seq64::sequence::get_raise (
    void ) const [inline]
```

10.91.5.62 set_length()

```
void seq64::sequence::set_length (
    midipulse len = 0,
    bool adjust_triggers = true,
    bool verify = true )
```

This function is called in [seqedit::apply_length\(\)](#), when the user selects a sequence length in measures. This function is also called when reading a MIDI file.

There's an issue, though. If the application is compiled to use the original `std::list` container for MIDI events, that implementation sorts the container after every event insertion. If the application is compiled to use the `std::map` container (to speed up the reading of large MIDI files *greatly*), sorting happens automatically. But, if we use the original `std::list` implementation, but leave the sorting until later (to speed up the reading of large MIDI files *greatly*), then the [verify_and_link\(\)](#) call that happens with every new event happens before the events are sorted, and the result is elongated notes showing up in the pattern slot in the main window. Therefore, we need a way to skip the verification when reading a MIDI file, and do the verification only after all events are read.

That function calculates the length in ticks:

```

L = M x B x 4 x P / W
L == length (ticks or pulses)
M == number of measures
B == beats per measure
P == pulses per quarter-note
W == beat width in beats per measure

```

For our "b4uacuse" MIDI file, M can be about 100 measures, B is 4, P can be 192 (but we want to support higher values), and W is 4. So $L = 100 * 4 * 4 * 192 / 4 = 76800$ ticks. Seems small.

Threadsafe

Parameters

<i>len</i>	The length value to be set. If it is smaller than $ppqn/4$, then it is set to that value, unless it is zero, in which case <i>m_length</i> is used and does not change. It also sets the length value for the sequence's triggers.
<i>adjust_triggers</i>	If true, <i>m_triggers.adjust_offsets_to_length()</i> is called. The value defaults to true.
<i>verify</i>	This new parameter defaults to true. If true, verify_and_link() is called. Otherwise, it is not, and the caller should call this function with the default value after reading all the events.

10.91.5.63 set_num_measures()

```

void seq64::sequence::set_num_measures (
    int measures ) [inline]

```

10.91.5.64 get_num_measures()

```

int seq64::sequence::get_num_measures ( )

```

10.91.5.65 apply_length() [1/2]

```

void seq64::sequence::apply_length (
    int bpb,
    int ppqn,
    int bw,
    int measures = 1 )

```

There's an implicit "adjust-triggers = true" parameter used in this function. Please note that there is an overload that takes only a measure number and uses the current beats/bar, PPQN, and beat-width values of this sequence. The [set_unit_measure\(\)](#) function is called, but won't change any values just because the length (number of measures) changed.

Warning

The measures calculation is useless if the BPM (beats/minute) varies throughout the song.

Parameters

<i>bpb</i>	Provides the beats per bar (measure).
<i>ppqn</i>	Provides the pulses-per-quarter-note to apply to the length application.
<i>bw</i>	Provides the beatwidth (typically 4) from the time signature.
<i>measures</i>	Provides the number of measures the sequence should cover, obtained from the user-interface.

10.91.5.66 `extend()`

```
int seq64::sequence::extend (
    midipulse len )
```

Calls `set_length()` with the new length and its default parameters. Not sure how useful this function is.

Parameters

<i>len</i>	The new length of the sequence.
------------	---------------------------------

Returns

Returns the new number of measures.

10.91.5.67 `apply_length()` [2/2]

```
void seq64::sequence::apply_length (
    int meas = 1 ) [inline]
```

Parameters

<i>meas</i>	The number of measures to apply. Defaults to 1.
-------------	---

10.91.5.68 `get_length()`

```
midipulse seq64::sequence::get_length ( ) const [inline]
```

10.91.5.69 get_last_tick()

```
midipulse seq64::sequence::get_last_tick ( ) const
```

If `m_length` is 0, this function returns `m_last_tick - m_trigger_offset`, to avoid an arithmetic exception. Should we return 0 instead?

Note that `seqroll` calls this function to help get the location of the progress bar. What does `perfedit` do?

10.91.5.70 set_last_tick()

```
void seq64::sequence::set_last_tick (
    midipulse tick )
```

Threadsafe

10.91.5.71 mod_last_tick()

```
midipulse seq64::sequence::mod_last_tick ( ) [inline]
```

This function replaces the "`m_last_tick % m_length`", returning `m_last_tick` if `m_length` is 0 or 1.

10.91.5.72 set_playing()

```
void seq64::sequence::set_playing (
    bool p )
```

When playing, and the sequencer is running, notes get dumped to the ALSA buffers.

Parameters

<i>p</i>	Provides the playing status to set. True means to turn on the playing, false means to turn it off, and turn off any notes still playing.
----------	--

10.91.5.73 get_playing()

```
bool seq64::sequence::get_playing ( ) const [inline]
```

10.91.5.74 toggle_playing() [1/2]

```
void seq64::sequence::toggle_playing ( ) [inline]
```

10.91.5.75 toggle_playing() [2/2]

```
void seq64::sequence::toggle_playing (
    midipulse tick,
    bool resumenoteons )
```

How exactly does this differ from toggling the mute status?

Parameters

<i>tick</i>	The position from which to resume Note Ons, if applicable. Resuming is a song-recording feature.
<i>resumenoteons</i>	A song-recording option.

10.91.5.76 toggle_queued()

```
void seq64::sequence::toggle_queued ( )
```

Also calculates the queued tick based on `m_last_tick`.

Threadsafe

10.91.5.77 get_queued()

```
bool seq64::sequence::get_queued ( ) const [inline]
```

10.91.5.78 get_queued_tick()

```
midipulse seq64::sequence::get_queued_tick ( ) const [inline]
```

10.91.5.79 check_queued_tick()

```
bool seq64::sequence::check_queued_tick (
    midipulse tick ) const [inline]
```

10.91.5.80 set_recording()

```
void seq64::sequence::set_recording (
    bool r )
```

This function sets `m_notes_on` to 0, but this should be done only if the recording status has changed.

Threadsafe

10.91.5.81 set_quantized_recording()

```
void seq64::sequence::set_quantized_recording (
    bool qr )
```

What about doing this?

```
m_master_bus->set_sequence_input(record_active, this);
```

Threadsafe

10.91.5.82 set_input_recording()

```
void seq64::sequence::set_input_recording (
    bool record_active,
    bool toggle = false )
```

Do we need a quantized recording version, or is setting the quantized-recording flag sufficient?

Parameters

<i>record_active</i>	Provides the desired status to set recording.
<i>toggle</i>	If true, ignore the first parameter and toggle the flag. The default value is false.

10.91.5.83 get_recording()

```
bool seq64::sequence::get_recording ( ) const [inline]
```

10.91.5.84 recording_next_measure()

```
bool seq64::sequence::recording_next_measure ( ) const [inline]
```

10.91.5.85 get_snap_tick()

```
midipulse seq64::sequence::get_snap_tick ( ) const [inline]
```

10.91.5.86 set_snap_tick()

```
void seq64::sequence::set_snap_tick (
    int st )
```

Threadsafe

10.91.5.87 get_quantized_rec()

```
bool seq64::sequence::get_quantized_rec ( ) const [inline]
```

10.91.5.88 set_thru()

```
void seq64::sequence::set_thru (
    bool r )
```

Threadsafe

10.91.5.89 set_input_thru()

```
void seq64::sequence::set_input_thru (
    bool thru_active,
    bool toggle = false )
```

Parameters

<i>thru_active</i>	Provides the desired status to set the through state.
<i>toggle</i>	If true, ignore the first parameter and toggle the flag. The default value is false.

10.91.5.90 get_thru()

```
bool seq64::sequence::get_thru ( ) const [inline]
```

10.91.5.91 off_one_shot()

```
void seq64::sequence::off_one_shot ( )
```

This function remains unused here and in Kepler34.

10.91.5.92 song_recording_start()

```
void seq64::sequence::song_recording_start (
    midipulse tick,
    bool snap )
```

This process starts by adding a chunk of SEQ64_SONG_RECORD_INC ticks to the trigger, which allows the rest of the threads to notice the change.

Question Do we need to call [set_dirty_mp\(\)](#) here?

Parameters

<i>tick</i>	Provides the current tick, which helps set the recorded block's boundaries, and is copied into m_song_record_tick.
<i>snap</i>	Indicates if we are to snap the recording to the configured boundary, and is copied into m_song_recording_snap.

10.91.5.93 song_recording_stop()

```
void seq64::sequence::song_recording_stop (
    midipulse tick )
```

If we have been recording, we snap the end of the trigger segment to the next whole sequence interval.

Question Do we need to call [set_dirty_mp\(\)](#) here?

Parameters

<i>tick</i>	Provides the current tick, which helps set the recorded block's boundaries.
-------------	---

10.91.5.94 one_shot_tick()

```
midipulse seq64::sequence::one_shot_tick ( ) const [inline]
```

10.91.5.95 song_recording() [1/2]

```
bool seq64::sequence::song_recording ( ) const [inline]
```

10.91.5.96 one_shot() [1/2]

```
bool seq64::sequence::one_shot ( ) const [inline]
```

10.91.5.97 off_from_snap() [1/2]

```
bool seq64::sequence::off_from_snap ( ) const [inline]
```

10.91.5.98 song_playback_block() [1/2]

```
bool seq64::sequence::song_playback_block ( ) const [inline]
```

10.91.5.99 song_recording_snap() [1/2]

```
bool seq64::sequence::song_recording_snap ( ) const [inline]
```

10.91.5.100 song_record_tick() [1/2]

```
midipulse seq64::sequence::song_record_tick ( ) const [inline]
```

10.91.5.101 resume_note_ons()

```
void seq64::sequence::resume_note_ons (
    midipulse tick )
```

Parameters

<i>tick</i>	The current tick-time, in MIDI pulses.
-------------	--

10.91.5.102 toggle_one_shot()

```
void seq64::sequence::toggle_one_shot ( )
```

10.91.5.103 `is_dirty_main()`

```
bool seq64::sequence::is_dirty_main ( )
```

resets it). This flag signals that a redraw is needed from recording.

Threadsafe

Returns

Returns the dirty status.

10.91.5.104 `is_dirty_edit()`

```
bool seq64::sequence::is_dirty_edit ( )
```

The `m_dirty_edit` flag is set by the function [set_dirty\(\)](#).

Threadsafe

Returns

Returns the dirty status.

10.91.5.105 `is_dirty_perf()`

```
bool seq64::sequence::is_dirty_perf ( )
```

Threadsafe

Returns

Returns the dirty status.

10.91.5.106 `is_dirty_names()`

```
bool seq64::sequence::is_dirty_names ( )
```

Not sure that we need to lock a boolean on modern processors.

Threadsafe

Returns

Returns the dirty status.

10.91.5.107 set_dirty_mp()

```
void seq64::sequence::set_dirty_mp ( )
```

These flags are meant for causing user-interface refreshes, not for performance modification.

m_dirty_names is set to false in [is_dirty_names\(\)](#); m_dirty_names is set to false in [is_dirty_main\(\)](#); m_dirty_names is set to false in [is_dirty_perf\(\)](#).

Not threadsafe

10.91.5.108 set_dirty()

```
void seq64::sequence::set_dirty ( )
```

Threadsafe

10.91.5.109 get_midi_channel()

```
midibyte seq64::sequence::get_midi_channel ( ) const [inline]
```

10.91.5.110 is_smf_0()

```
bool seq64::sequence::is_smf_0 ( ) const [inline]
```

10.91.5.111 set_midi_channel()

```
void seq64::sequence::set_midi_channel (
    midibyte ch,
    bool user_change = false )
```

Threadsafe

Parameters

<i>ch</i>	The MIDI channel to set as the output channel number for this sequence.
<i>user_change</i>	If true (the default value is false), the user has decided to change this value, and we might need to modify the perform's dirty flag, so that the user gets prompted for a change, This is a response to GitHub issue #47, where channel changes do not cause a prompt to save the sequence.

10.91.5.112 print()

```
void seq64::sequence::print ( ) const
```

Not threadsafe

10.91.5.113 print_triggers()

```
void seq64::sequence::print_triggers ( ) const
```

Not threadsafe

10.91.5.114 play()

```
void seq64::sequence::play (
    midipulse tick,
    bool playback_mode,
    bool resume_note_ons = false )
```

This function is called by the sequencer thread, performance. The tick comes in as global tick. It turns the sequence off after we play in this frame.

Note

With pause support, the progress bar for the pattern/sequence editor does what we want: pause with the pause button, and rewind with the stop button. Works with JACK, with issues, but we'd like to have the stop button do a rewind in JACK, too.

If we are playing the song data (sequence on/off triggers, we are in playback mode. And if we are song-recording, we then keep growing the sequence's song-data triggers.

The trigger calculations have been offloaded to the [triggers::play\(\)](#) function. Its return value and side-effects tell if there's a change in playing based on triggers, and provides the ticks that bracket it.

Parameters

<i>tick</i>	Provides the current end-tick value. The tick comes in as a global tick.
<i>playback_mode</i>	Provides how playback is managed. True indicates that it is performance/song-editor playback, controlled by the set of patterns and triggers set up in that editor, and saved with the song in seq24 format. False indicates that the playback is controlled by the main window, in live mode.
<i>resume_note_ons</i>	A song-recording parameter.

Threadsafe

10.91.5.115 play_queue()

```
void seq64::sequence::play_queue (
    midipulse tick,
```

```
bool playbackmode,
bool resumenoteons )
```

We refactored this, Chris. Remember? :-D

Parameters

<i>tick</i>	Provides the current active pulse position, the tick/pulse from which to start playing.
<i>playbackmode</i>	If true, we are in Song mode. Otherwise, Live mode.
<i>resumenoteons</i>	Indicates if we are to resume Note Ons. Used by perform::play() .

10.91.5.116 add_note()

```
bool seq64::sequence::add_note (
    midipulse tick,
    midipulse len,
    int note,
    bool paint = false,
    int velocity = SEQ64_PRESERVE_VELOCITY )
```

It adds a single Note-On/Note-Off pair.

The paint parameter indicates if we care about the painted event, so then the function runs though the events and deletes the painted ones that overlap the ones we want to add.

Also note that [push_undo\(\)](#) is not incorporated into this function, for the sake of speed.

Here, we could ignore events not on the sequence's channel, as an option. We have to be careful because this function can be used in painting notes.

Stazed:

<http://www.blitter.com/~russtopia/MIDI/~jglatt/tech/midispec.htm>

Note Off: The first data is the note number. There are 128 possible notes on a MIDI device, numbered 0 to 127 (where Middle C is note number 60). This indicates which note should be released. The second data byte is the velocity, a value from 0 to 127. This indicates how quickly the note should be released (where 127 is the fastest). It's up to a MIDI device how it uses velocity information. Often velocity will be used to tailor the VCA release time. MIDI devices that can generate Note Off messages, but don't implement velocity features, will transmit Note Off messages with a preset velocity of 64.

Also, we now see that seq24 never used the recording-velocity member (`m_rec_vol`). We use it to modify the new `m_note_on_velocity` member if the user changes it in the seqedit window.

Threadsafe

Parameters

<i>tick</i>	The time destination of the new note, in pulses.
<i>len</i>	The duration of the new note, in pulses.
<i>note</i>	The pitch destination of the new note.
<i>paint</i>	If true, repaint the whole set of events, in order to be left with a clean view of the inserted event. The default is false.
<i>velocity</i>	If not set to SEQ64_PRESERVE_VELOCITY, the velocity of the note is set to this value. Otherwise, it is hard-wired to the stored note-on velocity. The name of this macro is counter-intuitive here. Currently, the note-off velocity is HARD-WIRED!

Returns

Returns true if the event was added.

10.91.5.117 add_chord()

```
bool seq64::sequence::add_chord (
    int chord,
    midipulse tick,
    midipulse len,
    int note )
```

If SEQ64_STAZED_CHORD_GENERATOR is not defined, it devolves to [add_note\(\)](#).

Todo Add the ability to preserve the incoming velocity.

*Threadsafe***Parameters**

<i>chord</i>	If greater than 0 (and less than c_chord_number), a chord (multiple notes) will be generated using this chord in the c_chord_table[] array. Otherwise, only a single note will be added.
<i>tick</i>	The time destination of the new note, in pulses.
<i>len</i>	The duration of the new note, in pulses.
<i>note</i>	The pitch destination of the new note.

Returns

Returns true if the events were added.

10.91.5.118 add_event() [1/2]

```
bool seq64::sequence::add_event (
    const event & er )
```

Then it reset the draw-marker and sets the dirty flag.

Currently, when reading a MIDI file [see the [midifile::parse\(\)](#) function], only the main events (notes, after-touch, pitch, program changes, etc.) are added with this function. So, we can rely on reading only playable events into a sequence. Well, actually, certain meta-events are also read, to obtain channel, buss, and more settings. Also read for a sequence, if the global-sequence flag is not set, are the new key, scale, and background sequence parameters.

This module (sequencer) adds all of those events as well, but it can surely add other events. We should assume that any events added by sequencer are playable/usable.

Here, we could ignore events not on the sequence's channel, as an option. We have to be careful because this function can be used in painting events.

Threadsafe

Warning

This pushing (and, in writing the MIDI file, the popping), causes events with identical timestamps to be written in reverse order. Doesn't affect functionality, but it's puzzling until one understands what is happening. Actually, this is true only in Seq24, we've fixed that behavior for Sequencer64.

Parameters

<i>er</i>	Provide a reference to the event to be added; the event is copied into the events container.
-----------	--

Returns

Returns true if the event was added.

10.91.5.119 add_event() [2/2]

```
bool seq64::sequence::add_event (
    midipulse tick,
    midibyte status,
    midibyte d0,
    midibyte d1,
    bool paint = false )
```

The paint parameter indicates if we care about the painted event, so then the function runs though the events and deletes the painted ones that overlap the ones we want to add.

Threadsafe**Parameters**

<i>tick</i>	The time destination of the event.
<i>status</i>	The type of event to add.
<i>d0</i>	The first data byte for the event.
<i>d1</i>	The second data byte for the event (if needed).
<i>paint</i>	If true, the inserted event is marked for painting.

10.91.5.120 append_event()

```
bool seq64::sequence::append_event (
    const event & er )
```

This function is meant mainly for reading the MIDI file, to save a lot of time.

Parameters

<i>er</i>	Provide a reference to the event to be added; the event is copied into the events container.
-----------	--

Returns

Returns true if the event was appended.

10.91.5.121 sort_events()

```
void seq64::sequence::sort_events ( ) [inline]
```

10.91.5.122 add_trigger()

```
void seq64::sequence::add_trigger (
    midipulse tick,
    midipulse len,
    midipulse offset = 0,
    bool fixoffset = true )
```

A pass-through function that calls [triggers::add\(\)](#). See that function for more details.

Threadsafe**Parameters**

<i>tick</i>	The time destination of the trigger.
<i>len</i>	The duration of the trigger.
<i>offset</i>	The performance offset of the trigger.
<i>fixoffset</i>	If true, adjust the offset.

10.91.5.123 split_trigger()

```
void seq64::sequence::split_trigger (
    midipulse splittick )
```

This is the public overload of `split_trigger`.

Threadsafe**Parameters**

<i>splittick</i>	The time location of the split.
------------------	---------------------------------

10.91.5.124 half_split_trigger()

```
void seq64::sequence::half_split_trigger (
    midipulse splittick )
```

Parameters

<i>splittick</i>	The time location of the split.
------------------	---------------------------------

10.91.5.125 exact_split_trigger()

```
void seq64::sequence::exact_split_trigger (
    midipulse splittick )
```

Parameters

<i>splittick</i>	The time location of the split.
------------------	---------------------------------

10.91.5.126 grow_trigger()

```
void seq64::sequence::grow_trigger (
    midipulse tickfrom,
    midipulse tickto,
    midipulse len )
```

See [triggers::grow\(\)](#) for more information.

Parameters

<i>tickfrom</i>	The desired from-value back which to expand the trigger, if necessary.
<i>tickto</i>	The desired to-value towards which to expand the trigger, if necessary.
<i>len</i>	The additional length to append to tickto for the check.

Threadsafe**10.91.5.127 delete_trigger()**

```
void seq64::sequence::delete_trigger (
    midipulse tick )
```

See [triggers::remove\(\)](#).

Threadsafe

Parameters

<i>tick</i>	Provides the tick to be used for finding the trigger to be erased.
-------------	--

10.91.5.128 get_trigger_state()

```
bool seq64::sequence::get_trigger_state (
    midipulse tick ) const
```

If any trigger is found to bracket that tick, then true is returned.

Parameters

<i>tick</i>	Provides the tick of interest.
-------------	--------------------------------

Returns

Returns true if a trigger is found that brackets the given tick.

10.91.5.129 select_trigger()

```
bool seq64::sequence::select_trigger (
    midipulse tick )
```

If any trigger is found to bracket that tick, then true is returned, and the trigger is marked as selected.

Parameters

<i>tick</i>	Provides the tick of interest.
-------------	--------------------------------

Returns

Returns true if a trigger is found that brackets the given tick; this is the return value of `m_triggers.select()`.

10.91.5.130 get_triggers()

```
triggers::List seq64::sequence::get_triggers ( ) const
```

This function is basically a threadsafe version of [sequence::triggerlist\(\)](#).

Returns

Returns of copy of `m_triggers.triggerlist()`.

10.91.5.131 unselect_trigger()

```
bool seq64::sequence::unselect_trigger (
    midipulse tick )
```

Parameters

<i>tick</i>	Indicates the trigger to be unselected.
-------------	---

10.91.5.132 unselect_triggers()

```
bool seq64::sequence::unselect_triggers ( )
```

Returns

Returns the m_triggers.unselect() return value.

10.91.5.133 intersect_triggers() [1/2]

```
bool seq64::sequence::intersect_triggers (
    midipulse position,
    midipulse & start,
    midipulse & ender )
```

If the given position is between the current trigger's tick-start and tick-end values, the these values are copied to the start and end parameters, respectively, and then we exit. See [triggers::intersect\(\)](#).

Threadsafe

Parameters

<i>position</i>	The position to examine.
<i>start</i>	The destination for the starting tick of the matching trigger.
<i>ender</i>	The destination for the ending tick of the matching trigger.

Returns

Returns true if a trigger was found whose start/end ticks contained the position. Otherwise, false is returned, and the start and end return parameters should not be used.

10.91.5.134 intersect_triggers() [2/2]

```
bool seq64::sequence::intersect_triggers (
    midipulse pos )
```

10.91.5.135 intersect_notes()

```
bool seq64::sequence::intersect_notes (
    midipulse position,
    int position_note,
    midipulse & start,
    midipulse & ender,
    int & note )
```

If the given position is between the current note's on and off time values, the these values are copied to the start and end parameters, respectively, and the note value is copied to the note parameter, and then we exit.

Threadsafe

Parameters

	<i>position</i>	The tick position to examine; where the mouse pointer is horizontally, already converted to a tick number.
	<i>position_note</i>	This is the position of the mouse pointer vertically, already converted to a note number.
out	<i>start</i>	The destination for the starting timestamp of the matching note.
out	<i>ender</i>	The destination for the ending timestamp of the matching note.
out	<i>note</i>	The destination for the note of the matching event.

Returns

Returns true if a note-on event was found whose start/end ticks contained the position. Otherwise, false is returned, and the start and end return parameters should not be used.

10.91.5.136 intersect_events()

```
bool seq64::sequence::intersect_events (
    midipulse posstart,
    midipulse posend,
    midibyte status,
    midipulse & start )
```

If the given position is between the current notes's timestamp-start and timestamp-end values, the these values are copied to the posstart and posend parameters, respectively, and then we exit.

Threadsafe

Parameters

<i>posstart</i>	The starting position to examine.
<i>posend</i>	The ending position to examine.
<i>status</i>	The desired status value.
<i>start</i>	The destination for the starting timestamp of the matching trigger.

Returns

Returns true if a event was found whose start/end timestamps contained the position. Otherwise, false is returned, and the start and end return parameters should not be used.

10.91.5.137 delete_selected_triggers()

```
void seq64::sequence::delete_selected_triggers ( )
```

10.91.5.138 cut_selected_trigger()

```
void seq64::sequence::cut_selected_trigger ( )
```

10.91.5.139 copy_selected_trigger()

```
void seq64::sequence::copy_selected_trigger ( )
```

Then it copies the first selected trigger that is found.

10.91.5.140 paste_trigger()

```
void seq64::sequence::paste_trigger (
    midipulse paste_tick = SEQ64_NO_PASTE_TRIGGER )
```

Why isn't this protected by a mutex? We will enable this if anything bad happens, such as a deadlock, or corruption, that we can prove happens here.

Parameters

<i>paste_tick</i>	A new parameter that provides the tick for pasting, or SEQ64_NO_PASTE_TRIGGER (-1) if there is none.
-------------------	--

10.91.5.141 move_triggers() [1/2]

```
bool seq64::sequence::move_triggers (
    midipulse tick,
    bool adjustoffset,
    triggers::grow_edit_t which = triggers::GROW_MOVE )
```

```
min_tick][0          1][max_tick
                2
```

The `which` parameter has three possible values:

```
-# If we are moving the 0, use first as offset.
-# If we are moving the 1, use the last as the offset.
-# If we are moving both (2), use first as offset.
```

Threadsafe

Parameters

<i>tick</i>	The tick at which the trigger starts.
<i>adjustoffset</i>	Set to true if the offset is to be adjusted.
<i>which</i>	Selects which movement will be done, as discussed above.

Returns

Returns the value of [triggers::move_selected\(\)](#), which indicates that the movement could be made. Used in [Seq24PerfInput::handle_motion_key\(\)](#).

10.91.5.142 selected_trigger_start()

```
midipulse seq64::sequence::selected_trigger_start ( )
```

Threadsafe

Returns

Returns the `tick_start()` value of the last-selected trigger. If no triggers are selected, then -1 is returned.

10.91.5.143 selected_trigger_end()

```
midipulse seq64::sequence::selected_trigger_end ( )
```

Threadsafe

Returns

Returns the `tick_end()` value of the last-selected trigger. If no triggers are selected, then -1 is returned.

10.91.5.144 get_max_trigger()

```
midipulse seq64::sequence::get_max_trigger ( ) const
```

Threadsafe

Returns

Returns the maximum trigger value.

10.91.5.145 move_triggers() [2/2]

```
void seq64::sequence::move_triggers (
    midipulse starttick,
    midipulse distance,
    bool direction )
```

Note the dependence on the `m_length` member being kept in sync with the parent's value of `m_length`.

Threadsafe

Parameters

<i>starttick</i>	The current location of the triggers.
<i>distance</i>	The distance away from the current location to which to move the triggers.
<i>direction</i>	If true, the triggers are moved forward. If false, the triggers are moved backward.

10.91.5.146 copy_triggers()

```
void seq64::sequence::copy_triggers (
    midipulse starttick,
    midipulse distance )
```

Threadsafe

Parameters

<i>starttick</i>	The current location of the triggers.
<i>distance</i>	The distance away from the current location to which to copy the triggers.

10.91.5.147 clear_triggers()

```
void seq64::sequence::clear_triggers ( )
```

Threadsafe

10.91.5.148 `get_trigger_offset()`

```
midipulse seq64::sequence::get_trigger_offset ( ) const [inline]
```

10.91.5.149 `set_midi_bus()`

```
void seq64::sequence::set_midi_bus (
    char mb,
    bool user_change = false )
```

Threadsafe

Parameters

<i>mb</i>	The MIDI buss to set as the buss number for this sequence. Also called the "MIDI port" number.
<i>user_change</i>	If true (the default value is false), the user has decided to change this value, and we might need to modify the perform's dirty flag, so that the user gets prompted for a change, This is a response to GitHub issue #47, where buss changes do not cause a prompt to save the sequence.

10.91.5.150 `get_midi_bus()`

```
char seq64::sequence::get_midi_bus ( ) const [inline]
```

10.91.5.151 `set_master_midi_bus()`

```
void seq64::sequence::set_master_midi_bus (
    mastermidibus * mmb )
```

Threadsafe

Parameters

<i>mmb</i>	Provides a pointer to the master MIDI buss for this sequence. This should be a reference, but isn't, nor is it checked.
------------	---

10.91.5.152 select_note_events()

```
int seq64::sequence::select_note_events (
    midipulse tick_s,
    int note_h,
    midipulse tick_f,
    int note_l,
    select_action_t action )
```

Be aware the the [event::is_note\(\)](#) function is used, and that it includes Aftertouch events, which generally need to stick with their Note On counterparts.

If a "note" event is detected, then we skip it. This is necessary since channel pressure and control change use d0 for seqdata, and d0 is returned by [get_note\(\)](#). This causes note selection to occasionally select them when their seqdata values are within range of the tick selection. So therefore we want only Note Ons and Note Offs.

Note

The continuation below ("continue") is necessary since channel pressure and control change use d0 for seqdata [which is returned by [get_note\(\)](#)]. This causes seqroll note selection to occasionally select them when their seqdata values are within the range of tick selection. So only, note ons and offs. What about Aftertouch? We have the [event::is_note\(\)](#) function for that.

Parameters

<i>tick_s</i>	The starting tick.
<i>note↔ _h</i>	The highest note selected.
<i>tick_f</i>	The ending, or finishing, tick.
<i>note↔ _l</i>	The lowest note selected.
<i>action</i>	The action to perform on the selection.

Returns

Returns the number of notes selected.

10.91.5.153 select_events() [1/3]

```
int seq64::sequence::select_events (
    midipulse tick_s,
    midipulse tick_f,
    midibyte status,
    midibyte cc,
    select_action_t action )
```

Note that there is also an overloaded version of this function.

Threadsafe

Parameters

<i>tick</i> ↔ <i>_s</i>	The start time of the selection.
<i>tick</i> ↔ <i>_f</i>	The finish time of the selection.
<i>status</i>	The desired event in the selection. Now, as a new feature, tempo events are also selectable, in addition to events selected by this parameter.
<i>cc</i>	The desired control-change in the selection, if the event is a control-change.
<i>action</i>	The desired selection action.

Returns

Returns the number of events selected.

10.91.5.154 select_events() [2/3]

```
int seq64::sequence::select_events (
    midibyte status,
    midibyte cc,
    bool inverse = false )
```

Note that there is also an overloaded version of this function.

Threadsafe

Warning

This used to be a void function, so it just returns 0 for now.

Parameters

<i>status</i>	Provides the status value to be selected.
<i>cc</i>	If the status is EVENT_CONTROL_CHANGE, then data byte 0 must match this value.
<i>inverse</i>	If true, invert the selection.

Returns

Always returns 0.

10.91.5.155 select_events() [3/3]

```
int seq64::sequence::select_events (
    midipulse tick_s,
    midipulse tick_f,
    midibyte status )
```


10.91.5.156 select_event_handle()

```
int seq64::sequence::select_event_handle (
    midipulse tick_s,
    midipulse tick_f,
    midibyte status,
    midibyte cc,
    int dats )
```

Parameters

<i>tick</i> ↔ _s	Provides the starting tick.
<i>tick</i> ↔ _f	Provides the ending (finishing) tick.
<i>status</i>	Provides the desired MIDI event to be selected.
<i>cc</i>	Provides the desired MIDI control value to be selected.
<i>dats</i>	Provides the center of a small data value range of plus or minus 2.

Returns

Returns the number of events selected.

10.91.5.157 select_linked()

```
int seq64::sequence::select_linked (
    long tick_s,
    long tick_f,
    midibyte status )
```

This is a Stazed selection fix we have activated unilaterally.

Parameters

<i>tick</i> ↔ _s	Provides the starting tick.
<i>tick</i> ↔ _f	Provides the ending (finishing) tick.
<i>status</i>	Provides the desired MIDI event to be selected.

Returns

Returns the number of notes selected.

10.91.5.158 select_even_or_odd_notes()

```
int seq64::sequence::select_even_or_odd_notes (
    int note_len,
    bool even )
```

Enabled only if USE_STAZED_ODD_EVEN_SELECTION is defined.

Parameters

<i>note_len</i>	The desired note lengths for the selection.
<i>even</i>	True if we want the even notes.

Returns

Returns the number of notes selected.

10.91.5.159 select_all_notes()

```
void seq64::sequence::select_all_notes (
    bool inverse = false ) [inline]
```

What about Aftertouch events? I think we need to select them as well in seqedit, so let's add that selection here as well.

Parameters

<i>inverse</i>	If set to true (the default is false), then this causes the selection to be inverted.
----------------	---

10.91.5.160 get_num_selected_notes()

```
int seq64::sequence::get_num_selected_notes ( ) const
```

Threadsafe**Returns**

Returns m_events.count_selected_notes().

10.91.5.161 get_num_selected_events()

```
int seq64::sequence::get_num_selected_events (
    midibyte status,
    midibyte cc ) const
```

If the event is a control change (CC), then it must also match the given CC value.

Threadsafe

Parameters

<i>status</i>	The desired kind of event to count.
<i>cc</i>	The desired control-change to count, if the event is a control-change.

Returns

Returns `m_events.count_selected_events()`.

10.91.5.162 select_all()

```
void seq64::sequence::select_all ( )
```

*Threadsafe***10.91.5.163 copy_selected()**

```
void seq64::sequence::copy_selected ( )
```

This function also has the danger, discovered by user Orel, of events being modified after being added to the clipboard. So we add his reconstruction fix here as well. To summarize the steps:

```
-# Clear the m_events_clipboard. NO! If we have no events to
  copy to the clipboard, we do not want to clear it. This kills
  cut-and-paste functionality.
-# Add all selected events in this clipboard to the sequence.
-# Normalize the timestamps of the events in the clip relative to the
  timestamp of the first selected event. (Is this really needed?)
-# Reconstruct/reconstitute the m_events_clipboard.
```

This process is a bit easier to manage than erase/insert on events because `std::multimap` has no `erase()` function that returns the next valid iterator. Also, we use a local clipboard first, to save on copying. We've enhanced the error-checking, too.

Finally, note that `m_events_clipboard` is a static member of `sequence`, so:

```
-# Copying can be done between sequences.
-# Access to it needs to be protected by a mutex.
```

*Threadsafe***10.91.5.164 cut_selected()**

```
void seq64::sequence::cut_selected (
    bool copyevents = true )
```

Pushes onto the undo stack, may copy the events, marks the selected events, and removes them. Now also sets the dirty flag so that the caller doesn't have to. Also raises the modify flag on the parent perform object.

Threadsafe

Parameters

<i>copyevents</i>	If true, copy the selected events before marking and removing them.
-------------------	---

10.91.5.165 `paste_selected()`

```
void seq64::sequence::paste_selected (
    midipulse tick,
    int note )
```

Also, we've moved external calls to `push_undo()` into this function. The caller shouldn't have to do that.

The `event_keys` used to access/sort the multimap `event_list` is not updated after changing timestamp/rank of the stored events. Regenerating all key/value pairs before merging them solves this issue, so that the order of events in the sequence will be preserved. This action is not needed for moving or growing events. Nor is it needed if the old `std::list` implementation of the event container is compiled in. However, it is needed in any operation that modifies the timestamp of an event inside the container:

```
- copy_selected()
- paste_selected()
- quantize_events() TODO TODO TODO!
```

The alternative to reconstructing the map is to erase-and-insert the events modified in the code above, rather than just tweaking their values, which have an effect on sorting for the event-map implementation. However, multimap does not provide an `erase()` function that returns the next valid iterator, which would complicate this method of operation. So we're inclined to stick with this solution.

There was an issue with copy/pasting a whole sequence. The pasted events did not go to their destination, but overlaid the original events. This bug also occurred in Seq24 0.9.2. It occurs with the `allofarow.mid` file when doing Ctrl-A Ctrl-C Ctrl-V Move-Mouse Left-Click. It turns out the original code was checking only the first event to see if it was a Note event. For sequences that started with a Control Change or Program Change (or other non-Note events), the highest note was never modified, and none of the note events were adjusted.

Finally, we only want to transpose note events (i.e. alter `m_data[0]`), and not other kinds of events. We still need to figure out what to do with `aftertouch`, though. Currently likely to be covered by the processing of the note that it accompanies.

Threadsafe

Parameters

<i>tick</i>	The time destination for the paste. This represents the "x" coordinate of the upper left corner of the paste-box. It will be converted to an offset, for example pasting every event 48 ticks forward from the original copy.
<i>note</i>	The note/pitch destination for the paste. This represents the "y" coordinate of the upper left corner of the paste-box. It will be converted to an offset, for example pasting every event 7 notes higher than the original copy.

10.91.5.166 `get_selected_box()`

```
void seq64::sequence::get_selected_box (
    midipulse & tick_s,
    int & note_h,
    midipulse & tick_f,
    int & note_l )
```

Note the common-code between this function and [get_clipboard_box\(\)](#). Also note we could return a boolean indicating if the return values were filled in.

Threadsafe

Parameters

out	<i>tick_s</i>	Side-effect return reference for the start time.
out	<i>note</i> ↔ <i>_h</i>	Side-effect return reference for the high note.
out	<i>tick_f</i>	Side-effect return reference for the finish time.
out	<i>note</i> ↔ <i>_l</i>	Side-effect return reference for the low note.

10.91.5.167 `get_onsets_selected_box()`

```
void seq64::sequence::get_onsets_selected_box (
    midipulse & tick_s,
    int & note_h,
    midipulse & tick_f,
    int & note_l )
```

Compare to [get_selected_box\(\)](#).

Threadsafe

Parameters

out	<i>tick_s</i>	Side-effect return reference for the start time.
out	<i>note</i> ↔ <i>_h</i>	Side-effect return reference for the high note.
out	<i>tick_f</i>	Side-effect return reference for the finish time.
out	<i>note</i> ↔ <i>_l</i>	Side-effect return reference for the low note.

10.91.5.168 `get_clipboard_box()`

```
void seq64::sequence::get_clipboard_box (
    midipulse & tick_s,
```

```

    int & note_h,
    midipulse & tick_f,
    int & note_l )

```

Note the common-code between this function and [get_selected_box\(\)](#). Also note we could return a boolean indicating if the return values were filled in.

Threadsafe

Parameters

out	<i>tick_s</i>	Side-effect return reference for the start time.
out	<i>note_↔_h</i>	Side-effect return reference for the high note.
out	<i>tick_f</i>	Side-effect return reference for the finish time.
out	<i>note_↔_l</i>	Side-effect return reference for the low note.

10.91.5.169 [adjust_timestamp\(\)](#)

```

midipulse seq64::sequence::adjust_timestamp (
    midipulse t,
    bool isnoteoff )

```

- If the timestamp plus the delta is greater than `m_length`, we do round robin magic.
- If the timestamp is greater than `m_length`, then it is wrapped around to the beginning.
- If the timestamp equals `m_length`, then it is set to 0, and later, trimmed.
- If the timestamp is less than 0, then it is set to the end.

Taken from similar code in [move_selected_notes\(\)](#) and [grow_selected\(\)](#). Be careful using this function.

Parameters

<i>t</i>	Provides the timestamp to be adjusted based on <code>m_length</code> .
<i>isnoteoff</i>	Used for "expanding" the timestamp from 0 to just less than <code>m_length</code> , if necessary. Should be set to true only for Note Off events; it defaults to false, which means to wrap the events around the end of the sequence if necessary, and is used only in movement, not in growth.

Returns

Returns the adjusted timestamp.

10.91.5.170 [trim_timestamp\(\)](#)

```

midipulse seq64::sequence::trim_timestamp (
    midipulse t )

```

Similar to `adjust_timestamp`, but it doesn't have an *isnoteoff* parameter.

Parameters

<i>t</i>	Provides the timestamp to be adjusted based on <code>m_length</code> .
----------	--

Returns

Returns the adjusted timestamp.

10.91.5.171 clip_timestamp()

```
midipulse seq64::sequence::clip_timestamp (
    midipulse ontime,
    midipulse offtime )
```

If the new (off) timestamp is less than the on-time, it is clipped to the snap value. If it is greater than the length of the sequence, then it is clipped to the sequence length. No wrap-around.

Parameters

<i>ontime</i>	Provides the original time, which limits the amount of negative adjustment that can be done.
<i>offtime</i>	Provides the timestamp to be adjusted and clipped.

Returns

Returns the adjusted timestamp.

10.91.5.172 move_selected_notes()

```
void seq64::sequence::move_selected_notes (
    midipulse delta_tick,
    int delta_note )
```

Also currently moves any other events in the range of the selection.

Also, we've moved external calls to `push_undo()` into this function. The caller shouldn't have to do that.

Another thing this function does is wrap-around when movement occurs. Any events (except Note Off) that will start just after the END of the pattern will be wrapped around to the beginning of the pattern.

Fixed:

Select all notes in a short pattern that starts at time 0 and has non-note events starting at time 0 (see `contrib/midi/allofarow.mid`); move them with the right arrow, and move them back with the left arrow; then view in the event editor, and see that the non-Note events have not moved back, and in fact move way too far to the right, actually to near the END marker. We've fixed that in the new `adjust_timestamp()` function.

This function checks for any marked events in `seq24`, but now we make sure the event is a Note On or Note Off event before dealing with it. We now handle properly events like Program Change, Control Change, and Pitch Wheel. Remember that Aftertouch is treated like a note, as it has velocity. For non-Notes, `event::get_note()` returns `m_data[0]`, and we don't want to adjust that.

Note

We leave a small gap where `mark_selected()` locks and unlocks, then we lock again. This should only be an issue if moving notes while the sequence is playing.

Parameters

<i>delta_tick</i>	Provides the amount of time to move the selected notes. Note that it also applies to events. Note-Off events are expanded to <code>m_length</code> if their timestamp would be 0. All other events will wrap around to 0.
<i>delta_note</i>	Provides the amount of pitch to move the selected notes. This value is applied only to Note (On and Off) events. Also, if this value would bring a note outside the range of 0 to 127, that note is not changed and the event is not moved.

10.91.5.173 stream_event()

```
bool seq64::sequence::stream_event (
    event & ev )
```

The event's timestamp is adjusted, if needed. If recording:

- If the pattern is playing, the event is added.
- If the pattern is playing and quantized record is in force, the note's timestamp is altered.
- If not playing, but the event is a Note On or Note Off, we add it and keep track of it.

If MIDI Thru is enabled, the event is put on the buss.

We are adding a feature where events are rejected if their channel doesn't match that of the sequence. This has been a complaint of some people. Could modify the `add_event()` and `add_note()` functions, but better to do it here for comprehensive event support. Also have to make sure the event-channel is preserved before this function is called, and also need to make sure that the channel is appended on both playback and in saving of the MIDI file.

We are also adding the usage, at last, of the `m_rec_vol` member, including the "Free" menu entry in `seqedit`, which sets the velocity to `SEQ64_PRESERVE_VELOCITY` (-1).

Todo When we feel like debugging, we will replace the global is-playing call with the parent perform's is-running call.

Threadsafe**Parameters**

<i>ev</i>	Provides the event to stream.
-----------	-------------------------------

Returns

Returns true if the event's channel matched that of this sequence, and the channel-matching feature was set to true. Also returns true if we're not using channel-matching. A return value of true means the event should be saved.

If in overwrite loop-record mode, any events after reset should clear the old items from the previous pass through the loop.

Todo If the last event was a Note Off, we should clear it here, and how?

10.91.5.174 change_event_data_range()

```
bool seq64::sequence::change_event_data_range (
    midipulse tick_s,
    midipulse tick_f,
    midibyte status,
    midibyte cc,
    int data_s,
    int data_f )
```

Changes only selected events, if any.

Threadsafe

Let t == the current tick value; ts == tick start value; tf == tick finish value; ds = data start value; df == data finish value; d = the new data value. Then

$$d = \frac{df (t - ts) + ds (tf - t)}{tf - ts}$$

If this were an interpolation formula it would be:

$$d = ds + (df - ds) \frac{t - ts}{tf - ts}$$

Something is not quite right; to be investigated.

Parameters

<i>tick</i> ↔ _s	Provides the starting tick value.
<i>tick</i> ↔ _f	Provides the ending tick value.
<i>status</i>	Provides the event status that is to be changed.
<i>cc</i>	Provides the event control value.
<i>data</i> ↔ _s	Provides the starting data value.
<i>data</i> ↔ _f	Provides the finishing data value.
Generated by Doxygen	

Returns

Returns true if the data was changed.

10.91.5.175 change_event_data_lfo()

```
void seq64::sequence::change_event_data_lfo (
    double value,
    double range,
    double speed,
    double phase,
    wave_type_t wave,
    midibyte status,
    midibyte cc )
```

Parameters

<i>value</i>	Provides the base value for the event data value. Ranges from 0 to 127 in increments of 0.1. This amount is added to the result of the wave_func() calculation.
<i>range</i>	Provides the range for the event data value. Ranges from 0 to 127 in increments of 0.1.
<i>speed</i>	Provides the inverse periodicity (?) for the modifications. Ranges from 0 to 16 in increments of 0.01. Not sure what units this value is in.
<i>phase</i>	The phase of the event modification. Ranges from 0 to 1 (what units?) in increments of 0.01.
<i>wave</i>	The wave type to apply. Ranges from 1 to 5.
<i>status</i>	The status value for the events to modify.
<i>cc</i>	Provides the control-change value for Control Change events that are to be modified.

10.91.5.176 increment_selected()

```
void seq64::sequence::increment_selected (
    midibyte astat,
    midibyte )
```

The supported statuses are:

- EVENT_NOTE_ON
- EVENT_NOTE_OFF
- EVENT_AFTERTOUCH
- EVENT_CONTROL_CHANGE
- EVENT_PITCH_WHEEL
- EVENT_PROGRAM_CHANGE
- EVENT_CHANNEL_PRESSURE

Threadsafe**Parameters**

<i>astat</i>	The desired event.
--------------	--------------------

Parameter "acontrol", the desired control-change, is unused. This might be a bug, or at least a missing feature.

10.91.5.177 decrement_selected()

```
void seq64::sequence::decrement_selected (
    midibyte astat,
    midibyte )
```

The supported statuses are:

- One-byte messages
 - EVENT_PROGRAM_CHANGE
 - EVENT_CHANNEL_PRESSURE
- Two-byte messages
 - EVENT_NOTE_ON
 - EVENT_NOTE_OFF
 - EVENT_AFTERTOUCH
 - EVENT_CONTROL_CHANGE
 - EVENT_PITCH_WHEEL

Threadsafe

Parameters

<i>astat</i>	The desired event.
--------------	--------------------

Parameter "acontrol", the desired control-change, is unused. This might be a bug, or at least a missing feature.

10.91.5.178 grow_selected()

```
void seq64::sequence::grow_selected (
    midipulse delta )
```

And, though it doesn't move Note Off events, it does reconstruct them.

This function is called when doing a ctrl-left mouse move on the selected notes or when using ctrl-left-arrow or ctrl-right-arrow to shrink or stretch the selected notes. Using the mouse allows pretty much any amount of growth or shrinkage, but use the arrow keys limits the changes to the current snap value.

This function grows/shrinks only Note On events that are marked and linked. If an event is not linked, this function now ignores the event's timestamp, rather than risk a segfault on a null pointer. Compare this function to the [stretch_selected\(\)](#) and [move_selected_notes\(\)](#) functions.

This function would strip out non-Notes, but now it at least preserves them and moves them, to try to preserve their relative position re the notes.

In any case, we want to mark the original off-event for deletion, otherwise we get duplicate off events, for example in the "Begin/End" pattern in the test.midi file.

This function now tries to prevent pathological growth, such as trying to shrink the notes to zero length or less, or stretch them beyond the length of the sequence. Otherwise we get weird and unexpected results. Also, we've moved external calls to [push_undo\(\)](#) into this function. The caller shouldn't have to do that.

A comment on terminology: The user "selects" notes, while the sequencer "marks" notes. The first thing this function does is mark all the selected notes.

Threadsafe

Parameters

<i>delta</i>	An offset for each linked event's timestamp.
--------------	--

10.91.5.179 [stretch_selected\(\)](#)

```
void seq64::sequence::stretch_selected (
    midipulse delta_tick )
```

This should move a note off event, according to old comments, but it doesn't seem to do that. See the [grow_↔selected\(\)](#) function. Rather, it moves any event in the selection.

Also, we've moved external calls to [push_undo\(\)](#) into this function. The caller shouldn't have to do that.

Threadsafe

Parameters

<i>delta_tick</i>	Provides the amount of time to stretch the selected notes.
-------------------	--

10.91.5.180 [mark_selected\(\)](#)

```
bool seq64::sequence::mark_selected ( )
```

Threadsafe

Returns

Returns true if there were any events that got marked.

10.91.5.181 [remove_selected\(\)](#)

```
void seq64::sequence::remove_selected ( )
```

This is a new convenience function to fold in the [push_undo\(\)](#) and [mark_selected\(\)](#) calls. It makes the process slightly faster, as well.

Threadsafe Also makes the whole process threadsafe.

10.91.5.182 remove_marked()

```
bool seq64::sequence::remove_marked ( )
```

Note how this function forwards the call to `m_event.remove_marked()`.

Threadsafe

Returns

Returns true if at least one event was removed.

10.91.5.183 unpaint_all()

```
void seq64::sequence::unpaint_all ( )
```

Threadsafe

10.91.5.184 unselect()

```
void seq64::sequence::unselect ( )
```

Threadsafe

10.91.5.185 verify_and_link()

```
void seq64::sequence::verify_and_link ( )
```

Threadsafe

10.91.5.186 link_new()

```
void seq64::sequence::link_new ( )
```

Threadsafe

10.91.5.187 link_tempos()

```
void seq64::sequence::link_tempos ( ) [inline]
```

10.91.5.188 zero_markers()

```
void seq64::sequence::zero_markers ( ) [inline]
```

This function is used when the sequencer stops. This function currently sets `m_last_tick = 0`, but we would like to avoid that if doing a pause, rather than a stop, of playback.

10.91.5.189 play_note_on()

```
void seq64::sequence::play_note_on (
    int note )
```

It flushes a note to the midibus to preview its sound, used by the virtual piano.

Threadsafe

Parameters

<i>note</i>	The note to play. It is not checked for range validity, for the sake of speed.
-------------	--

10.91.5.190 `play_note_off()`

```
void seq64::sequence::play_note_off (
    int note )
```

Threadsafe

Parameters

<i>note</i>	The note to turn off. It is not checked for range validity, for the sake of speed.
-------------	--

10.91.5.191 `off_playing_notes()`

```
void seq64::sequence::off_playing_notes ( )
```

This function does not bother checking if `m_master_bus` is a null pointer.

*Threadsafe*10.91.5.192 `stop()`

```
void seq64::sequence::stop (
    bool song_mode = false )
```

In Live mode, the user controls playback, while in Song mode, JACK or the performance/song editor controls playback. This function used to be called "reset()".

Parameters

<i>song_mode</i>	True if song mode is on. This can mean that JACK transport is not in control of playback.
------------------	---

10.91.5.193 `pause()`

```
void seq64::sequence::pause (
    bool song_mode = false )
```

It still includes the note-shutoff capability to prevent notes from lingering. Note that we do not call `set_playing(false)`... it disarms the sequence, which we do not want upon pausing.

10.91.5.194 inc_draw_marker()

```
void seq64::sequence::inc_draw_marker ( )
```

Warning

This iterator is shared by about four GUI object, and they might interfere with each other!

*Threadsafe***10.91.5.195 reset_draw_marker()**

```
void seq64::sequence::reset_draw_marker ( )
```

It resets the draw marker so that calls to [get_next_note_event\(\)](#) will start from the first event.

Warning

This iterator is shared by about four GUI object, and they might interfere with each other!

*Threadsafe***10.91.5.196 reset_draw_trigger_marker()**

```
void seq64::sequence::reset_draw_trigger_marker ( )
```

*Threadsafe***10.91.5.197 reset_ex_iterator()**

```
void seq64::sequence::reset_ex_iterator (
    event_list::const_iterator & evi )
```

10.91.5.198 get_next_note_event()

```
draw_type_t seq64::sequence::get_next_note_event (
    midipulse & tick_s,
    midipulse & tick_f,
    int & note,
    bool & selected,
    int & velocity )
```

When it has no more events, returns a false.

Note that, before the first call to draw a sequence, the [reset_draw_marker\(\)](#) function must be called, to reset `m_iterator_draw`.

Parameters

out	<i>tick_s</i>	Provides a pointer destination for the start time.
out	<i>tick_f</i>	Provides a pointer destination for the finish time.
out	<i>note</i>	Provides a pointer destination for the note pitch value. Probably should be a midibyte value. If the event is the special case of a tempo event, then this value is the tempo value scaled to 0 to 127 for display purposes.
out	<i>selected</i>	Provides a pointer destination for the selection status of the note.
out	<i>velocity</i>	Provides a pointer destination for the note velocity. Probably should be a midibyte value.

Returns

Returns a `draw_type_t` value: `DRAW_NORMAL_LINKED`, `DRAW_NOTE_ON`, `DRAW_NOTE_OFF`, or `DRAW_FIN`. Note that the new value `DRAW_TEMPO` could be returned, as well.

10.91.5.199 `get_minmax_note_events()`

```
bool seq64::sequence::get_minmax_note_events (
    int & lowest,
    int & highest )
```

Todo For efficiency, we should calculate this only when the event set changes, and save the results and return them if good.

Threadsafe**Parameters**

<i>lowest</i>	A reference parameter to return the note with the lowest value. if there are no notes, then it is set to <code>SEQ64_MAX_DATA_VALUE</code> , and false is returned.
<i>highest</i>	A reference parameter to return the note with the highest value. if there are no notes, then it is set to 0, and false is returned.

Returns

If there are no notes or tempo events in the list, then false is returned, and the results should be disregarded. If true is returned, but there are only tempo events, then the low/high range is 0 to 127.

10.91.5.200 `get_next_event()`

```
bool seq64::sequence::get_next_event (
    midibyte & status,
    midibyte & cc )
```

Then set the status and control character parameters using that event. This overload is used only in `seqedit::popup_event_menu()`.

Parameters

<i>status</i>	Provides a pointer to the MIDI status byte to be set, as a way to retrieve the event.
<i>cc</i>	The return pointer for the control value.

Returns

Returns true if the data is useable, and false if there are no more events.

10.91.5.201 `get_next_event_ex()`

```
bool seq64::sequence::get_next_event_ex (
    midibyte status,
    midibyte cc,
    event_list::const_iterator & evi,
    int evtype = EVENTS_ALL )
```

This version makes the caller responsible for providing and maintaining the iterator, so that there are no conflicting operations on `m_draw_iterator` from `seqdata`, `sequevent`, `seqroll`, and `perfroll`.

This rational version of `get_next_event()` returns the whole event, rather than filling in a bunch of parameters. In addition, it always allows Tempo events to be found. Gets the next event in the event list that matches the given status and control character. Then set the rest of the parameters parameters using that event. If the status is the new value `EVENT_ANY`, then any event will be obtained.

Note the usage of `event::is_desired_cc_or_not_cc(status, cc, *d0)`; Either we have a control change with the right CC or it's a different type of event.

Parameters

	<i>status</i>	The type of event to be obtained. The special value <code>EVENT_ANY</code> can be provided so that no event statuses are filtered.
	<i>cc</i>	The continuous controller value that might be desired.
out	<i>evi</i>	An iterator return value for the next event found. The caller might want to check if it is a Tempo event. Do not use this iterator if false is returned!
	<i>evtype</i>	A stazed parameter for picking either all event or unselected events. Defaults to <code>EVENTS_ALL</code> . Not used unless the macro <code>USE_STAZED_SELECTION_EXTENSIONS</code> is defined.

Returns

Returns true if the current event was one of the desired ones, or was a Tempo event. In this case, the caller *must* increment the iterator.

10.91.5.202 `get_next_trigger()`

```
bool seq64::sequence::get_next_trigger (
    midipulse & tick_on,
```

```

midipulse & tick_off,
bool & selected,
midipulse & tick_offset )

```

10.91.5.203 quantize_events()

```

void seq64::sequence::quantize_events (
    midibyte status,
    midibyte cc,
    midipulse snap_tick,
    int divide,
    bool linked = false )

```

One confusing thing is why the original versions of the events don't seem to be deleted.

Parameters

<i>status</i>	Indicates the type of event to be quantized.
<i>cc</i>	The desired control-change to count, if the event is a control-change.
<i>snap_tick</i>	Provides the maximum amount to move the events. Actually, events are moved to the previous or next snap_tick value depend on whether they are halfway to the next one or not.
<i>divide</i>	A rough indicator of the amount of quantization. The only values used in the application are either 1 ("quantize") or 2 ("tighten"). The latter value reduces the amount of change slightly.
<i>linked</i>	False by default, this parameter indicates if marked events are to be relinked, as far as we can tell.

10.91.5.204 push_quantize()

```

void seq64::sequence::push_quantize (
    midibyte status,
    midibyte cc,
    midipulse snap_tick,
    int divide,
    bool linked = false )

```

See the [sequence::quantize_events\(\)](#) function for more information. This function just does locking and a push-undo before calling that function.

Parameters

<i>status</i>	The kind of event to quantize, such as Note On, or the event type selected in the pattern editor's data pane.
<i>cc</i>	The control-change value to quantize, again as selected in the pattern editor's data pane. For Note Ons, this value should be set to 0.
<i>snap_tick</i>	The number of ticks to use for quantizing the events. Usually, this is the snap value selected in the pattern editor.
<i>divide</i>	Provides a division value, usually either 1 ("quantize") or 2 ("tighten").
<i>linked</i>	Set this value to true for tightening notes. The default value of this parameter is false.

10.91.5.205 transpose_notes()

```
void seq64::sequence::transpose_notes (
    int steps,
    int scale )
```

If the scale value is 0, this is "no scale", which is the chromatic scale, where all 12 notes, including sharps and flats, are part of the scale.

Also, we've moved external calls to [push_undo\(\)](#) into this function. The caller shouldn't have to do that.

Note

We noticed (ca 2016-06-10) that MIDI aftertouch events need to be transposed, but are not being transposed here. Assuming they are selectable (another question!), the test for note-on and note-off is not sufficient, and so has been replaced by a call to [event::is_note_msg\(\)](#).

Parameters

<i>steps</i>	The number of steps to transpose the notes.
<i>scale</i>	The scale to make the notes adhere to while transposing.

10.91.5.206 shift_notes()

```
void seq64::sequence::shift_notes (
    midipulse ticks )
```

10.91.5.207 multiply_pattern()

```
void seq64::sequence::multiply_pattern (
    double multiplier )
```

10.91.5.208 musical_key() [1/2]

```
midibyte seq64::sequence::musical_key ( ) const [inline]
```

10.91.5.209 musical_key() [2/2]

```
void seq64::sequence::musical_key (
    int key ) [inline]
```

10.91.5.210 musical_scale() [1/2]

```
midibyte seq64::sequence::musical_scale ( ) const [inline]
```

10.91.5.211 musical_scale() [2/2]

```
void seq64::sequence::musical_scale (
    int scale ) [inline]
```

10.91.5.212 background_sequence() [1/2]

```
int seq64::sequence::background_sequence ( ) const [inline]
```

10.91.5.213 background_sequence() [2/2]

```
void seq64::sequence::background_sequence (
    int bs ) [inline]
```

Disabling the sequence number (setting it to SEQ64_SEQUENCE_LIMIT) is valid.

10.91.5.214 show_events()

```
void seq64::sequence::show_events ( ) const
```

10.91.5.215 copy_events()

```
void seq64::sequence::copy_events (
    const event_list & newevents )
```

Compare this function to the [remove_all\(\)](#) function. Copying the container is a lot of work, but fairly fast, even with an `std::multimap` as the container. Also note that we have to recalculate the length of the sequence.

Threadsafe Note that we had to consolidate the replacement of all the events in the container in order to prevent the "Save to Sequence" button in the `eventedit` object from causing the application to segfault. It would segfault when the mainwnd timer callback would fire, causing updates to the sequence's slot pixmap, which would then try to access deleted events. Part of the issue was that note links were dropped when copying the events, so now we call [verify_and_link\(\)](#) to hopefully reconstitute the links.

Parameters

<i>newevents</i>	Provides the container of MIDI events that will completely replace the current container. Normally this container is supplied by the event editor, via the eventslots class.
------------------	--

10.91.5.216 note_off_margin()

```
midipulse seq64::sequence::note_off_margin ( ) const [inline]
```

10.91.5.217 set_unit_measure()

```
void seq64::sequence::set_unit_measure ( ) const
```

10.91.5.218 get_unit_measure()

```
midipulse seq64::sequence::get_unit_measure ( ) const [inline]
```

10.91.5.219 channel_match()

```
bool seq64::sequence::channel_match ( ) const [inline]
```

Compare this function to [channels_match\(\)](#).

10.91.5.220 set_overwrite_rec()

```
void seq64::sequence::set_overwrite_rec (
    bool ovwr )
```

Threadsafe

10.91.5.221 get_overwrite_rec()

```
bool seq64::sequence::get_overwrite_rec ( ) [inline]
```

10.91.5.222 set_loop_reset()

```
void seq64::sequence::set_loop_reset (
    bool reset )
```

*Threadsafe***10.91.5.223 get_loop_reset()**

```
bool seq64::sequence::get_loop_reset ( ) [inline]
```

10.91.5.224 handle_size()

```
midipulse seq64::sequence::handle_size (
    midipulse start,
    midipulse finish )
```

The s_handlesize value is an internal variable for handle size. Note that, with the default PPQN of 192, a sixteenth note (a typical snap value) is 48 pulses (ticks), so that a sixteenth note is broken into equal left, center, and right sides. However, for a PPQN of, say, 960, 16 pulses is 5 times smaller in width. We really need to scale the handle size.

Parameters

<i>start</i>	The starting tick of the note event.
<i>finish</i>	The ending tick of the note event.

Returns

Returns 16 or one-third of the note length. This value is scaled according to PPQN if different from 192.

10.91.5.225 event_in_range()

```
bool seq64::sequence::event_in_range (
    const event & e,
    midibyte status,
    midipulse tick_s,
    midipulse tick_f ) const [private]
```

Makes if-clauses easier to read.

Parameters

<i>e</i>	Provides the event to be checked.
<i>status</i>	Provides the event type that must be matched. However, Set Tempo events will always be matched.
<i>tick_s</i>	The lower end of the range of timestamps that the event must fall within.
<i>tick_f</i>	The upper end of the range of timestamps that the event must fall within.

Returns

Returns true if the event matches all of the restrictions noted.

10.91.5.226 set_parent()

```
void seq64::sequence::set_parent (
    perform * p ) [private]
```

Remember that m_parent is not at all owned by the sequence. We just don't want to do all the work necessary to make it a reference, at this time.

Parameters

<i>p</i>	A pointer to the parent, assigned only if not already assigned.
----------	---

10.91.5.227 put_event_on_bus()

```
void seq64::sequence::put_event_on_bus (
    event & ev ) [private]
```

This function does not bother checking if m_master_bus is a null pointer.

Parameters

<i>ev</i>	The event to put on the buss.
-----------	-------------------------------

Threadsafe**10.91.5.228 reset_loop()**

```
void seq64::sequence::reset_loop ( ) [private]
```

10.91.5.229 set_trigger_offset()

```
void seq64::sequence::set_trigger_offset (
    midipulse trigger_offset ) [private]
```

If m_length is 0, then m_trigger_offset is simply set to the parameter.

Threadsafe

Parameters

<i>trigger_offset</i>	The full trigger offset to set.
-----------------------	---------------------------------

10.91.5.230 `adjust_trigger_offsets_to_length()`

```
void seq64::sequence::adjust_trigger_offsets_to_length (
    midipulse newlength ) [private]
```

Threadsafe

Might can get rid of this function?

Parameters

<i>newlength</i>	The new length of the adjusted trigger.
------------------	---

10.91.5.231 `adjust_offset()`

```
midipulse seq64::sequence::adjust_offset (
    midipulse offset ) [private]
```

10.91.5.232 `remove()` [1/2]

```
void seq64::sequence::remove (
    event_list::iterator i ) [private]
```

We no longer bother checking the pointer. If it is bad, all hope is lost. If the event is a note off, and that note is currently playing, then send a note off.

Not threadsafe

Parameters

<i>i</i>	Provides the iterator to the event to remove from the event list.
----------	---

10.91.5.233 `remove()` [2/2]

```
void seq64::sequence::remove (
    event & e ) [private]
```


Finds the given event in `m_events`, and removes the first iterator matching that. If there are events that would match after that, they remain in the container. This matches `seq24` behavior.

Not threadsafe

Parameters

<code>e</code>	Provides a reference to the event to be removed.
----------------	--

10.91.5.234 `remove_all()`

```
void seq64::sequence::remove_all ( ) [private]
```

Unsets the modified flag. (Why?) Also see the new [copy_events\(\)](#) function.

10.91.5.235 `channels_match()`

```
bool seq64::sequence::channels_match (
    const event & e ) const [inline], [private]
```

Parameters

<code>e</code>	The event whose channel nybble is to be checked.
----------------	--

Returns

Returns true if the channel-matching feature is enabled and the channel matches, or true if the channel-matching feature is turned off, in which case the sequence accepts events on any channel.

10.91.5.236 `one_shot()` [2/2]

```
void seq64::sequence::one_shot (
    bool f ) [inline], [private]
```

10.91.5.237 `off_from_snap()` [2/2]

```
void seq64::sequence::off_from_snap (
    bool f ) [inline], [private]
```

10.91.5.238 song_playback_block() [2/2]

```
void seq64::sequence::song_playback_block (
    bool f ) [inline], [private]
```

10.91.5.239 song_recording() [2/2]

```
void seq64::sequence::song_recording (
    bool f ) [inline], [private]
```

10.91.5.240 song_recording_snap() [2/2]

```
void seq64::sequence::song_recording_snap (
    bool f ) [inline], [private]
```

10.91.5.241 song_record_tick() [2/2]

```
void seq64::sequence::song_record_tick (
    midipulse t ) [inline], [private]
```

10.91.6 Friends And Related Function Documentation**10.91.6.1 perform**

```
friend class perform [friend]
```

10.91.6.2 triggers

```
friend class triggers [friend]
```

10.91.7 Field Documentation

10.91.7.1 m_events_clipboard

```
event_list seq64::sequence::m_events_clipboard [static], [private]
```

Being static allows for copy/paste between patterns.

10.91.7.2 m_parent

```
perform* seq64::sequence::m_parent [private]
```

We can use the [rc_settings](#) flag(s), but JACK could be disconnected. We could use a reference here, but, to avoid modifying the midfile class as well, we use a pointer. It is set in [perform::add_sequence\(\)](#). This member would also be using for passing modification status to the parent, so that the GUI code doesn't have to do it.

10.91.7.3 m_events

```
event_list seq64::sequence::m_events [private]
```

It used to be called m_list_events, but a map implementation is now available, and is the default.

10.91.7.4 m_triggers

```
triggers seq64::sequence::m_triggers [private]
```

10.91.7.5 m_events_undo_hold

```
event_list seq64::sequence::m_events_undo_hold [private]
```

Changed, of course, from `std::list<event>` to the `sequence::Events` typedef.

```
Events m_events_undo_hold;
```

10.91.7.6 m_have_undo

```
bool seq64::sequence::m_have_undo [private]
```

10.91.7.7 m_have_redo

```
bool seq64::sequence::m_have_redo [private]
```

Previously, unlike the `perfedit`, the `seqedit` did not provide a redo facility.

10.91.7.8 m_events_undo

```
EventStack seq64::sequence::m_events_undo [private]
```

10.91.7.9 m_events_redo

```
EventStack seq64::sequence::m_events_redo [private]
```

10.91.7.10 m_iterator_draw

```
event_list::iterator seq64::sequence::m_iterator_draw [private]
```

10.91.7.11 m_channel_match

```
bool seq64::sequence::m_channel_match [private]
```

If true (not yet the default), then the seqedit window will record only MIDI events that match its channel. The old behavior is preserved if this variable is set to false.

10.91.7.12 m_midi_channel

```
midibyte seq64::sequence::m_midi_channel [private]
```

However, if this value is EVENT_NULL_CHANNEL (0xFF), then this sequence is an SMF 0 track, and has no single channel. Please note that this is the output channel.

10.91.7.13 m_bus

```
midibyte seq64::sequence::m_bus [private]
```

10.91.7.14 m_song_mute

```
bool seq64::sequence::m_song_mute [private]
```

10.91.7.15 m_transposable

```
bool seq64::sequence::m_transposable [private]
```

A potential feature from stazed's seq32 project. Now it is an actual, configurable feature.

10.91.7.16 m_notes_on

```
short seq64::sequence::m_notes_on [private]
```

We will never come close to the short limit of 32767.

10.91.7.17 m_master_bus

```
mastermidibus* seq64::sequence::m_master_bus [private]
```

10.91.7.18 m_playing_notes

```
short seq64::sequence::m_playing_notes[SEQ64_MIDI_NOTES_MAX] [private]
```

It is used when muting, to shut off the notes that are playing. The number of notes playing will never come close to the short limit of 32767.

10.91.7.19 m_was_playing

```
bool seq64::sequence::m_was_playing [private]
```

10.91.7.20 m_playing

```
bool seq64::sequence::m_playing [private]
```

10.91.7.21 m_recording

```
bool seq64::sequence::m_recording [private]
```

10.91.7.22 m_quantized_rec

```
bool seq64::sequence::m_quantized_rec [private]
```

10.91.7.23 m_thru

```
bool seq64::sequence::m_thru [private]
```

10.91.7.24 m_queued

```
bool seq64::sequence::m_queued [private]
```

10.91.7.25 m_one_shot

```
bool seq64::sequence::m_one_shot [private]
```

Set to false whenever playing-state changes. Used in sequence :: [play_queue\(\)](#) to maybe play from the one-shot tick, then toggle play and toggle queuing before playing normally.

One-shot mode is entered when the MIDI control c_status_oneshot event is received. Kepler34 reserves the period '.' to initiate this event.

10.91.7.26 m_one_shot_tick

```
midipulse seq64::sequence::m_one_shot_tick [private]
```

. Compare this member to m_queued_tick.

10.91.7.27 m_off_from_snap

```
bool seq64::sequence::m_off_from_snap [private]
```

10.91.7.28 m_song_playback_block

```
bool seq64::sequence::m_song_playback_block [private]
```

Set to false if at a trigger transition in trigger playback. Otherwise, triggers are allow to be processed. Turned off when song-recording stops.

10.91.7.29 m_song_recording

```
bool seq64::sequence::m_song_recording [private]
```

Allows recording a live performance, by storing the sequence triggers. Adapted from Kepler34.

10.91.7.30 m_song_recording_snap

```
bool seq64::sequence::m_song_recording_snap [private]
```

10.91.7.31 m_song_record_tick

```
midipulse seq64::sequence::m_song_record_tick [private]
```

10.91.7.32 m_overwrite_recording

```
bool seq64::sequence::m_overwrite_recording [private]
```

10.91.7.33 m_loop_reset

```
bool seq64::sequence::m_loop_reset [private]
```

10.91.7.34 m_unit_measure

```
midipulse seq64::sequence::m_unit_measure [mutable], [private]
```

Need to clarify this one. It is calculated when needed (lazy evaluation).

10.91.7.35 m_dirty_main

```
bool seq64::sequence::m_dirty_main [private]
```

Provides the main dirtiness flag.

10.91.7.36 m_dirty_edit

```
bool seq64::sequence::m_dirty_edit [private]
```

10.91.7.37 m_dirty_perf

```
bool seq64::sequence::m_dirty_perf [private]
```

10.91.7.38 m_dirty_names

```
bool seq64::sequence::m_dirty_names [private]
```

10.91.7.39 m_editing

```
bool seq64::sequence::m_editing [private]
```

10.91.7.40 m_raise

```
bool seq64::sequence::m_raise [private]
```

It allows a sequence editor window to pop up if not already raised, in [seqedit::timeout\(\)](#).

10.91.7.41 m_name

```
std::string seq64::sequence::m_name [private]
```

10.91.7.42 sm_default_name

```
const std::string seq64::sequence::sm_default_name [static], [private]
```

10.91.7.43 m_last_tick

```
midipulse seq64::sequence::m_last_tick [private]
```

Provides the last tick played.

10.91.7.44 m_queued_tick

```
midipulse seq64::sequence::m_queued_tick [private]
```


10.91.7.45 m_trigger_offset

```
midipulse seq64::sequence::m_trigger_offset [private]
```

10.91.7.46 m_maxbeats

```
const int seq64::sequence::m_maxbeats [private]
```

Hardwired to c_maxbeats at present.

10.91.7.47 m_ppqn

```
unsigned short seq64::sequence::m_ppqn [private]
```

10.91.7.48 m_seq_number

```
short seq64::sequence::m_seq_number [private]
```

This number is set in the [perform::install_sequence\(\)](#) function.

10.91.7.49 m_seq_color

```
colorbyte seq64::sequence::m_seq_color [private]
```

It will be an index into a palette. The colorbyte type is defined in the midibyte.hpp file.

10.91.7.50 m_seq_edit_mode

```
edit_mode_t seq64::sequence::m_seq_edit_mode [private]
```

10.91.7.51 m_length

```
midipulse seq64::sequence::m_length [private]
```

This value should be a power of two when used as a bar unit. This value depends on the settings of beats/minute, pulses/quarter-note, the beat width, and the number of measures.

10.91.7.52 m_snap_tick

```
midipulse seq64::sequence::m_snap_tick [private]
```

It starts out as the value `m_ppqn / 4`.

10.91.7.53 m_time_beats_per_measure

```
unsigned short seq64::sequence::m_time_beats_per_measure [private]
```

Defaults to 4. Used by the sequence editor to mark things in correct time on the user-interface.

10.91.7.54 m_time_beat_width

```
unsigned short seq64::sequence::m_time_beat_width [private]
```

Defaults to 4, which means the beat is a quarter note. A value of 8 would mean it is an eighth note. Used by the sequence editor to mark things in correct time on the user-interface.

10.91.7.55 m_clocks_per_metronome

```
int seq64::sequence::m_clocks_per_metronome [private]
```

This value provides the number of MIDI clocks between metronome clicks. The default value of this item is 24. It can also be read from some SMF 1 files, such as our `hymne.mid` example.

10.91.7.56 m_32nds_per_quarter

```
int seq64::sequence::m_32nds_per_quarter [private]
```

This value provides the number of notated 32nd notes in a MIDI quarter note (24 MIDI clocks). The usual (and default) value of this parameter is 8; some sequencers allow this to be changed.

10.91.7.57 m_us_per_quarter_note

```
long seq64::sequence::m_us_per_quarter_note [private]
```

This value can be extracted from the beats-per-minute value ([mastermidibus::m_beats_per_minute](#)), but here we set it to 0 by default, indicating that we don't want to write it. Otherwise, it can be read from a MIDI file, and saved here to be restored later.

10.91.7.58 m_rec_vol

```
short seq64::sequence::m_rec_vol [private]
```

It can range from 0 to 127, or be set to `SEQ64_PRESERVE_VELOCITY` (-1).

10.91.7.59 m_note_on_velocity

```
short seq64::sequence::m_note_on_velocity [private]
```

Currently set to SEQ64_DEFAULT_NOTE_ON_VELOCITY. If the recording velocity (m_rec_vol) is non-zero, this value will be set to the desired recording velocity. A "stazed" feature.

10.91.7.60 m_note_off_velocity

```
short seq64::sequence::m_note_off_velocity [private]
```

Currently set to SEQ64_DEFAULT_NOTE_OFF_VELOCITY, and currently unmodifiable. A "stazed" feature.

10.91.7.61 m_musical_key

```
midibyte seq64::sequence::m_musical_key [private]
```

If the value is SEQ64_KEY_OF_C, then there is no musical key to be set.

10.91.7.62 m_musical_scale

```
midibyte seq64::sequence::m_musical_scale [private]
```

If the value is the enumeration value c_scale_off, then there is no musical scale to be set.

10.91.7.63 m_background_sequence

```
short seq64::sequence::m_background_sequence [private]
```

If the value is greater than max_sequence(), then there is no background sequence to be set.

10.91.7.64 m_mutex

```
mutex seq64::sequence::m_mutex [mutable], [private]
```

Made mutable for use in certain locked getter functions.

10.91.7.65 m_note_off_margin

```
const midipulse seq64::sequence::m_note_off_margin [private]
```

Also used when the user attempts to shrink a note to zero (or less than zero) length.

10.92 seq64::trigger Class Reference

This class hold a single trigger for a sequence object.

Public Member Functions

- [trigger](#) ()
Initializes the trigger structure.
- bool [operator<](#) (const [trigger](#) &rhs)
This operator compares only the m_tick_start members.
- [midipulse length](#) () const
'Getter' function for member m_tick_end and m_tick_start.
- [midipulse tick_start](#) () const
'Getter' function for member m_tick_start
- void [tick_start](#) ([midipulse](#) s)
'Setter' function for member m_tick_start
- void [increment_tick_start](#) ([midipulse](#) s)
'Setter' function for member m_tick_start
- void [decrement_tick_start](#) ([midipulse](#) s)
'Setter' function for member m_tick_start
- bool [at_trigger_transition](#) ([midipulse](#) s, [midipulse](#) e)
Test if the input parameters indicate we are touching a trigger transtion.
- [midipulse tick_end](#) () const
'Getter' function for member m_tick_end
- void [tick_end](#) ([midipulse](#) e)
'Setter' function for member m_tick_end
- void [increment_tick_end](#) ([midipulse](#) s)
'Setter' function for member m_tick_end
- void [decrement_tick_end](#) ([midipulse](#) s)
'Setter' function for member m_tick_end
- [midipulse offset](#) () const
'Getter' function for member m_offset
- void [offset](#) ([midipulse](#) o)
'Setter' function for member m_offset
- void [increment_offset](#) ([midipulse](#) s)
'Setter' function for member m_offset
- void [decrement_offset](#) ([midipulse](#) s)
'Setter' function for member m_offset
- bool [selected](#) () const
'Getter' function for member m_selected
- void [selected](#) (bool s)
'Setter' function for member m_selected

Private Attributes

- [midipulse m_tick_start](#)
Provides the starting tick for this trigger.
- [midipulse m_tick_end](#)
Provides the ending tick for this trigger.
- [midipulse m_offset](#)
Provides the offset for this trigger.
- bool [m_selected](#)
Indicates that the trigger is part of a selection.

10.92.1 Detailed Description

This class is used in playback, and is contained in the triggers class.

10.92.2 Constructor & Destructor Documentation

10.92.2.1 trigger()

```
seq64::trigger::trigger ( ) [inline]
```

10.92.3 Member Function Documentation

10.92.3.1 operator<()

```
bool seq64::trigger::operator< (
    const trigger & rhs ) [inline]
```

Parameters

<i>rhs</i>	The "right-hand side" of the less-than operation.
------------	---

Returns

Returns true if m_tick_start is less than rhs's.

10.92.3.2 length()

```
midipulse seq64::trigger::length ( ) const [inline]
```

We've seen that some of the calculations of trigger length are wrong, being 1 tick less than the true length of the trigger in pulses. This function calculates trigger length the correct way.

10.92.3.3 tick_start() [1/2]

```
midipulse seq64::trigger::tick_start ( ) const [inline]
```

10.92.3.4 tick_start() [2/2]

```
void seq64::trigger::tick_start (
    midipulse s ) [inline]
```

10.92.3.5 increment_tick_start()

```
void seq64::trigger::increment_tick_start (
    midipulse s ) [inline]
```

10.92.3.6 decrement_tick_start()

```
void seq64::trigger::decrement_tick_start (
    midipulse s ) [inline]
```

10.92.3.7 at_trigger_transition()

```
bool seq64::trigger::at_trigger_transition (
    midipulse s,
    midipulse e ) [inline]
```

Parameters

s	The starting tick.
e	The ending tick.

10.92.3.8 tick_end() [1/2]

```
midipulse seq64::trigger::tick_end ( ) const [inline]
```

10.92.3.9 tick_end() [2/2]

```
void seq64::trigger::tick_end (
    midipulse e ) [inline]
```

10.92.3.10 increment_tick_end()

```
void seq64::trigger::increment_tick_end (
    midipulse s ) [inline]
```

10.92.3.11 decrement_tick_end()

```
void seq64::trigger::decrement_tick_end (
    midipulse s ) [inline]
```

10.92.3.12 offset() [1/2]

```
midipulse seq64::trigger::offset ( ) const [inline]
```

10.92.3.13 offset() [2/2]

```
void seq64::trigger::offset (
    midipulse o ) [inline]
```

10.92.3.14 increment_offset()

```
void seq64::trigger::increment_offset (
    midipulse s ) [inline]
```

10.92.3.15 decrement_offset()

```
void seq64::trigger::decrement_offset (
    midipulse s ) [inline]
```

10.92.3.16 selected() [1/2]

```
bool seq64::trigger::selected ( ) const [inline]
```

10.92.3.17 selected() [2/2]

```
void seq64::trigger::selected (
    bool s ) [inline]
```

10.92.4 Field Documentation

10.92.4.1 m_tick_start

```
midipulse seq64::trigger::m_tick_start [private]
```

10.92.4.2 m_tick_end

```
midipulse seq64::trigger::m_tick_end [private]
```

10.92.4.3 m_offset

```
midipulse seq64::trigger::m_offset [private]
```

10.92.4.4 m_selected

```
bool seq64::trigger::m_selected [private]
```

10.93 seq64::triggers Class Reference

The triggers class is a receptable the triggers that can be used with a sequence object.

Public Types

- enum [grow_edit_t](#) {
 [GROW_START](#),
 [GROW_END](#),
 [GROW_MOVE](#) }

Provides a typedef introduced by Stazed to make the trigger grow/move code easier to understand.

Public Member Functions

- [triggers](#) ([sequence](#) &parent)
Principal constructor.
- [~triggers](#) ()
A rote destructor.
- [triggers](#) & [operator=](#) (const [triggers](#) &rhs)
Principal assignment operator.
- void [set_ppqn](#) (int ppqn)
'Setter' function for member m_ppqn We have to set this value after construction for best safety.
- void [set_length](#) (int len)
'Setter' function for member m_length We have to set this value after construction for best safety.
- const [List](#) & [triggerlist](#) () const
'Getter' function for member m_triggers This is the const version
- [List](#) & [triggerlist](#) ()
'Getter' function for member m_triggers
- int [count](#) () const
'Getter' function for member m_triggers.size()
- int [number_selected](#) () const
'Getter' function for member m_number_selected
- void [push_undo](#) ()
Pushes the list-trigger into the trigger undo-list, then flags each item in the undo-list as unselected.
- void [pop_undo](#) ()
If the trigger undo-list has any items, the list-trigger is pushed into the redo list, the top of the undo-list is copied into the list-trigger, and then pops from the undo-list.
- void [pop_redo](#) ()
If the trigger redo-list has any items, the list-trigger is pushed into the undo list, the top of the redo-list is copied into the list-trigger, and then pops from the redo-list.
- void [print](#) (const std::string &seqname) const
Prints a list of the currently-held triggers.
- bool [play](#) ([midipulse](#) &starttick, [midipulse](#) &endtick, bool resume=false)
If playback-mode (song mode) is in force, that is, if using in-triggers and on/off triggers, this function handles that kind of playback.
- void [add](#) ([midipulse](#) tick, [midipulse](#) len, [midipulse](#) offset=0, bool adjustoffset=true)
Adds a trigger.
- void [adjust_offsets_to_length](#) ([midipulse](#) newlen)
Adjusts trigger offsets to the length specified for all triggers, and undo triggers.
- void [split](#) ([midipulse](#) tick)
Splits the first trigger that brackets the splittick parameter.
- void [half_split](#) ([midipulse](#) tick)
If the tick is between the start and end of this trigger...
- void [exact_split](#) ([midipulse](#) tick)
If the tick is between the start and end of this trigger...
- void [grow](#) ([midipulse](#) tickfrom, [midipulse](#) tickto, [midipulse](#) length)
Grows a trigger.
- void [remove](#) ([midipulse](#) tick)
Deletes the first trigger that brackets the given tick from the trigger-list.
- bool [get_state](#) ([midipulse](#) tick) const
Checks the list of triggers against the given tick.
- bool [select](#) ([midipulse](#) tick)
Selects the desired trigger.
- bool [unselect](#) ([midipulse](#) tick)

- Unselects the desired trigger.*
- bool `unselect ()`
 - Unselects all triggers for the sequence.*
- bool `intersect (midipulse position, midipulse &start, midipulse &end)`
 - This function examines each trigger in the trigger list.*
- bool `intersect (midipulse position)`
- void `remove_selected ()`
 - Deletes the first selected trigger that is found.*
- void `copy_selected ()`
 - Copies the first selected trigger that is found.*
- void `paste (midipulse paste_tick=SEQ64_NO_PASTE_TRIGGER)`
 - If there is a copied trigger, then this function grabs it from the trigger clipboard and adds it.*
- bool `move_selected (midipulse tick, bool adjustoffset, grow_edit_t which=GROW_MOVE)`
 - Moves selected triggers as per the given parameters.*
- midipulse `get_selected_start ()`
 - Gets the selected trigger's start tick.*
- midipulse `get_selected_end ()`
 - Gets the selected trigger's end tick.*
- midipulse `get_maximum () const`
 - Get the ending value of the last trigger in the trigger-list.*
- void `move (midipulse starttick, midipulse distance, bool direction)`
 - Moves triggers in the trigger-list.*
- void `copy (midipulse starttick, midipulse distance)`
 - Not sure what these diagrams are for yet.*
- void `clear ()`
 - Clears the whole list of triggers, and zeroes the number selected.*
- bool `next (midipulse &tick_on, midipulse &tick_off, bool &selected, midipulse &tick_offset)`
 - Get the next trigger in the trigger list, and set the parameters based on that trigger.*
- trigger `next_trigger ()`
 - Get the next trigger in the trigger list.*
- void `reset_draw_trigger_marker ()`
 - Sets the draw-trigger iterator to the beginning of the trigger list.*
- void `set_trigger_paste_tick (midipulse tick)`
- midipulse `get_trigger_paste_tick () const`

Private Types

- typedef std::list< trigger > List
 - Exposes the triggers type, currently needed for midi_container only.*
- typedef std::stack< List > Stack
 - Provides a stack for use with the undo/redo features of the trigger support.*

Private Member Functions

- midipulse `adjust_offset (midipulse offset)`
 - Adjusts the given offset by mod'ing it with m_length and adding m_length if needed, and returning the result.*
- void `split (trigger &t, midipulse splittick)`
 - Splits the trigger given by the parameter into two triggers.*
- void `select (trigger &t, bool count=true)`
 - Selects the given trigger and increments the count of selected triggers if appropriate.*
- void `unselect (trigger &t, bool count=true)`
 - Unselects the given trigger and decrements the count of selected triggers if appropriate.*

Private Attributes

- [sequence](#) & [m_parent](#)
Holds a reference to the parent sequence object that owns this trigger object.
- [List](#) [m_triggers](#)
This list holds the current pattern/triggers events.
- [int](#) [m_number_selected](#)
Holds a count of the selected triggers, for better control over selections.
- [trigger](#) [m_clipboard](#)
This item holds a single copied trigger, to be pasted later.
- [Stack](#) [m_undo_stack](#)
Handles the undo list for a series of operations on triggers.
- [Stack](#) [m_redo_stack](#)
Handles the redo list for a series of operations on triggers.
- [List::iterator](#) [m_iterator_play_trigger](#)
An iterator for cycling through the triggers during playback.
- [List::iterator](#) [m_iterator_draw_trigger](#)
An iterator for cycling through the triggers during drawing.
- [bool](#) [m_trigger_copied](#)
Set to true if there is an active trigger in the trigger clipboard.
- [midipulse](#) [m_paste_tick](#)
The tick point for pasting.
- [int](#) [m_ppqn](#)
Holds the value of the PPQN from the parent sequence, for easy access.
- [int](#) [m_length](#)
Holds the value of the length from the parent sequence, for easy access.

Friends

- class [midi_container](#)
- class [midifile](#)
- class [sequence](#)
- class [Seq24PerfInput](#)
- class [FruityPerfInput](#)

10.93.1 Member Typedef Documentation

10.93.1.1 List

```
typedef std::list<trigger> seq64::triggers::List [private]
```

10.93.1.2 Stack

```
typedef std::stack<List> seq64::triggers::Stack [private]
```

10.93.2 Member Enumeration Documentation

10.93.2.1 grow_edit_t

```
enum seq64::triggers::grow_edit_t
```

Enumerator

GROW_START	Grow the start of the trigger.
GROW_END	Grow the end of the trigger.
GROW_MOVE	Move the entire trigger block.

10.93.3 Constructor & Destructor Documentation

10.93.3.1 triggers()

```
seq64::triggers::triggers (
    sequence & parent )
```

Parameters

<i>parent</i>	The triggers object often needs to tell its parent sequence object what to do (such as stop playing).
---------------	---

10.93.3.2 ~triggers()

```
seq64::triggers::~~triggers ( )
```

10.93.4 Member Function Documentation

10.93.4.1 operator=()

```
triggers & seq64::triggers::operator= (
    const triggers & rhs )
```

Follows the stock rules for such an operator, but does a little more then just assign member values.

FIXED, BEWARE: Currently, it does not assign them all, so we should create a `partial_copy()` function to do this work, and use it where it is needed.

Parameters

<i>rhs</i>	Provides the "right-hand side" of the assignment operation.
------------	---

Returns

Returns a reference to self, for use in concatenated assignment operations.

10.93.4.2 set_ppqn()

```
void seq64::triggers::set_ppqn (
    int ppqn ) [inline]
```

10.93.4.3 set_length()

```
void seq64::triggers::set_length (
    int len ) [inline]
```

Also, there a chance that the length of the parent might change from time to time. Currently, only the sequence constructor and midifile call this function.

10.93.4.4 triggerlist() [1/2]

```
const List& seq64::triggers::triggerlist ( ) const [inline]
```

10.93.4.5 triggerlist() [2/2]

```
List& seq64::triggers::triggerlist ( ) [inline]
```

10.93.4.6 count()

```
int seq64::triggers::count ( ) const [inline]
```

10.93.4.7 number_selected()

```
int seq64::triggers::number_selected ( ) const [inline]
```

10.93.4.8 push_undo()

```
void seq64::triggers::push_undo ( )
```

10.93.4.9 pop_undo()

```
void seq64::triggers::pop_undo ( )
```

10.93.4.10 pop_redo()

```
void seq64::triggers::pop_redo ( )
```

10.93.4.11 print()

```
void seq64::triggers::print (
    const std::string & seqname ) const
```

Parameters

<i>seqname</i>	A tag name to accompany the print-out, for the human to read.
----------------	---

10.93.4.12 play()

```
bool seq64::triggers::play (
    midipulse & start_tick,
    midipulse & end_tick,
    bool resume_note_ons = false )
```

This is a new function for [sequence::play\(\)](#) to call.

The for-loop goes through all the triggers, determining if there is are trigger start/end values before the *end_tick*. If so, then the trigger state is set to true (start only within the tick range) or false (end is within the tick range), and the trigger tick is set to start or end. The first start or end trigger that is past the end tick cause the search to end.

tick_start | | **tick_end**

start_tick | | **start_tick** | | **end_tick** | | **end_tick**

Song recording:

If we've reached a new chunk of drawn sequences in the song data, and we're not recording, unset the block on this sequence's events.

If the trigger state has changed, then the start/end ticks are passed back to the sequence, and the trigger offset is adjusted.

Parameters

<i>start_tick</i>	Provides the starting tick value, and returns the modified value as a side-effect.
<i>end_tick</i>	Provides the ending tick value, and returns the modified value as a side-effect.
<i>resume_note_ons</i>	Indicates what to do with notes when song-recording.

Returns

Returns true if we're through playing the frame (trigger turning off), and the caller should stop the playback.

10.93.4.13 add()

```
void seq64::triggers::add (
    midipulse tick,
    midipulse len,
    midipulse offset = 0,
    bool fixoffset = true )
```

What is this?

```

is   ie
<   ><       ><       >
es           ee
<           >
XX
es ee
<   >
<>
es   ee
<   >
<   >
es   ee
<   >
<   >
```

Parameters

<i>tick</i>	Provides the tick (pulse) time at which the trigger goes on.
<i>len</i>	Provides the length of the trigger. This value is actually calculated from the "on" value minus the "off" value read from the MIDI file.
<i>offset</i>	This value specifies the offset of the trigger. It is a feature of the <code>c_triggers_new</code> that <code>c_triggers</code> doesn't have. It is the third value in the trigger specification of the Sequencer64 MIDI file.
<i>fixoffset</i>	If true, the offset parameter is modified by adjust_offset() first. We think that basically makes sure it is positive.

10.93.4.14 adjust_offsets_to_length()

```
void seq64::triggers::adjust_offsets_to_length (
    midipulse newlength )
```


Parameters

<i>newlength</i>	Provides the length to which to adjust the offsets.
------------------	---

COMMON CODE?

10.93.4.15 split() [1/2]

```
void seq64::triggers::split (
    midipulse splittick )
```

This is the first trigger where splittick is greater than L and less than R.

Parameters

<i>splittick</i>	Provides the tick that must be bracketed for the split to be made.
------------------	--

10.93.4.16 half_split()

```
void seq64::triggers::half_split (
    midipulse tick )
```

10.93.4.17 exact_split()

```
void seq64::triggers::exact_split (
    midipulse tick )
```

10.93.4.18 grow()

```
void seq64::triggers::grow (
    midipulse tickfrom,
    midipulse tickto,
    midipulse len )
```

This function looks for the first trigger where the tickfrom parameter is between the trigger's tick-start and tick-end values. If found then the trigger's start is moved back to tickto, if necessary, or the trigger's end is moved to tickto plus the length parameter, if necessary.

Then this new trigger is added, and the function breaks from the search loop.

Parameters

<i>tickfrom</i>	The desired from-value back which to expand the trigger, if necessary.
<i>tickto</i>	The desired to-value towards which to expand the trigger, if necessary.
<i>len</i>	The additional length to append to tickto for the check.

10.93.4.19 remove()

```
void seq64::triggers::remove (
    midipulse tick )
```

Parameters

<i>tick</i>	Provides the tick to be examined.
-------------	-----------------------------------

10.93.4.20 get_state()

```
bool seq64::triggers::get_state (
    midipulse tick ) const
```

If any trigger is found to bracket that tick, then true is returned.

Parameters

<i>tick</i>	Provides the tick of interest.
-------------	--------------------------------

Returns

Returns true if a trigger is found that brackets the given tick.

10.93.4.21 select() [1/2]

```
bool seq64::triggers::select (
    midipulse tick )
```

Checks the list of triggers against the given tick. If any trigger is found to bracket that tick, then true is returned, and the trigger is marked as selected.

Parameters

<i>tick</i>	Provides the tick of interest.
-------------	--------------------------------

Returns

Returns true if a trigger is found that brackets the given tick.

10.93.4.22 unselect() [1/3]

```
bool seq64::triggers::unselect (
    midipulse tick )
```

Checks the list of triggers against the given tick. If any trigger is found to bracket that tick, then true is returned, and the trigger is marked as unselected.

Parameters

<i>tick</i>	Provides the tick of interest.
-------------	--------------------------------

Returns

Returns true if a trigger is found that brackets the given tick.

10.93.4.23 unselect() [2/3]

```
bool seq64::triggers::unselect ( )
```

Returns

Always returns false.

10.93.4.24 intersect() [1/2]

```
bool seq64::triggers::intersect (
    midipulse position,
    midipulse & start,
    midipulse & ender )
```

If the given position is between the current trigger's tick-start and tick-end values, the these values are copied to the start and end parameters, respectively, and then we exit.

Parameters

<i>position</i>	The position to examine.
<i>start</i>	The destination for the starting tick (m_tick_start) of the matching trigger.
<i>ender</i>	The destination for the ending tick (m_tick_end) of the matching trigger.

Returns

Returns true if a trigger was found whose start/end ticks contained the position. Otherwise, false is returned, and the start and end return parameters should not be used.

10.93.4.25 intersect() [2/2]

```
bool seq64::triggers::intersect (
    midipulse position )
```

10.93.4.26 remove_selected()

```
void seq64::triggers::remove_selected ( )
```

10.93.4.27 copy_selected()

```
void seq64::triggers::copy_selected ( )
```

10.93.4.28 paste()

```
void seq64::triggers::paste (
    midipulse paste_tick = SEQ64_NO_PASTE_TRIGGER )
```

It pastes at the copy end. or at the paste-tick, if supplied.

Parameters

<i>paste_tick</i>	Provides the optional tick at which to paste the trigger. If not set to SEQ64_NO_PASTE_TRIGGER, this value is used to adjust the paste offset.
-------------------	--

10.93.4.29 move_selected()

```
bool seq64::triggers::move_selected (
    midipulse tick,
    bool fixoffset,
    grow_edit_t which = GROW_MOVE )
```

```

    mintick][0          1][maxtick
                2

```

The `which` parameter has three possible values:

```

-# If we are moving 0 (GROW_START), use first as offset.
-# If we are moving the 1 (GROW_END), use the last as the offset.
-# If we are moving both, 2 (GROW_MOVE), use first as offset.

```

Parameters

<i>tick</i>	The tick at which the trigger starts.
<i>fixoffset</i>	Set to true if the offset is to be adjusted.
<i>which</i>	Selects which movement will be done, as discussed above. See the values of the <code>trigger::grow_edit_t</code> type.

Returns

Returns true if there was room to move. Otherwise, false is returned. We need this feature to support keystroke movement of a selected trigger in the perfrull window, and keep it from continually incrementing when there can be no more movement. This causes moving the other direction to be delayed while the accumulating movement counter is used up. However, right now we can't rely on this result, and ignore it. There may be no way around this minor issue.

10.93.4.30 `get_selected_start()`

```
midipulse seq64::triggers::get_selected_start ( )
```

We guess this ends up selecting only one trigger, otherwise only the last selected one would effectively set the result.

Returns

Returns the `tick_start()` value of the last-selected trigger. If no triggers are selected, then `midipulse(-1)` is returned.

10.93.4.31 `get_selected_end()`

```
midipulse seq64::triggers::get_selected_end ( )
```

Returns

Returns the `tick_end()` value of the last-selected trigger. If no triggers are selected, then `midipulse(-1)` is returned.

10.93.4.32 get_maximum()

```
midipulse seq64::triggers::get_maximum ( ) const
```

Returns

Returns the tick-end for the last trigger, if available. Otherwise, 0 is returned.

10.93.4.33 move()

```
void seq64::triggers::move (
    midipulse starttick,
    midipulse distance,
    bool direction )
```

There's no way to optimize this by saving tick values, as they are potentially modified at each step.

Parameters

<i>starttick</i>	The current location of the triggers.
<i>distance</i>	The distance away from the current location to which to move the triggers.
<i>direction</i>	If true, the triggers are moved forward. If false, the triggers are moved backward.

10.93.4.34 copy()

```
void seq64::triggers::copy (
    midipulse starttick,
    midipulse distance )
```

```
... a
[      ][      ]
...
... a
...

5  7  play
3  offset
8  10 play

X...X...X...X...X...X...X...X...X...
L      R
[      ][      ][ ] orig
[      ][      ]

<<
[      ][ ] [ ] [ ] split on the R marker, shift first
[      ][      ][ ]
delete middle
[      ][ ] [ ] move ticks
[      ][      ]
```

```

      L      R
      [      ][ ] [      ] [ ] split on L
      [      ][      ]

      [      ]      [ ] [      ] [ ] increase all after L
      [      ]      [      ]

|...|...|...|...|...|...|...

0123456789abcdef0123456789abcdef
[      ][      ][      ][      ][      ][

[ ] [      ] [ ] [ ] [ ] [ ] [      ] [ ] [      ]
0  4      4  0 7 4 2 0      6  2
0  4      4  0 1 4 6 0      2  6 inverse offset

[      ] [      ] [      ] [      ]
[ ] [      ] [ ] [ ] [ ] [ ] [      ] [ ] [      ]
0  c      4  0 f c a 8      e  a
0  4      c  0 1 4 6 8      2  6 inverse offset

[      ] [      ] [      ] [      ]
[ ] [      ] [ ] [ ] [ ] [ ] [      ] [ ] [      ]
k   g f c a 8
0  4      c   g h k m n      inverse offset

0123456789abcdefghijklmonpq
ponmlkjihgfedcba9876543210
0fedcba9876543210fedcba9876543210fedcba9876543210fedcba9876543210

```

Copies triggers to a point distant from a given tick.

Parameters

<i>starttick</i>	The current location of the triggers.
<i>distance</i>	The distance away from the current location to which to copy the triggers.

10.93.4.35 clear()

```
void seq64::triggers::clear ( ) [inline]
```

10.93.4.36 next()

```
bool seq64::triggers::next (
    midipulse & tick_on,
    midipulse & tick_off,
    bool & selected,
    midipulse & offset )
```

Todo It would be a bit simpler to simply return a trigger object, wouldn't it?

Parameters

<i>tick_on</i>	Return value for the retrieval of the starting tick for the trigger.
<i>tick_off</i>	Return value for the retrieval of the ending tick for the trigger.
<i>selected</i>	Return value for the retrieval of the is-selected flag for the trigger.
<i>offset</i>	Return value for the retrieval of the offset for the trigger.

Returns

Returns true if a trigger was found. If false, the caller cannot rely on the values returned through the return parameters.

Side-effect(s) The value of the `m_iterator_draw_trigger` member will be altered by this call, unless pointing to the end of the triggerlist, or if there are no triggers.

10.93.4.37 next_trigger()

```
trigger seq64::triggers::next_trigger ( )
```

Returns

Returns the next trigger. If there is none, a default trigger object is returned.

10.93.4.38 reset_draw_trigger_marker()

```
void seq64::triggers::reset_draw_trigger_marker ( ) [inline]
```

10.93.4.39 set_trigger_paste_tick()

```
void seq64::triggers::set_trigger_paste_tick (
    midipulse tick ) [inline]
```

10.93.4.40 get_trigger_paste_tick()

```
midipulse seq64::triggers::get_trigger_paste_tick ( ) const [inline]
```

10.93.4.41 adjust_offset()

```
midipulse seq64::triggers::adjust_offset (
    midipulse offset ) [private]
```


Parameters

<i>offset</i>	Provides the offset, mod'ed against m_length, used to adjust the offset.
---------------	--

Returns

Returns the new offset. However, if m_length is 0, no change is made, and the original offset is returned.

10.93.4.42 split() [2/2]

```
void seq64::triggers::split (
    trigger & trig,
    midipulse splittick ) [private]
```

The original trigger ends 1 tick before the splittick parameter, and the new trigger starts at splittick and ends where the original trigger ended.

Parameters

<i>trig</i>	Provides the original trigger, and also holds the changes made to that trigger as it is shortened, as a side-effect.
<i>splittick</i>	The position just after where the original trigger will be truncated, and the new trigger begins.

10.93.4.43 select() [2/2]

```
void seq64::triggers::select (
    trigger & t,
    bool count = true ) [private]
```

Don't confuse this function with [select\(midipulse\)](#).

Parameters

<i>t</i>	Provides a reference to the desired trigger.
<i>count</i>	If true, count the selection. This can only be done in normal triggers, not triggers in the undo container.

10.93.4.44 unselect() [3/3]

```
void seq64::triggers::unselect (
    trigger & t,
    bool count = true ) [private]
```

Don't confuse this function with [unselect\(midipulse\)](#).

Parameters

<i>t</i>	Provides a reference to the desired trigger.
<i>count</i>	If true, uncount the selection. This can only be done in normal triggers, not triggers in the undo container.

10.93.5 Friends And Related Function Documentation

10.93.5.1 midi_container

```
friend class midi_container [friend]
```

10.93.5.2 midifile

```
friend class midifile [friend]
```

10.93.5.3 sequence

```
friend class sequence [friend]
```

10.93.5.4 Seq24PerfInput

```
friend class Seq24PerfInput [friend]
```

10.93.5.5 FruityPerfInput

```
friend class FruityPerfInput [friend]
```

10.93.6 Field Documentation

10.93.6.1 m_parent

`sequence& seq64::triggers::m_parent [private]`

10.93.6.2 m_triggers

`List seq64::triggers::m_triggers [private]`

10.93.6.3 m_number_selected

`int seq64::triggers::m_number_selected [private]`

10.93.6.4 m_clipboard

`trigger seq64::triggers::m_clipboard [private]`

10.93.6.5 m_undo_stack

`Stack seq64::triggers::m_undo_stack [private]`

10.93.6.6 m_redo_stack

`Stack seq64::triggers::m_redo_stack [private]`

10.93.6.7 m_iterator_play_trigger

`List::iterator seq64::triggers::m_iterator_play_trigger [private]`

10.93.6.8 m_iterator_draw_trigger

`List::iterator seq64::triggers::m_iterator_draw_trigger [private]`

10.93.6.9 m_trigger_copied

```
bool seq64::triggers::m_trigger_copied [private]
```

10.93.6.10 m_paste_tick

```
midipulse seq64::triggers::m_paste_tick [private]
```

Set to -1 if not in force. This is a new feature from stazed's Seq32 project.

10.93.6.11 m_ppqn

```
int seq64::triggers::m_ppqn [private]
```

This should not change, but we have to set it after construction, and so we provide a setter for it, [set_ppqn\(\)](#), called by the sequence constructor.

10.93.6.12 m_length

```
int seq64::triggers::m_length [private]
```

This might change, we're not yet sure.

10.94 seq64::user_instrument Class Reference

Provides data about the MIDI instruments, readable from the "user" configuration file.

Public Member Functions

- [user_instrument](#) (const std::string &name="")
Default constructor.
- [user_instrument](#) (const [user_instrument](#) &rhs)
Copy constructor.
- [user_instrument](#) & [operator=](#) (const [user_instrument](#) &rhs)
Principal assignment operator.
- bool [is_valid](#) () const
'Getter' function for member m_is_valid
- void [set_defaults](#) ()
Sets the default values.
- const std::string & [name](#) () const
'Getter' function for member m_instrument_def.instrument (name of instrument)
- int [controller_count](#) () const
'Getter' function for member m_controller_count This function returns the number of active controllers.
- int [controller_max](#) () const
'Getter' function for member MIDI_CONTROLLER_MAX This function returns the maximum number of controllers, active or inactive.
- const std::string & [controller_name](#) (int c) const
'Getter' function for member m_instrument_def.controllers[c]
- bool [controller_active](#) (int c) const
'Getter' function for member m_instrument_def.controllers_active[c]
- void [set_controller](#) (int c, const std::string &cname, bool isactive)
'Setter' function for member m_instrument_def.controllers[c] and .controllers_active[c] Only sets the controller values if the object is already valid.

Private Member Functions

- void [set_name](#) (const std::string &instname)
'Setter' function for member m_instrument_def.instrument If the name parameter is not empty, the validity flag is set to true, otherwise it is set to false.
- void [copy_definitions](#) (const [user_instrument](#) &rhs)
Copies the array members from one instance of [user_instrument](#) to this one.

Private Attributes

- bool [m_is_valid](#)
Provides a validity flag, useful in returning a reference to a bogus object for internal error-check.
- int [m_controller_count](#)
Provides the actual number of non-default controllers actually set.
- [user_instrument_t](#) [m_instrument_def](#)
The instance of the structure that this class wraps.

10.94.1 Detailed Description

Will later make the size adjustable, if it makes sense to do so.

10.94.2 Constructor & Destructor Documentation

10.94.2.1 [user_instrument\(\)](#) [1/2]

```
seq64::user_instrument::user_instrument (
    const std::string & name = "" )
```

Fills in the defaults for the instrument definition, sets its name, and provides some light validation.

Parameters

<i>name</i>	The name of the instrument, valid only if it is not empty.
-------------	--

10.94.2.2 [user_instrument\(\)](#) [2/2]

```
seq64::user_instrument::user_instrument (
    const user\_instrument & rhs )
```

Parameters

<i>rhs</i>	The sources of the data for the copy.
------------	---------------------------------------

10.94.3 Member Function Documentation

10.94.3.1 operator=()

```
user_instrument & seq64::user_instrument::operator= (
    const user_instrument & rhs )
```

Parameters

<i>rhs</i>	The sources of the data for the assignment.
------------	---

Returns

Returns a reference to this object.

10.94.3.2 is_valid()

```
bool seq64::user_instrument::is_valid ( ) const [inline]
```

10.94.3.3 set_defaults()

```
void seq64::user_instrument::set_defaults ( )
```

Also invalidates the object.

10.94.3.4 name()

```
const std::string& seq64::user_instrument::name ( ) const [inline]
```

10.94.3.5 controller_count()

```
int seq64::user_instrument::controller_count ( ) const [inline]
```

10.94.3.6 controller_max()

```
int seq64::user_instrument::controller_max ( ) const [inline]
```

Remember that the controller numbers for each MIDI instrument range from 0 to 127 (MIDI_CONTROLLER_MAX-1).

10.94.3.7 controller_name()

```
const std::string & seq64::user_instrument::controller_name (
    int c ) const
```

Parameters

<i>c</i>	The index of the desired controller.
----------	--------------------------------------

Returns

The name of the desired controller has is returned. If the index *c* is out of range, or the object is not valid, then a reference to an internal, empty string is returned.

10.94.3.8 controller_active()

```
bool seq64::user_instrument::controller_active (
    int c ) const
```

Parameters

<i>c</i>	The index of the desired controller.
----------	--------------------------------------

Returns

The status of the desired controller has is returned. If the index *c* is out of range, or the object is not valid, then false is returned.

10.94.3.9 set_controller()

```
void seq64::user_instrument::set_controller (
    int c,
    const std::string & cname,
    bool isactive )
```

Parameters

<i>c</i>	The index of the desired controller.
<i>cname</i>	The name of the controller to be set as the controller name.
<i>isactive</i>	A flag that indicates if the desired controller is active.

10.94.3.10 set_name()

```
void seq64::user_instrument::set_name (
    const std::string & instname ) [private]
```

Too tricky?

Parameters

<i>instname</i>	The name of the instrument, valid only if it is not empty.
-----------------	--

10.94.3.11 copy_definitions()

```
void seq64::user_instrument::copy_definitions (
    const user\_instrument & rhs ) [private]
```

Does not include the validity flag.

Parameters

<i>rhs</i>	The sources of the data for the partial copy.
------------	---

10.94.4 Field Documentation**10.94.4.1 m_is_valid**

```
bool seq64::user_instrument::m_is_valid [private]
```

Callers should check this flag via the [is_valid\(\)](#) accessor before using this object. This flag is set to true when any valid member assignment occurs via a public setter call. However, setting an empty name for the instrument member will render the object invalid.

10.94.4.2 m_controller_count

```
int seq64::user_instrument::m_controller_count [private]
```

Often, the "user" configuration file has only a few out of the 128 assigned explicitly.

10.94.4.3 m_instrument_def

```
user\_instrument\_t seq64::user_instrument::m_instrument_def [private]
```

10.95 seq64::user_instrument_t Struct Reference

This structure corresponds to [user-instrument-N] definitions in the `~/ .seq24usr` or `~/ .config/sequencer64/seq64usr` file.

Data Fields

- std::string [instrument](#)
Provides the name of the "instrument" being supported.
- std::string [controllers](#) [SEQ64_MIDI_CONTROLLER_MAX]
Provides a list of up to 128 controllers (e.g.
- bool [controllers_active](#) [SEQ64_MIDI_CONTROLLER_MAX]
Provides a flag that indicates if each of up to 128 controller is active and supported.

10.95.1 Field Documentation

10.95.1.1 instrument

```
std::string seq64::user_instrument_t::instrument
```

Do not confuse "instrument" with "program" here. An "instrument" is most likely a hardware MIDI sound-box (though it could be a software synthesizer as well.

10.95.1.2 controllers

```
std::string seq64::user_instrument_t::controllers[SEQ64_MIDI_CONTROLLER_MAX]
```

"Modulation"). If a controller isn't present, or if General MIDI is in force, this name might be empty.

10.95.1.3 controllers_active

```
bool seq64::user_instrument_t::controllers_active[SEQ64_MIDI_CONTROLLER_MAX]
```

If false, it might be an unsupported controller or a General MIDI device.

10.96 seq64::user_midi_bus Class Reference

Provides data about the MIDI busses, readable from the "user" configuration file.

Public Member Functions

- `user_midi_bus` (const std::string &name="")
Default constructor.
- `user_midi_bus` (const `user_midi_bus` &rhs)
Copy constructor.
- `user_midi_bus & operator=` (const `user_midi_bus` &rhs)
Principal assignment operator.
- bool `is_valid` () const
'Getter' function for member m_is_valid
- void `set_defaults` ()
Sets the default values.
- const std::string & `name` () const
'Getter' function for member m_midi_bus_def.alias (name of alias)
- int `channel_count` () const
'Getter' function for member m_channel_count
- void `channel_count` (int count)
'Setter' function for member m_channel_count
- int `channel_max` () const
'Getter' function for member SEQ64_MIDI_BUS_CHANNEL_MAX
- int `instrument` (int channel) const
'Getter' function for member m_midi_bus_def.instrument[channel]
- void `set_instrument` (int channel, int instrum)
'Getter' function for member m_midi_bus_def.instrument[channel]

Private Member Functions

- void `set_name` (const std::string &name)
'Setter' function for member m_midi_bus_def.alias (name of alias) Also sets the validity flag according to the emptiness of the name parameter.
- void `copy_definitions` (const `user_midi_bus` &rhs)
Copies the member fields from one instance of `user_midi_bus` to this one.

Private Attributes

- bool `m_is_valid`
Provides a validity flag, useful in returning a reference to a bogus object for internal error-check.
- int `m_channel_count`
Provides the actual number of non-default buss channels actually set.
- `user_midi_bus_t m_midi_bus_def`
The instance of the structure that this class wraps.

10.96.1 Detailed Description

Will later make the size adjustable, if it makes sense to do so.

10.96.2 Constructor & Destructor Documentation

10.96.2.1 `user_midi_bus()` [1/2]

```
seq64::user_midi_bus::user_midi_bus (
    const std::string & name = "" )
```

Parameters

<i>name</i>	The name of the buss, valid only if it is not empty.
-------------	--

10.96.2.2 user_midi_bus() [2/2]

```
seq64::user_midi_bus::user_midi_bus (
    const user_midi_bus & rhs )
```

Parameters

<i>rhs</i>	The sources of the data for the copy.
------------	---------------------------------------

10.96.3 Member Function Documentation

10.96.3.1 operator=()

```
user_midi_bus & seq64::user_midi_bus::operator= (
    const user_midi_bus & rhs )
```

Parameters

<i>rhs</i>	The sources of the data for the assignment.
------------	---

Returns

Returns a reference to this object.

10.96.3.2 is_valid()

```
bool seq64::user_midi_bus::is_valid ( ) const [inline]
```

10.96.3.3 set_defaults()

```
void seq64::user_midi_bus::set_defaults ( )
```

Also invalidates the object. All 16 of the channels are set to SEQ64_GM_INSTRUMENT_FLAG (-1).

10.96.3.4 name()

```
const std::string& seq64::user_midi_bus::name ( ) const [inline]
```

10.96.3.5 channel_count() [1/2]

```
int seq64::user_midi_bus::channel_count ( ) const [inline]
```

Returns

This function returns the actual number of channels. This is different from before, when the maximum number was always returned.

10.96.3.6 channel_count() [2/2]

```
void seq64::user_midi_bus::channel_count (
    int count ) [inline]
```

10.96.3.7 channel_max()

```
int seq64::user_midi_bus::channel_max ( ) const [inline]
```

Returns

Returns the maximum number of MIDI buss channels. Remember that the instrument channels for each MIDI buss range from 0 to 15 (MIDI_BUS_CHANNEL_MAX-1).

10.96.3.8 instrument()

```
int seq64::user_midi_bus::instrument (
    int channel ) const
```

Parameters

<i>channel</i>	Provides the desired buss channel number.
----------------	---

Returns

The instrument number of the desired buss channel is returned. If the channel number is out of range, or the object is not valid, then SEQ64_GM_INSTRUMENT_FLAG (-1) is returned.

10.96.3.9 set_instrument()

```
void seq64::user_midi_bus::set_instrument (
    int channel,
    int instrum )
```

Does not alter the validity flag, just checks it.

Parameters

<i>channel</i>	Provides the desired buss channel number.
<i>instrum</i>	Provides the instrument number to set that channel to.

10.96.3.10 set_name()

```
void seq64::user_midi_bus::set_name (
    const std::string & name ) [inline], [private]
```

10.96.3.11 copy_definitions()

```
void seq64::user_midi_bus::copy_definitions (
    const user_midi_bus & rhs ) [private]
```

Does not include the validity flag.

10.96.4 Field Documentation**10.96.4.1 m_is_valid**

```
bool seq64::user_midi_bus::m_is_valid [private]
```

Callers should check this flag via the [is_valid\(\)](#) accessor before using this object. This flag is set to true when any valid member assignment occurs via a public setter call.

10.96.4.2 m_channel_count

```
int seq64::user_midi_bus::m_channel_count [private]
```

Often, the "user" configuration file has only a few out of the 16 assigned explicitly.

10.96.4.3 m_midi_bus_def

```
user_midi_bus_t seq64::user_midi_bus::m_midi_bus_def [private]
```

10.97 seq64::user_midi_bus_t Struct Reference

This structure corresponds to [user-midi-bus-0] definitions in the `~/.seq24usr` ("user") file (`~/.config/sequencer64/sequencer64 usr` in the latest version of the application).

Data Fields

- `std::string alias`
Provides the user's desired name for the MIDI bus.
- `int instrument [SEQ64_MIDI_BUS_CHANNEL_MAX]`
Provides an implicit list of MIDI channels from 0 to 15 (1 to 16) and the "instrument" number assigned to each channel.

10.97.1 Field Documentation

10.97.1.1 alias

```
std::string seq64::user_midi_bus_t::alias
```

For example, "2x2 A" for some kind of MIDI card or USB MIDI cable. If `manual-alsa-ports` is enabled, this could be something like "[0] seq24 0", and that is what should be shown in that case.

10.97.1.2 instrument

```
int seq64::user_midi_bus_t::instrument [SEQ64_MIDI_BUS_CHANNEL_MAX]
```

Note that the "instrument" is not a MIDI program number. Instead, it is the number associated with a [user-instrument-definitions] section in the "user" configuration file.

10.98 seq64::user_settings Class Reference

Holds the current values of sequence settings and settings that can modify the number of sequences and the configuration of the user-interface.

Public Member Functions

- [user_settings](#) ()
Default constructor.
- [user_settings](#) (const [user_settings](#) &rhs)
Copy constructor.
- [user_settings](#) & [operator=](#) (const [user_settings](#) &rhs)
Principal assignment operator.
- void [set_defaults](#) ()
Sets the default values.
- void [normalize](#) ()
Calculate the derived values from the already-set values.
- bool [add_bus](#) (const std::string &alias)
Adds a user buss to the container, but only does so if the name parameter is not empty.
- bool [add_instrument](#) (const std::string &instname)
Adds a user instrument to the container, but only does so if the name parameter is not empty.
- const [user_midi_bus](#) & [bus](#) (int index)
'Getter' function for member Unlike the non-const version this function is public.
- const [user_instrument](#) & [instrument](#) (int index)
'Getter' function for member Unlike the non-const version this function is public.
- int [bus_count](#) () const
'Getter' function for member m_midi_buses.size()
- void [set_bus_instrument](#) (int index, int channel, int instrum)
'Getter' function for member m_midi_buses[index].instrument[channel] Currently this function is used, in the [userfile::parse\(\)](#) function.
- int [bus_instrument](#) (int buss, int channel)
'Getter' function for member m_midi_buses[buss].instrument[channel]
- const std::string & [bus_name](#) (int buss)
'Getter' function for member m_midi_buses[buss].name
- int [instrument_count](#) () const
'Getter' function for member m_instruments.size()
- void [set_instrument_controllers](#) (int index, int cc, const std::string &ccname, bool isactive)
'Setter' function for member m_midi_instrument_defs[index].controllers, controllers_active
- const std::string & [instrument_name](#) (int instrum)
'Getter' function for member m_instruments[instrument].instrument (name of instrument)
- const std::string & [instrument_name](#) (int buss, int channel)
Gets the correct instrument number from the buss and channel, and then looks up the name of the instrument.
- bool [instrument_controller_active](#) (int instrum, int cc)
'Getter' function for member m_instruments[instrument].controllers_active[controller]
- bool [controller_active](#) (int buss, int channel, int cc)
A convenience function so that the caller doesn't have to get the instrument number from the [bus_instrument\(\)](#) member function.
- const std::string & [instrument_controller_name](#) (int instrum, int cc)
'Getter' function for member m_instruments[instrument].controllers_active[controller]
- const std::string & [controller_name](#) (int buss, int channel, int cc)
'Getter' function for member m_instruments[instrument].controllers_active[controller] A convenience function so that the caller doesn't have to get the instrument number from the [bus_instrument\(\)](#) member function.
- const std::string & [comments_block](#) () const
'Getter' function for member m_comments_block
- void [clear_comments](#) ()
'Setter' function for member m_comments_block
- void [append_comment_line](#) (const std::string &line)

- 'Setter' function for member `m_comments_block`
- float `window_scale` () const
 - 'Getter' function for member `m_window_scale`
- bool `window_scaled_down` () const
 - Returns true if we're reducing the size of the main window.
- int `scale_size` (int value) const
 - 'Getter' function for member `m_window_scale`
- int `grid_style` () const
 - 'Getter' function for member `m_grid_style` Checks for normal style.
- bool `grid_is_normal` () const
 - 'Getter' function for member `m_grid_style` Checks for normal style.
- bool `grid_is_white` () const
 - 'Getter' function for member `m_grid_style` Checks for the white style.
- bool `grid_is_black` () const
 - 'Getter' function for member `m_grid_style` Checks for the black style.
- int `grid_brackets` () const
 - 'Getter' function for member `m_grid_brackets`
- int `mainwnd_rows` () const
 - 'Getter' function for member `m_mainwnd_rows`
- int `mainwnd_cols` () const
 - 'Getter' function for member `m_mainwnd_cols`
- bool `is_variset` () const
 - 'Getter' function for member `m_mainwnd_rows` and `m_mainwnd_cols` Returns true if either value is not the default.
- bool `is_default_mainwid_size` () const
 - 'Getter' function for member `m_mainwnd_rows` and `m_mainwnd_cols` Returns true if both values are the default.
- int `seqs_in_set` () const
 - 'Getter' function for member `m_seqs_in_set`, dependent member
- int `gmute_tracks` () const
 - 'Getter' function for member `m_gmute_tracks`, dependent member
- int `max_sets` () const
 - 'Getter' function for member `m_max_sets`
- int `max_sequence` () const
 - 'Getter' function for member `m_max_sequence`, dependent member
- int `text_x` () const
 - 'Getter' function for member `m_text_x`, not user modifiable, not saved
- int `text_y` () const
 - 'Getter' function for member `m_text_y`, not user modifiable, not saved
- int `seqchars_x` () const
 - 'Getter' function for member `m_seqchars_x`, not user modifiable, not saved
- int `seqchars_y` () const
 - 'Getter' function for member `m_seqchars_y`, not user modifiable, not saved
- int `seqarea_x` () const
 - 'Getter' function for member `m_seqarea_x`, not user modifiable, not saved
- int `seqarea_y` () const
 - 'Getter' function for member `m_seqarea_y`, not user modifiable, not saved
- int `seqarea_seq_x` () const
 - 'Getter' function for member `m_seqarea_seq_x`, not user modifiable, not saved
- int `seqarea_seq_y` () const
 - 'Getter' function for member `m_seqarea_seq_y`, not user modifiable, not saved
- int `mainwid_border` () const
 - 'Getter' function for member `m_mainwid_border`

- int [mainwid_spacing](#) () const
'Getter' function for member m_mainwid_spacing
- int [mainwid_x](#) () const
'Getter' function for member m_mainwid_x, dependent member
- int [mainwid_y](#) () const
'Getter' function for member m_mainwid_y, dependent member
- int [mainwnd_x](#) () const
'Getter' function for member m_mainwnd_x
- int [mainwnd_y](#) () const
'Getter' function for member m_mainwnd_y Scaled only if window scaling is less than 1.0.
- int [mainwid_border_x](#) () const
Returns the mainwid border thickness plus a fudge constant.
- int [mainwid_border_y](#) () const
Returns the mainwid border thickness plus a fudge constant.
- int [control_height](#) () const
'Getter' function for member m_control_height
- int [zoom](#) () const
'Getter' function for member m_current_zoom
- void [zoom](#) (int value)
'Setter' function for member m_current_zoom This value is not modified unless the value parameter is between 1 and 512, inclusive.
- bool [global_seq_feature](#) () const
'Getter' function for member m_global_seq_feature_save
- void [global_seq_feature](#) (bool flag)
'Setter' function for member m_global_seq_feature_save
- int [seqedit_scale](#) () const
'Getter' function for member m_seqedit_scale
- void [seqedit_scale](#) (int scale)
'Setter' function for member m_seqedit_scale
- int [seqedit_key](#) () const
'Getter' function for member m_seqedit_key
- void [seqedit_key](#) (int key)
'Setter' function for member m_seqedit_key
- int [seqedit_bgsequence](#) () const
'Getter' function for member m_seqedit_bgsequence
- void [seqedit_bgsequence](#) (int seqnum)
'Setter' function for member m_seqedit_bgsequence Note that SEQ64_IS_LEGAL_SEQUENCE() allows the SEQ64_SEQUENCE_LIMIT (0x800 = 2048) value, to turn off the use of a background sequence.
- bool [use_new_font](#) () const
'Getter' function for member m_use_new_font
- bool [allow_two_perfedits](#) () const
'Getter' function for member m_allow_two_perfedits
- int [perf_h_page_increment](#) () const
'Getter' function for member m_h_perf_page_increment
- int [perf_v_page_increment](#) () const
'Getter' function for member m_v_perf_page_increment
- int [progress_bar_colored](#) () const
'Getter' function for member m_progress_bar_colored
- bool [progress_bar_thick](#) () const
'Getter' function for member m_progress_bar_thick
- bool [inverse_colors](#) () const

Accessor *m_inverse_colors*

- int [window_redraw_rate](#) () const
'Getter' function for member m_window_redraw_rate_ms
- bool [use_more_icons](#) () const
'Getter' function for member m_use_more_icons
- int [block_rows](#) () const
'Getter' function for member m_mainwid_block_rows
- int [block_columns](#) () const
'Getter' function for member m_mainwid_block_cols
- int [block_independent](#) () const
'Getter' function for member m_mainwid_block_independent
- bool [save_user_config](#) () const
'Getter' function for member m_save_user_config
- void [save_user_config](#) (bool flag)
'Setter' function for member m_save_user_config
- int [midi_ppqn](#) () const
'Getter' function for member m_midi_ppqn
- int [midi_beats_per_bar](#) () const
'Getter' function for member m_midi_beats_per_measure
- [midibpm](#) [midi_bpm_minimum](#) () const
'Getter' function for member m_midi_bpm_minimum
- [midibpm](#) [midi_beats_per_minute](#) () const
'Getter' function for member m_midi_beats_per_minute
- [midibpm](#) [midi_bpm_maximum](#) () const
'Getter' function for member m_midi_bpm_maximum
- int [midi_beat_width](#) () const
'Getter' function for member m_midi_beat_width
- char [midi_buss_override](#) () const
'Getter' function for member m_midi_buss_override
- int [velocity_override](#) () const
'Getter' function for member m_velocity_override
- int [bpm_precision](#) () const
'Getter' function for member m_bpm_precision
- [midibpm](#) [bpm_step_increment](#) () const
'Getter' function for member m_bpm_step_increment
- [midibpm](#) [bpm_page_increment](#) () const
'Getter' function for member m_bpm_page_increment
- int [min_zoom](#) () const
'Getter' function for member mc_min_zoom
- int [max_zoom](#) () const
'Getter' function for member mc_max_zoom
- int [baseline_ppqn](#) () const
'Getter' function for member mc_baseline_ppqn
- bool [option_daemonize](#) () const
'Getter' function for member m_user_option_daemonize
- bool [option_use_logfile](#) () const
'Getter' function for member m_user_use_logfile
- std::string [option_logfile](#) () const
'Getter' function for member m_user_option_logfile
- bool [work_around_play_image](#) () const
'Getter' function for member m_work_around_play_image

- bool [work_around_transpose_image](#) () const
'Getter' function for member m_work_around_transpose_image
- int [key_height](#) () const
'Getter' function for member m_user_ui_key_height
- void [use_new_font](#) (bool flag)
'Setter' function for member m_use_new_font
- void [allow_two_perfedits](#) (bool flag)
Sets the value of allowing two perfedits to be created and shown to the user.
- void [perf_h_page_increment](#) (int inc)
Sets the horizontal page increment size for the horizontal scrollbar of a perfedit window.
- void [perf_v_page_increment](#) (int inc)
Sets the vertical page increment size for the vertical scrollbar of a perfedit window.
- void [progress_bar_colored](#) (int palcode)
'Setter' function for member m_progress_bar_colored
- void [progress_bar_thick](#) (bool flag)
'Setter' function for member m_progress_bar_thick
- void [inverse_colors](#) (bool flag)
'Setter' function for member m_inverse_colors
- void [window_redraw_rate](#) (int ms)
'Setter' function for member m_window_redraw_rate_ms
- void [use_more_icons](#) (bool flag)
'Setter' function for member m_use_more_icons
- void [block_rows](#) (int count)
'Setter' function for member m_mainwid_block_rows
- void [block_columns](#) (int count)
'Setter' function for member m_mainwid_block_cols
- void [block_independent](#) (bool flag)
'Setter' function for member m_mainwid_block_independent
- void [option_daemonize](#) (bool flag)
'Setter' function for member m_user_option_daemonize
- void [option_use_logfile](#) (bool flag)
'Setter' function for member m_user_use_logfile
- void [option_logfile](#) (const std::string &logfile)
'Setter' function for member m_user_option_logfile
- void [work_around_play_image](#) (bool flag)
'Setter' function for member m_work_around_play_image
- void [work_around_transpose_image](#) (bool flag)
'Setter' function for member m_work_around_transpose_image
- void [key_height](#) (int h)
'Setter' function for member m_user_ui_key_height Do we want to add scaling to this at this time? m_window_scale
- void [midi_ppqn](#) (int ppqn)
'Setter' function for member m_midi_ppqn This value can be set from 96 to 19200 (this upper limit will be determined by what Sequencer64 can actually handle).
- void [midi_buss_override](#) (char buss)
'Setter' function for member m_midi_buss_override This value can be set from 0 to 31.
- void [velocity_override](#) (int vel)
'Setter' function for member m_velocity_override
- void [bpm_precision](#) (int precision)
'Setter' function for member m_bpm_precision
- void [bpm_step_increment](#) (midibpm increment)
'Setter' function for member m_bpm_step_increment

- void [bpm_page_increment](#) (midibpm increment)
'Setter' function for member m_bpm_page_increment
- int [mainwid_width](#) () const
Replaces the hard-wired calculation in the mainwid module.
- int [mainwid_height](#) () const
Replaces the hard-wired calculation in the mainwid module.
- int [mainwid_width_fudge](#) () const
'Getter' function for member MAINWID_WIDTH_FUDGE / 2
- int [mainwid_height_fudge](#) () const
'Getter' function for member MAINWID_HEIGHT_FUDGE / 2

Protected Member Functions

- void [grid_brackets](#) (int thickness)
'Getter' function for member m_grid_brackets
- void [window_scale](#) (float winscale)
'Setter' function for member m_window_scale
- void [grid_style](#) (int gridstyle)
'Setter' function for member m_grid_style
- void [mainwnd_rows](#) (int value)
'Setter' function for member m_mainwnd_rows This value is not modified unless the value parameter is between 4 and 8, inclusive.
- void [mainwnd_cols](#) (int value)
'Setter' function for member m_mainwnd_cols This value is not modified unless the value parameter is between 8 and 8, inclusive.
- void [max_sets](#) (int value)
'Setter' function for member m_max_sets This value is not modified unless the value parameter is between 16 and 32.
- void [text_x](#) (int value)
'Setter' function for member m_text_x This value is not modified unless the value parameter is between 6 and 6, inclusive.
- void [text_y](#) (int value)
'Setter' function for member m_text_y This value is not modified unless the value parameter is between 12 and 12, inclusive.
- void [seqchars_x](#) (int value)
'Setter' function for member m_seqchars_x This affects the size or crampiness of a pattern slot, and for now we will hardwire it to 15.
- void [seqchars_y](#) (int value)
'Setter' function for member m_seqchars_y This affects the size or crampiness of a pattern slot, and for now we will hardwire it to 5.
- void [seqarea_x](#) (int value)
'Setter' function for member m_seqarea_x
- void [seqarea_y](#) (int value)
'Setter' function for member m_seqarea_y
- void [seqarea_seq_x](#) (int value)
'Setter' function for member m_seqarea_seq_x
- void [seqarea_seq_y](#) (int value)
'Setter' function for member m_seqarea_seq_y
- void [mainwid_border](#) (int value)
'Setter' function for member m_mainwid_border This value is not modified unless the value parameter is between 0 and 3, inclusive.
- void [mainwid_spacing](#) (int value)

'Setter' function for member `m_mainwid_spacing` This value is not modified unless the value parameter is between 2 and 6, inclusive.

- void `control_height` (int value)

'Setter' function for member `m_control_height` This value is not modified unless the value parameter is between 0 and 4, inclusive.

- void `dump_summary` ()

Provides a debug dump of basic information to help debug a surprisingly intractable problem with all busses having the name and values of the last buss in the configuration.

- void `midi_beats_per_bar` (int beatsperbar)

'Setter' function for member `m_midi_beats_per_measure` This value can be set from 1 to 16.

- void `midi_bpm_minimum` (midibpm beatsperminute)

'Setter' function for member `m_midi_bpm_minimum` This value can be set from 20 to 500.

- void `midi_beats_per_minute` (midibpm beatsperminute)

'Setter' function for member `m_midi_beats_minute` This value can be set from 20 to 500.

- void `midi_bpm_maximum` (midibpm beatsperminute)

'Setter' function for member `m_midi_bpm_maximum` This value can be set from 20 to 500.

- void `midi_beat_width` (int beatwidth)

'Setter' function for member `m_midi_beatwidth` This value can be set to any power of 2 in the range from 1 to 16.

Private Types

- enum `mainwid_grid_style_t` {
`grid_style_normal`,
`grid_style_white`,
`grid_style_black`,
`grid_style_max` }
- typedef std::vector< `user_midi_bus` > `Busses`
[user-midi-bus-definitions]
- typedef std::vector< `user_midi_bus` >::iterator `BussIterator`
- typedef std::vector< `user_midi_bus` >::const_iterator `BussConstIterator`
- typedef std::vector< `user_instrument` > `Instruments`
[user-instrument-definitions]
- typedef std::vector< `user_instrument` >::iterator `InstrumentIterator`
- typedef std::vector< `user_instrument` >::const_iterator `InstrumentConstIterator`

Private Member Functions

- `user_midi_bus` & `private_bus` (int buss)

'Getter' function for member `m_midi_buses[index]` (internal function) If the index is out of range, then an invalid object is returned.

- `user_instrument` & `private_instrument` (int instrum)

'Getter' function for member `m_instruments[index]` If the index is out of range, then a invalid object is returned.

Private Attributes

- [Busses m_midi_buses](#)
Provides data about the MIDI busses, readable from the "user" configuration file.
- [Instruments m_instruments](#)
Provides data about the MIDI instruments, readable from the "user" configuration file.
- `std::string` [m_comments_block](#)
[comments]
- `mainwid_grid_style_t` [m_grid_style](#)
[user-interface-settings]
- `int` [m_grid_brackets](#)
Specify drawing brackets (like the old Seq24) or a solid box.
- `int` [m_mainwnd_rows](#)
Number of rows in the Patterns Panel.
- `int` [m_mainwnd_cols](#)
Number of columns in the Patterns Panel.
- `int` [m_max_sets](#)
Maximum number of screen sets that can be supported.
- `float` [m_window_scale](#)
Provide a scale factor to increase the size of the main window and its internals.
- `int` [m_mainwid_border](#)
These control sizes.
- `int` [m_mainwid_spacing](#)
- `int` [m_control_height](#)
This constants seems to be created for a future purpose, perhaps to reserve space for a new bar on the mainwid pane.
- `int` [m_current_zoom](#)
Provides the initial zoom value, in units of ticks per pixel.
- `bool` [m_global_seq_feature_save](#)
If true, this value provide a bit of backward-compatibility with the global key/scale/background-sequence persistence feature.
- `int` [m_seqedit_scale](#)
Replaces seqedit::m_initial_scale as the repository for the scale to apply when a sequence is loaded into the sequence editor.
- `int` [m_seqedit_key](#)
Replaces seqedit::m_initial_key as the repository for the key to apply when a sequence is loaded into the sequence editor.
- `int` [m_seqedit_bgsequence](#)
Replaces seqedit::m_initial_sequence as the repository for the background sequence to apply when a sequence is loaded into the sequence editor.
- `bool` [m_use_new_font](#)
Sets the usage of the font.
- `bool` [m_allow_two_perfedits](#)
Enables the usage of two perfedit windows, for added convenience in editing multi-set songs.
- `int` [m_h_perf_page_increment](#)
Allows a changed to the page size for the horizontal scroll bar.
- `int` [m_v_perf_page_increment](#)
Allows a changed to the page size for the vertical scroll bar.
- `int` [m_progress_bar_colored](#)
If set, makes progress bars have the "progress_color()", instead of black.
- `bool` [m_progress_bar_thick](#)
If set, makes progress bars thicker than 1 pixel...

- bool [m_inverse_colors](#)
If set, use an alternate, neo-inverse color palette.
- int [m_window_redraw_rate_ms](#)
Provides the global setting for redraw rate of windows.
- bool [m_use_more_icons](#)
Another [user-interface-settings] item.
- int [m_mainwid_block_rows](#)
New section [user-main-window].
- int [m_mainwid_block_cols](#)
This value specifies the number of columns of main windows.
- bool [m_mainwid_block_independent](#)
If true, this value will enable individual set-controls for the multiple mainwid objects shown in the main window.
- int [m_text_x](#)
Constants for the mainwid class.
- int [m_text_y](#)
- int [m_seqchars_x](#)
Constants for the mainwid class.
- int [m_seqchars_y](#)
- int [m_midi_ppqn](#)
Provides the universal PPQN setting for the duration of this setting.
- int [m_midi_beats_per_measure](#)
Provides the universal and unambiguous MIDI value for beats per measure, also called "beats per bar" (BPB).
- [midibpm m_midi_bpm_minimum](#)
Provides the minimum beats per minute, purely for providing the scale for drawing the tempo.
- [midibpm m_midi_beats_per_minute](#)
Provides the universal and unambiguous MIDI value for beats per minute (BPM).
- [midibpm m_midi_bpm_maximum](#)
Provides the maximum beats per minute, purely for providing the scale for drawing the tempo.
- int [m_midi_beat_width](#)
Provides the universal MIDI value for beats width (BW).
- char [m_midi_buss_override](#)
Provides a universal override of the buss number for all sequences, for the purpose of convenience of testing.
- int [m_velocity_override](#)
Sets the default velocity for note adding.
- int [m_bpm_precision](#)
Sets the precision of the BPM (beats-per-minute) setting.
- [midibpm m_bpm_step_increment](#)
The step increment value for BPM, regardless of the decimal precision.
- [midibpm m_bpm_page_increment](#)
This is the larger increment for paging the BPM.
- int [m_total_seqs](#)
The maximum number of patterns supported is given by the number of patterns supported in the panel (32) times the maximum number of sets (32), or 1024 patterns.
- int [m_seqs_in_set](#)
Number of patterns/sequences in the Patterns Panel, also known as a "set" or "screen set".
- int [m_gmute_tracks](#)
Number of group-mute tracks/sequences/patterns that can be supported, which is m_seqs_in_set squared, or 1024.
- int [m_max_sequence](#)
The maximum number of patterns supported is given by the number of patterns supported in the panel (32) times the maximum number of sets (32), or 1024 patterns.
- int [m_seqarea_x](#)

The `m_seqarea_x` and `m_seqarea_y` constants are derived from the width and heights of the default character set, and the number of characters in width, and the number of lines, in a pattern/sequence box.

- int `m_seqarea_y`
- int `m_seqarea_seq_x`

These values delineate the smaller rectangle inside of a mainwid cell, wherein the sequence events are drawn.

- int `m_seqarea_seq_y`
- int `m_mainwid_x`

The width of the main pattern/sequence grid, in pixels.

- int `m_mainwid_y`

The height of the main pattern/sequence grid, in pixels.

- int `m_mainwnd_x`

The hardwired base width of the whole main window.

- int `m_mainwnd_y`

The hardwired base height of the whole main window.

- bool `m_save_user_config`

Provides a temporary variable that can be set from the command line to cause the "user" state to be saved into the "user" configuration file.

- const int `mc_min_zoom`

Provides the minimum zoom value, currently a constant.

- const int `mc_max_zoom`

Provides the maximum zoom value, currently a constant.

- const int `mc_baseline_ppqn`

Permanent storage for the baseline, default PPQN used by Seq24.

- bool `m_user_option_daemonize`

Indicates if the application should be daemonized.

- bool `m_user_use_logfile`

If true, this value means that "-o log=..." (where the "..." is an optional filename) was specified on the command line.

- std::string `m_user_option_logfile`

If not empty, this file will be set up as the destination for all logging done by the `errprint()`, `infoprint()`, `warnprint()`, and `printf()` functions.

- bool `m_work_around_play_image`

We have an issue on some user's machines where toggling the image on the play button from the "play" image to the "pause" images causes segfaults.

- bool `m_work_around_transpose_image`

Another similar issue occurs in setting the transposable image in `seqedit`, even though there should be no thread conflicts! Weird.

- int `m_user_ui_key_height`

Defines the key height in the Kepler34 sequence editor.

Friends

- class `midifile`
- class `userfile`
- bool `parse_o_options` (int, char *[])

Checks the putative command-line arguments for any of the new "options" options.

10.98.1 Detailed Description

These settings will eventually be made part of the "user" settings file.

10.98.2 Member Typedef Documentation

10.98.2.1 Busses

```
typedef std::vector<user_midi_bus> seq64::user_settings::Busses [private]
```

Internal types for the container of `user_midi_bus` objects. Sorry about the "confusion" about "bus" versus "buss". See Google for arguments about it.

10.98.2.2 BussIterator

```
typedef std::vector<user_midi_bus>::iterator seq64::user_settings::BussIterator [private]
```

10.98.2.3 BussConstIterator

```
typedef std::vector<user_midi_bus>::const_iterator seq64::user_settings::BussConstIterator  
[private]
```

10.98.2.4 Instruments

```
typedef std::vector<user_instrument> seq64::user_settings::Instruments [private]
```

Internal type for the container of `user_instrument` objects.

10.98.2.5 InstrumentIterator

```
typedef std::vector<user_instrument>::iterator seq64::user_settings::InstrumentIterator [private]
```

10.98.2.6 InstrumentConstIterator

```
typedef std::vector<user_instrument>::const_iterator seq64::user_settings::InstrumentConst↵  
Iterator [private]
```

10.98.3 Member Enumeration Documentation

10.98.3.1 mainwid_grid_style_t

```
enum seq64::user_settings::mainwid_grid_style_t [private]
```

Enumerator

grid_style_normal	The grid background color is white. This style better fits displaying the white-on-black sequence numbers. The box is drawn with brackets on either side.	
grid_style_black		The grid background color is black.
grid_style_max		Marks the end of the list, and is an illegal

10.98.4 Constructor & Destructor Documentation

10.98.4.1 user_settings() [1/2]

```
seq64::user_settings::user_settings ( )
```

10.98.4.2 user_settings() [2/2]

```
seq64::user_settings::user_settings (
    const user_settings & rhs )
```

10.98.5 Member Function Documentation

10.98.5.1 operator=()

```
user_settings & seq64::user_settings::operator= (
    const user_settings & rhs )
```

10.98.5.2 set_defaults()

```
void seq64::user_settings::set_defaults ( )
```

For the `m_midi_buses` and `m_instruments` members, this function can only iterate over the current size of the vectors. But the default size is zero!

10.98.5.3 normalize()

```
void seq64::user_settings::normalize ( )
```

Should we normalize the BPM increment values here, in case they are irregular?

[gmute_tracks\(\)](#) is viable with variable set sizes only if we stick with the 32 sets by 32 patterns, at this time. It's semantic meaning is... TODO!!

m_max_sequence is now actually a constant (1024), so we enforce that here now.

10.98.5.4 add_bus()

```
bool seq64::user_settings::add_bus (
    const std::string & alias )
```

10.98.5.5 add_instrument()

```
bool seq64::user_settings::add_instrument (
    const std::string & instname )
```

10.98.5.6 bus()

```
const user_midi_bus& seq64::user_settings::bus (
    int index ) [inline]
```

Cannot append the const specifier.

10.98.5.7 instrument()

```
const user_instrument& seq64::user_settings::instrument (
    int index ) [inline]
```

Cannot append the const specifier.

10.98.5.8 bus_count()

```
int seq64::user_settings::bus_count ( ) const [inline]
```

10.98.5.9 set_bus_instrument()

```
void seq64::user_settings::set_bus_instrument (
    int index,
    int channel,
    int instrum )
```

10.98.5.10 bus_instrument()

```
int seq64::user_settings::bus_instrument (
    int buss,
    int channel ) [inline]
```

10.98.5.11 bus_name()

```
const std::string& seq64::user_settings::bus_name (
    int buss ) [inline]
```

10.98.5.12 instrument_count()

```
int seq64::user_settings::instrument_count ( ) const [inline]
```

10.98.5.13 set_instrument_controllers()

```
void seq64::user_settings::set_instrument_controllers (
    int index,
    int cc,
    const std::string & ccname,
    bool isactive )
```

10.98.5.14 instrument_name() [1/2]

```
const std::string& seq64::user_settings::instrument_name (
    int instrum ) [inline]
```

10.98.5.15 instrument_name() [2/2]

```
const std::string& seq64::user_settings::instrument_name (
    int buss,
    int channel ) [inline]
```

10.98.5.16 instrument_controller_active()

```
bool seq64::user_settings::instrument_controller_active (
    int instrum,
    int cc ) [inline]
```

10.98.5.17 controller_active()

```
bool seq64::user_settings::controller_active (
    int buss,
    int channel,
    int cc ) [inline]
```

It also has a shorter name.

10.98.5.18 instrument_controller_name()

```
const std::string& seq64::user_settings::instrument_controller_name (
    int instrum,
    int cc ) [inline]
```

10.98.5.19 controller_name()

```
const std::string& seq64::user_settings::controller_name (
    int buss,
    int channel,
    int cc ) [inline]
```

It also has a shorter name.

10.98.5.20 comments_block()

```
const std::string& seq64::user_settings::comments_block ( ) const [inline]
```

10.98.5.21 clear_comments()

```
void seq64::user_settings::clear_comments ( ) [inline]
```

10.98.5.22 append_comment_line()

```
void seq64::user_settings::append_comment_line (
    const std::string & line ) [inline]
```

10.98.5.23 window_scale() [1/2]

```
float seq64::user_settings::window_scale ( ) const [inline]
```

10.98.5.24 window_scaled_down()

```
bool seq64::user_settings::window_scaled_down ( ) const [inline]
```

10.98.5.25 scale_size()

```
int seq64::user_settings::scale_size (
    int value ) const [inline]
```

10.98.5.26 grid_style() [1/2]

```
int seq64::user_settings::grid_style ( ) const [inline]
```

10.98.5.27 grid_is_normal()

```
bool seq64::user_settings::grid_is_normal ( ) const [inline]
```

10.98.5.28 grid_is_white()

```
bool seq64::user_settings::grid_is_white ( ) const [inline]
```

10.98.5.29 grid_is_black()

```
bool seq64::user_settings::grid_is_black ( ) const [inline]
```

10.98.5.30 grid_brackets() [1/2]

```
int seq64::user_settings::grid_brackets ( ) const [inline]
```

10.98.5.31 mainwnd_rows() [1/2]

```
int seq64::user_settings::mainwnd_rows ( ) const [inline]
```

10.98.5.32 mainwnd_cols() [1/2]

```
int seq64::user_settings::mainwnd_cols ( ) const [inline]
```

10.98.5.33 is_variset()

```
bool seq64::user_settings::is_variset ( ) const [inline]
```

This function is the inverse of `is_default_m.inwid_size()`.

10.98.5.34 is_default_mainwid_size()

```
bool seq64::user_settings::is_default_mainwid_size ( ) const [inline]
```

This function is the inverse of [is_variset\(\)](#).

10.98.5.35 seqs_in_set()

```
int seq64::user_settings::seqs_in_set ( ) const [inline]
```

10.98.5.36 gmute_tracks()

```
int seq64::user_settings::gmute_tracks ( ) const [inline]
```

10.98.5.37 max_sets() [1/2]

```
int seq64::user_settings::max_sets ( ) const [inline]
```

10.98.5.38 max_sequence()

```
int seq64::user_settings::max_sequence ( ) const [inline]
```

10.98.5.39 text_x() [1/2]

```
int seq64::user_settings::text_x ( ) const [inline]
```

10.98.5.40 text_y() [1/2]

```
int seq64::user_settings::text_y ( ) const [inline]
```

10.98.5.41 seqchars_x() [1/2]

```
int seq64::user_settings::seqchars_x ( ) const [inline]
```

10.98.5.42 seqchars_y() [1/2]

```
int seq64::user_settings::seqchars_y ( ) const [inline]
```

10.98.5.43 seqarea_x() [1/2]

```
int seq64::user_settings::seqarea_x ( ) const [inline]
```


10.98.5.44 seqarea_y() [1/2]

```
int seq64::user_settings::seqarea_y ( ) const [inline]
```

10.98.5.45 seqarea_seq_x() [1/2]

```
int seq64::user_settings::seqarea_seq_x ( ) const [inline]
```

10.98.5.46 seqarea_seq_y() [1/2]

```
int seq64::user_settings::seqarea_seq_y ( ) const [inline]
```

10.98.5.47 mainwid_border() [1/2]

```
int seq64::user_settings::mainwid_border ( ) const [inline]
```

10.98.5.48 mainwid_spacing() [1/2]

```
int seq64::user_settings::mainwid_spacing ( ) const [inline]
```

10.98.5.49 mainwid_x()

```
int seq64::user_settings::mainwid_x ( ) const [inline]
```

10.98.5.50 mainwid_y()

```
int seq64::user_settings::mainwid_y ( ) const [inline]
```

10.98.5.51 mainwnd_x()

```
int seq64::user_settings::mainwnd_x ( ) const
```

10.98.5.52 mainwnd_y()

```
int seq64::user_settings::mainwnd_y ( ) const
```

10.98.5.53 mainwid_border_x()

```
int seq64::user_settings::mainwid_border_x ( ) const [inline]
```

10.98.5.54 mainwid_border_y()

```
int seq64::user_settings::mainwid_border_y ( ) const [inline]
```

10.98.5.55 control_height() [1/2]

```
int seq64::user_settings::control_height ( ) const [inline]
```

10.98.5.56 zoom() [1/2]

```
int seq64::user_settings::zoom ( ) const [inline]
```

10.98.5.57 zoom() [2/2]

```
void seq64::user_settings::zoom (
    int value )
```

The default value is 2. Note that 0 is allowed as a special case, which allows the default zoom to be adjusted when the PPQN value is different from the default.

10.98.5.58 global_seq_feature() [1/2]

```
bool seq64::user_settings::global_seq_feature ( ) const [inline]
```

10.98.5.59 global_seq_feature() [2/2]

```
void seq64::user_settings::global_seq_feature (
    bool flag ) [inline]
```

10.98.5.60 seqedit_scale() [1/2]

```
int seq64::user_settings::seqedit_scale ( ) const [inline]
```

10.98.5.61 seqedit_scale() [2/2]

```
void seq64::user_settings::seqedit_scale (
    int scale ) [inline]
```

10.98.5.62 seqedit_key() [1/2]

```
int seq64::user_settings::seqedit_key ( ) const [inline]
```

10.98.5.63 seqedit_key() [2/2]

```
void seq64::user_settings::seqedit_key (
    int key ) [inline]
```

10.98.5.64 seqedit_bgsequence() [1/2]

```
int seq64::user_settings::seqedit_bgsequence ( ) const [inline]
```

10.98.5.65 seqedit_bgsequence() [2/2]

```
void seq64::user_settings::seqedit_bgsequence (
    int seqnum ) [inline]
```

10.98.5.66 use_new_font() [1/2]

```
bool seq64::user_settings::use_new_font ( ) const [inline]
```

10.98.5.67 allow_two_perfedits() [1/2]

```
bool seq64::user_settings::allow_two_perfedits ( ) const [inline]
```

10.98.5.68 perf_h_page_increment() [1/2]

```
int seq64::user_settings::perf_h_page_increment ( ) const [inline]
```

10.98.5.69 perf_v_page_increment() [1/2]

```
int seq64::user_settings::perf_v_page_increment ( ) const [inline]
```

10.98.5.70 progress_bar_colored() [1/2]

```
int seq64::user_settings::progress_bar_colored ( ) const [inline]
```

10.98.5.71 progress_bar_thick() [1/2]

```
bool seq64::user_settings::progress_bar_thick ( ) const [inline]
```

10.98.5.72 inverse_colors() [1/2]

```
bool seq64::user_settings::inverse_colors ( ) const [inline]
```

10.98.5.73 window_redraw_rate() [1/2]

```
int seq64::user_settings::window_redraw_rate ( ) const [inline]
```

10.98.5.74 use_more_icons() [1/2]

```
bool seq64::user_settings::use_more_icons ( ) const [inline]
```

10.98.5.75 block_rows() [1/2]

```
int seq64::user_settings::block_rows ( ) const [inline]
```

10.98.5.76 block_columns() [1/2]

```
int seq64::user_settings::block_columns ( ) const [inline]
```

10.98.5.77 block_independent() [1/2]

```
int seq64::user_settings::block_independent ( ) const [inline]
```

10.98.5.78 save_user_config() [1/2]

```
bool seq64::user_settings::save_user_config ( ) const [inline]
```

10.98.5.79 save_user_config() [2/2]

```
void seq64::user_settings::save_user_config (
    bool flag ) [inline]
```

10.98.5.80 grid_brackets() [2/2]

```
void seq64::user_settings::grid_brackets (
    int thickness ) [inline], [protected]
```

10.98.5.81 window_scale() [2/2]

```
void seq64::user_settings::window_scale (
    float winscale ) [protected]
```

10.98.5.82 grid_style() [2/2]

```
void seq64::user_settings::grid_style (
    int gridstyle ) [protected]
```

10.98.5.83 mainwnd_rows() [2/2]

```
void seq64::user_settings::mainwnd_rows (
    int value ) [protected]
```

The default value is 4. Dependent values are recalculated after the assignment.

10.98.5.84 mainwnd_cols() [2/2]

```
void seq64::user_settings::mainwnd_cols (
    int value ) [protected]
```

The default value is 8. Dependent values are recalculated after the assignment.

10.98.5.85 max_sets() [2/2]

```
void seq64::user_settings::max_sets (
    int value ) [protected]
```

The default value is 32. Dependent values are recalculated after the assignment.

Parameters

<i>value</i>	Provides the desired setting. It might be modified by the call to normalize() . Not sure we really need this function.
--------------	--

10.98.5.86 text_x() [2/2]

```
void seq64::user_settings::text_x (
    int value ) [protected]
```

The default value is 6. Dependent values are recalculated after the assignment. This value is currently restricted, until we can code up a bigger font.

10.98.5.87 text_y() [2/2]

```
void seq64::user_settings::text_y (
    int value ) [protected]
```

The default value is 12. Dependent values are recalculated after the assignment. This value is currently restricted, until we can code up a bigger font.

10.98.5.88 seqchars_x() [2/2]

```
void seq64::user_settings::seqchars_x (
    int value ) [protected]
```

10.98.5.89 seqchars_y() [2/2]

```
void seq64::user_settings::seqchars_y (
    int value ) [protected]
```

10.98.5.90 seqarea_x() [2/2]

```
void seq64::user_settings::seqarea_x (
    int value ) [protected]
```

10.98.5.91 seqarea_y() [2/2]

```
void seq64::user_settings::seqarea_y (
    int value ) [protected]
```

10.98.5.92 seqarea_seq_x() [2/2]

```
void seq64::user_settings::seqarea_seq_x (
    int value ) [protected]
```

10.98.5.93 seqarea_seq_y() [2/2]

```
void seq64::user_settings::seqarea_seq_y (
    int value ) [protected]
```

10.98.5.94 mainwid_border() [2/2]

```
void seq64::user_settings::mainwid_border (
    int value ) [protected]
```

The default value is 0. Dependent values are recalculated after the assignment.

10.98.5.95 mainwid_spacing() [2/2]

```
void seq64::user_settings::mainwid_spacing (
    int value ) [protected]
```

The default value is 2. Dependent values are recalculated after the assignment.

10.98.5.96 control_height() [2/2]

```
void seq64::user_settings::control_height (
    int value ) [protected]
```

The default value is 0. Dependent values are recalculated after the assignment.

10.98.5.97 dump_summary()

```
void seq64::user_settings::dump_summary ( ) [protected]
```

Does its work only if PLATFORM_DEBUG and SEQ64_USE_DEBUG_OUTPUT are defined. Only enabled in emergencies :-D.

10.98.5.98 midi_ppqn() [1/2]

```
int seq64::user_settings::midi_ppqn ( ) const [inline]
```

10.98.5.99 midi_beats_per_bar() [1/2]

```
int seq64::user_settings::midi_beats_per_bar ( ) const [inline]
```

10.98.5.100 midi_bpm_minimum() [1/2]

```
midibpm seq64::user_settings::midi_bpm_minimum ( ) const [inline]
```


10.98.5.101 midi_beats_per_minute() [1/2]

```
midibpm seq64::user_settings::midi_beats_per_minute ( ) const [inline]
```

10.98.5.102 midi_bpm_maximum() [1/2]

```
midibpm seq64::user_settings::midi_bpm_maximum ( ) const [inline]
```

10.98.5.103 midi_beat_width() [1/2]

```
int seq64::user_settings::midi_beat_width ( ) const [inline]
```

10.98.5.104 midi_buss_override() [1/2]

```
char seq64::user_settings::midi_buss_override ( ) const [inline]
```

10.98.5.105 velocity_override() [1/2]

```
int seq64::user_settings::velocity_override ( ) const [inline]
```

10.98.5.106 bpm_precision() [1/2]

```
int seq64::user_settings::bpm_precision ( ) const [inline]
```

10.98.5.107 bpm_step_increment() [1/2]

```
midibpm seq64::user_settings::bpm_step_increment ( ) const [inline]
```

10.98.5.108 bpm_page_increment() [1/2]

```
midibpm seq64::user_settings::bpm_page_increment ( ) const [inline]
```

10.98.5.109 min_zoom()

```
int seq64::user_settings::min_zoom ( ) const [inline]
```

10.98.5.110 max_zoom()

```
int seq64::user_settings::max_zoom ( ) const [inline]
```

10.98.5.111 baseline_ppqn()

```
int seq64::user_settings::baseline_ppqn ( ) const [inline]
```

10.98.5.112 option_daemonize() [1/2]

```
bool seq64::user_settings::option_daemonize ( ) const [inline]
```

10.98.5.113 option_use_logfile() [1/2]

```
bool seq64::user_settings::option_use_logfile ( ) const [inline]
```

10.98.5.114 option_logfile() [1/2]

```
std::string seq64::user_settings::option_logfile ( ) const
```

Returns

This function returns `rc().config_directory() + m_user_option_logfile` if the latter does not contain a path marker ("`/`"). Otherwise, it returns `m_user_option_logfile`, which must be a full path specification to the desired log-file.

10.98.5.115 work_around_play_image() [1/2]

```
bool seq64::user_settings::work_around_play_image ( ) const [inline]
```

10.98.5.116 work_around_transpose_image() [1/2]

```
bool seq64::user_settings::work_around_transpose_image ( ) const [inline]
```

10.98.5.117 key_height() [1/2]

```
int seq64::user_settings::key_height ( ) const [inline]
```

10.98.5.118 use_new_font() [2/2]

```
void seq64::user_settings::use_new_font (
    bool flag ) [inline]
```

10.98.5.119 allow_two_perfedits() [2/2]

```
void seq64::user_settings::allow_two_perfedits (
    bool flag ) [inline]
```

10.98.5.120 perf_h_page_increment() [2/2]

```
void seq64::user_settings::perf_h_page_increment (
    int inc )
```

This value ranges from 1 (the original value, really too small for a "page" operation) to 6 (which is 24 measures, the same as the typical width of the perffroll)

10.98.5.121 perf_v_page_increment() [2/2]

```
void seq64::user_settings::perf_v_page_increment (
    int inc )
```

This value ranges from 1 (the original value, really too small for a "page" operation) to 18 (which is 18 tracks, slightly more than the typical height of the perffroll)

10.98.5.122 progress_bar_colored() [2/2]

```
void seq64::user_settings::progress_bar_colored (
    int palcode ) [inline]
```

10.98.5.123 progress_bar_thick() [2/2]

```
void seq64::user_settings::progress_bar_thick (  
    bool flag ) [inline]
```

10.98.5.124 inverse_colors() [2/2]

```
void seq64::user_settings::inverse_colors (  
    bool flag ) [inline]
```

10.98.5.125 window_redraw_rate() [2/2]

```
void seq64::user_settings::window_redraw_rate (  
    int ms ) [inline]
```

10.98.5.126 use_more_icons() [2/2]

```
void seq64::user_settings::use_more_icons (  
    bool flag ) [inline]
```

10.98.5.127 block_rows() [2/2]

```
void seq64::user_settings::block_rows (  
    int count )
```

10.98.5.128 block_columns() [2/2]

```
void seq64::user_settings::block_columns (  
    int count )
```

10.98.5.129 block_independent() [2/2]

```
void seq64::user_settings::block_independent (  
    bool flag ) [inline]
```

10.98.5.130 option_daemonize() [2/2]

```
void seq64::user_settings::option_daemonize (
    bool flag ) [inline]
```

10.98.5.131 option_use_logfile() [2/2]

```
void seq64::user_settings::option_use_logfile (
    bool flag ) [inline]
```

10.98.5.132 option_logfile() [2/2]

```
void seq64::user_settings::option_logfile (
    const std::string & logfile ) [inline]
```

10.98.5.133 work_around_play_image() [2/2]

```
void seq64::user_settings::work_around_play_image (
    bool flag ) [inline]
```

10.98.5.134 work_around_transpose_image() [2/2]

```
void seq64::user_settings::work_around_transpose_image (
    bool flag ) [inline]
```

10.98.5.135 key_height() [2/2]

```
void seq64::user_settings::key_height (
    int h ) [inline]
```

10.98.5.136 midi_ppqn() [2/2]

```
void seq64::user_settings::midi_ppqn (
    int value )
```

The default value is 192.

10.98.5.137 midi_buss_override() [2/2]

```
void seq64::user_settings::midi_buss_override (
    char buss )
```

The default value is -1, which means that there is no buss override. It provides a way to override the buss number for smallish MIDI files. It replaces the buss-number read from the file. This option is turned on by the `-bus` option, and is merely a convenience feature for the quick previewing of a tune. (It's called "developer laziness".)

10.98.5.138 velocity_override() [2/2]

```
void seq64::user_settings::velocity_override (
    int vel )
```

10.98.5.139 bpm_precision() [2/2]

```
void seq64::user_settings::bpm_precision (
    int precision )
```

10.98.5.140 bpm_step_increment() [2/2]

```
void seq64::user_settings::bpm_step_increment (
    midibpm increment )
```

10.98.5.141 bpm_page_increment() [2/2]

```
void seq64::user_settings::bpm_page_increment (
    midibpm increment )
```

10.98.5.142 mainwid_width()

```
int seq64::user_settings::mainwid_width ( ) const
```

Affected by the `c_mainwid_border` and `c_mainwid_spacing` values.

Returns

Returns the width, in pixels, of a mainwid grid.

10.98.5.143 mainwid_height()

```
int seq64::user_settings::mainwid_height ( ) const
```

Affected by the `c_mainwid_border` and `c_control_height` values.

Change Note ca 2017-08-13 Issue #104. Add 8 to the height calculation until we can figure out how to adjust for button height increases due to adding the main-window time-stamp field.

Returns

Returns the height, in pixels, of a mainwid grid.

10.98.5.144 mainwid_width_fudge()

```
int seq64::user_settings::mainwid_width_fudge ( ) const [inline]
```

10.98.5.145 mainwid_height_fudge()

```
int seq64::user_settings::mainwid_height_fudge ( ) const [inline]
```

10.98.5.146 midi_beats_per_bar() [2/2]

```
void seq64::user_settings::midi_beats_per_bar (
    int value ) [protected]
```

The default value is 4.

10.98.5.147 midi_bpm_minimum() [2/2]

```
void seq64::user_settings::midi_bpm_minimum (
    midibpm value ) [protected]
```

The default value is 120.

10.98.5.148 midi_beats_per_minute() [2/2]

```
void seq64::user_settings::midi_beats_per_minute (
    midibpm value ) [protected]
```

The default value is 120.

10.98.5.149 midi_bpm_maximum() [2/2]

```
void seq64::user_settings::midi_bpm_maximum (
    midibpm value ) [protected]
```

The default value is 120.

10.98.5.150 midi_beat_width() [2/2]

```
void seq64::user_settings::midi_beat_width (
    int bw ) [protected]
```

The default value is 4.

10.98.5.151 private_bus()

```
user_midi_bus & seq64::user_settings::private_bus (
    int index ) [private]
```

This invalid object has an empty alias, and all the instrument numbers are -1.

10.98.5.152 private_instrument()

```
user_instrument & seq64::user_settings::private_instrument (
    int index ) [private]
```

This invalid object has an empty(), instrument name, false for all controllers_active[] values, and empty controllers[] string values.

10.98.6 Friends And Related Function Documentation**10.98.6.1 midifile**

```
friend class midifile [friend]
```

10.98.6.2 userfile

```
friend class userfile [friend]
```

10.98.6.3 parse_o_options

```
bool parse_o_options (
    int argc,
    char * argv[] ) [friend]
```

These are flagged by "-o" or "--option". These options are then passed to the [user_settings](#) module. Multiple options can be supplied; the whole command-line is processed.

Parameters

<i>argc</i>	The number of command-line parameters, including the name of the application as parameter 0.
<i>argv</i>	The array of pointers to the command-line parameters.

Returns

Returns true if any "options" option was found, and false otherwise. If the options flags ("-o" or "--option") were found, but were not valid options, then we break out of processing and return false.

10.98.7 Field Documentation

10.98.7.1 m_midi_buses

`Busses seq64::user_settings::m_midi_buses [private]`

Since this object is a vector, its size is adjustable.

10.98.7.2 m_instruments

`Instruments seq64::user_settings::m_instruments [private]`

The size is adjustable, and grows as objects are added.

10.98.7.3 m_comments_block

`std::string seq64::user_settings::m_comments_block [private]`

Provides a way to embed comments in the "usr" file and not lose them when the "usr" file is auto-saved.

10.98.7.4 m_grid_style

`mainwid_grid_style_t seq64::user_settings::m_grid_style [private]`

These are not labelled, but are present in the "user" configuration file in the following order:

```
-# grid-style
-# grid-brackets
-# mainwnd-rows
-# mainwnd-cols
-# max-set
-# mainwid-border
-# control-height
-# zoom
-# global-seq-feature
-# use-new-font
-# allow-two-perfedits
-# perf-h-page-increment
-# perf-v-page-increment
-# progress-bar-colored (new)
-# progress-bar-thick (new)
-# window-redraw-rate-ms (new)
```

Specifies the current grid style.

10.98.7.5 m_grid_brackets

```
int seq64::user_settings::m_grid_brackets [private]
```

0 = no brackets, 1 and above is the thickness of the brackets. 1 is the normal thickness of the brackets, 2 is a two-pixel thickness, and so on.

10.98.7.6 m_mainwnd_rows

```
int seq64::user_settings::m_mainwnd_rows [private]
```

The current value is 4, and if changed, many other values depend on it. Together with m_mainwnd_cols, this value fixes the patterns grid into a 4 x 8 set of patterns known as a "screen set". We would like to be able to change this value from 4 to 8, and maybe allow the values of 5, 6, and 7 as well. But if we could just get 8 working, then well would Sequencer64 deserve the 64 in its name.

10.98.7.7 m_mainwnd_cols

```
int seq64::user_settings::m_mainwnd_cols [private]
```

The current value is 4, and probably won't change, since other values depend on it. Together with m_mainwnd_rows, this value fixes the patterns grid into a 4 x 8 set of patterns known as a "screen set".

10.98.7.8 m_max_sets

```
int seq64::user_settings::m_max_sets [private]
```

Basically, that the number of times the Patterns Panel can be filled. 32 sets can be created. Although this value is part of the "user" configuration file, it is likely that it will never change. Rather, the number of sequences per set would change. We'll see.

10.98.7.9 m_window_scale

```
float seq64::user_settings::m_window_scale [private]
```

Should be limited from 1.0 to 3.0, probably. Right now we allow 0.5 to 3.0.

10.98.7.10 m_mainwid_border

```
int seq64::user_settings::m_mainwid_border [private]
```

We'll try changing them and see what happens. Increasing these value spreads out the pattern grids a little bit and makes the Patterns panel slightly bigger. Seems like it would be useful to make these values user-configurable.

10.98.7.11 m_mainwid_spacing

```
int seq64::user_settings::m_mainwid_spacing [private]
```

10.98.7.12 m_control_height

```
int seq64::user_settings::m_control_height [private]
```

But it is used only in this header file, to define `m_mainwid_y`, but doesn't add anything to that value.

10.98.7.13 m_current_zoom

```
int seq64::user_settings::m_current_zoom [private]
```

The original default value was 32 ticks per pixel, but larger PPQN values need higher values, and we will have to adapt the default zoom to the PPQN value. Also, the zoom can never be zero, as it can appear as the divisor in scaling equations.

10.98.7.14 m_global_seq_feature_save

```
bool seq64::user_settings::m_global_seq_feature_save [private]
```

In this feature, applying one of these three changes to a sequence causes them to also be applied to sequences that are subsequently opened for editing. However, we improve on this feature by allowing the changes to be saved in the global, proprietary part of the saved MIDI file.

If false, the user can still save the key/scale/background-sequence values with each individual sequence, so they can be different.

This value will be true by default, unless changed in the "user" configuration file.

10.98.7.15 m_seqedit_scale

```
int seq64::user_settings::m_seqedit_scale [private]
```

Its default value is `c_scale_off`. Although this value is now stored in the [user_settings](#) class, it always comes from the currently loaded MIDI file, if present. If `m_global_seq_feature_save` is true, this variable is stored in the "proprietary" track at the end of the file, under the control tag `c_musicscale`, and will be applied to any sequence that is edited. If `m_global_seq_feature_save` is false, this variable is stored, if used, in the meta-data for the sequence to which it applies, and, again, is tagged with the control tag `c_musicscale`.

10.98.7.16 m_seqedit_key

```
int seq64::user_settings::m_seqedit_key [private]
```

Its default value is `SEQ64_KEY_OF_C`. Although this value is now stored in the [user_settings](#) class, it always comes from the currently loaded MIDI file, if present. If `m_global_seq_feature_save` is true, this variable is stored in the "proprietary" track at the end of the file, under the control tag `c_musickey`, and will be applied to any sequence that is edited. If `m_global_seq_feature_save` is false, this variable is stored, if used, in the meta-data for the sequence to which it applies, and, again, is tagged with the control tag `c_musickey`.

10.98.7.17 m_seqedit_bgsequence

```
int seq64::user_settings::m_seqedit_bgsequence [private]
```

Its default value is SEQ64_SEQUENCE_LIMIT. Although this value is now stored in the [user_settings](#) class, it always comes from the currently loaded MIDI file, if present. If `m_global_seq_feature_save` is true, this variable is stored, if it has a valid (but not "legal") value, in the "proprietary" track at the end of the file, under the control tag `c_backsequence`, and will be applied to any sequence that is edited. If `m_global_seq_feature_save` is false, this variable is stored, if used, in the meta-data for the sequence to which it applies, and, again, is tagged with the control tag `c_backsequence`.

10.98.7.18 m_use_new_font

```
bool seq64::user_settings::m_use_new_font [private]
```

By default, in normal mode, the new font is used. In legacy mode, the old font is used.

10.98.7.19 m_allow_two_perfedits

```
bool seq64::user_settings::m_allow_two_perfedits [private]
```

Defaults to true.

10.98.7.20 m_h_perf_page_increment

```
int seq64::user_settings::m_h_perf_page_increment [private]
```

The value used to be hardwired to 1 (in four-measure units), now it defaults to 4 (16 measures at a time). The value of 1 is already covered by the scrollbar arrows.

10.98.7.21 m_v_perf_page_increment

```
int seq64::user_settings::m_v_perf_page_increment [private]
```

The value used to be hardwired to 1 (in single-track units), now it defaults to 8. The value of 1 is already covered by the scrollbar arrows.

10.98.7.22 m_progress_bar_colored

```
int seq64::user_settings::m_progress_bar_colored [private]
```

This value is no longer hardwired in the [gui_palette_gtk2](#) module to be red. Now we want to let the color select from a slightly large palette. We change this from a boolean to an integer to allow the selection of more colors.

10.98.7.23 m_progress_bar_thick

```
bool seq64::user_settings::m_progress_bar_thick [private]
```

2 pixels. It isn't useful to support anything thicker.

10.98.7.24 m_inverse_colors

```
bool seq64::user_settings::m_inverse_colors [private]
```

Not all colors are reversed, though.

10.98.7.25 m_window_redraw_rate_ms

```
int seq64::user_settings::m_window_redraw_rate_ms [private]
```

Not all windows use this yet. The default is 40 ms (c_redraw_ms, which is 20 ms in Windows builds)), but some windows originally used 25 ms, so beware of side-effects.

10.98.7.26 m_use_more_icons

```
bool seq64::user_settings::m_use_more_icons [private]
```

If set to 1, icons will be used for more buttons. This setting affects only a few buttons so far, such as the buttons at the top of the main window.

10.98.7.27 m_mainwid_block_rows

```
int seq64::user_settings::m_mainwid_block_rows [private]
```

This section adds to the [user-interface-settings] configuration section. That section is big enough, and the new section is for newer features.

Currently these value are not saved; we want to test the viability of the concept, first. This value specifies the number of rows of main windows. The default is the legacy value, 1, to support the original paradigm of one set shown in the user interface. For now, we will restrict this value to range from 1 to 3, which will fit onto a 1920 x 1080 screen.

10.98.7.28 m_mainwid_block_cols

```
int seq64::user_settings::m_mainwid_block_cols [private]
```

The default is the legacy value, 1, to support the original paradigm of one set shown in the user interface. For now, we will restrict this value to range from 1 to 2, which will fit onto a 1920 x 1080 screen.

10.98.7.29 m_mainwid_block_independent

```
bool seq64::user_settings::m_mainwid_block_independent [private]
```

If false, then the main set spinner is the only one shown, and it makes all sets track the main set, which is always shown in the upper-right mainwid slot. If there is only a single window, this value is set to true, but it really doesn't matter what behavior is enabled for a single mainwid.

10.98.7.30 m_text_x

```
int seq64::user_settings::m_text_x [private]
```

These items are not read from the "usr", and are not currently part of any configuration section.

The `m_text_x` and `m_text_y` constants help define the "seqarea" size. It looks like these two values are the character width (x) and height (y) in pixels. Thus, these values would be dependent on the font chosen. But that, currently, is hard-wired. See the `m_font_6_12[]` array for the default font specification.

However, please not that font files are not used. Instead, the fonts are provided by two pixmaps in the `src/pixmap` directory: `font_b.xpm` (black lettering on a white background) and `font_w.xpm` (white lettering on a black background).

We have added black-on-yellow and yellow-on-black versions of the fonts, to support the highlighting of pattern boxes if they are empty of actual MIDI events.

We have also added a set of four new font files that are roughly the same size, and are treated as the same size, but look smooth and less like a DOS-era font.

The font module does not use these values directly, but does define some similar variables that differ slightly between the two styles of font. There are a lot of tricks and hard-wired places to fix before further work can be done with fonts in Sequencer64.

10.98.7.31 m_text_y

```
int seq64::user_settings::m_text_y [private]
```

10.98.7.32 m_seqchars_x

```
int seq64::user_settings::m_seqchars_x [private]
```

The `m_seqchars_x` and `m_seqchars_y` constants help define the "seqarea" size. These look like the number of characters per line and the number of lines of characters, in a pattern/sequence box.

10.98.7.33 m_seqchars_y

```
int seq64::user_settings::m_seqchars_y [private]
```

10.98.7.34 m_midi_ppqn

```
int seq64::user_settings::m_midi_ppqn [private]
```

This variable replaces the global `ppqn`. The default value of this setting is 192 parts-per-quarter-note (PPQN). There is still a lot of work to get a different PPQN to work properly in speed of playback, scaling of the user interface, and other issues. Note that this value can be changed by the still-experimental `-ppqn` option. There is one remaining trace of the global, though: `DEFAULT_PPQN`.

10.98.7.35 m_midi_beats_per_measure

```
int seq64::user_settings::m_midi_beats_per_measure [private]
```

This variable will replace the global beats per measure. The default value of this variable is SEQ64_DEFAULT_BEATS_PER_MEASURE (4). For external access, we will call this value "beats per bar", abbreviate it "BPB", and use "bpb" in any accessor function names. Now, although it applies to the whole session, we should be able to continue seq24's tradition of allowing each sequence to have its own time signature. Also, there are a number of places where the number 4 appears and looks like it might be a hardwired BPB value, either for MIDI purposes or for drawing the piano-roll grids. So we might need a couple different versions of this variable.

10.98.7.36 m_midi_bpm_minimum

```
midibpm seq64::user_settings::m_midi_bpm_minimum [private]
```

Defaults to 0.

10.98.7.37 m_midi_beats_per_minute

```
midibpm seq64::user_settings::m_midi_beats_per_minute [private]
```

This variable will replace the global beats per minute. The default value of this variable is DEFAULT_BPM (120). This variable should apply to the whole session; there's probably no way to support a different tempo for each sequence. But we shall see. For external access, we will call this value "beats per minute", abbreviate it "BPM", and use "bpm" in any accessor function names.

10.98.7.38 m_midi_bpm_maximum

```
midibpm seq64::user_settings::m_midi_bpm_maximum [private]
```

Defaults to 127.

10.98.7.39 m_midi_beat_width

```
int seq64::user_settings::m_midi_beat_width [private]
```

This variable will replace the global beat_width. The default value of this variable is DEFAULT_BEAT_WIDTH (4). Now, although it applies to the whole session, we should be able to continue seq24's tradition of allowing each sequence to have its own time signature. Also, there are a number of places where the number 4 appears and looks like it might be a hardwired BW value, either for MIDI purposes or for drawing the user-interface. So we might need a couple different versions of this variable. For external access, we will call this value "beat width", abbreviate it "BW", and use "bw" in any accessor function names.

10.98.7.40 m_midi_buss_override

```
char seq64::user_settings::m_midi_buss_override [private]
```

This variable replaces the global buss-override variable, and is set via the command-line option `-bus`.

10.98.7.41 m_velocity_override

```
int seq64::user_settings::m_velocity_override [private]
```

The value SEQ64_PRESERVE_VELOCITY (-1) preserves the velocity of incoming notes, so that nuances in live playing can be preserved. The popup-menu for the "Vol" button in the seqedit window shows this value as the "Free" menu entry. The rest of the values in the menu show a few select velocities, but any velocity from 0 to 127 can be entered here. Of course, 0 is not recommended.

10.98.7.42 m_bpm_precision

```
int seq64::user_settings::m_bpm_precision [private]
```

The original value was effectively 0, but we need to be able to support the following values:

- 0. The legacy default.
- 1. One decimal place in the BPM spinner.
- 2. Two decimal places in the BPM spinner.

10.98.7.43 m_bpm_step_increment

```
midibpm seq64::user_settings::m_bpm_step_increment [private]
```

The default value is the legacy value, 1, for a BPM precision value of 0. The default value is 0.1 if one decimal place of precision is in force, and 0.01 if two decimal places of precision is in force. This is the increment that is performed in the BPM field of the main window when the arrow-buttons are clicked, the up/down arrow keys are pressed, or the BPM MIDI controls are processed.

10.98.7.44 m_bpm_page_increment

```
midibpm seq64::user_settings::m_bpm_page_increment [private]
```

Currently, the only way to use this increment is to click in the BPM field of the main window and then use the Page-Up and Page-Down keys.

10.98.7.45 m_total_seqs

```
int seq64::user_settings::m_total_seqs [private]
```

It is basically the same value as m_max_sequence by default. It is a derived value, and not stored in the "usr" file. We might make it equal to the maximum number of sequences the currently-loaded MIDI file.

```
m_total_seqs = m_seqs_in_set * m_max_sets;
```


10.98.7.46 m_seqs_in_set

```
int seq64::user_settings::m_seqs_in_set [private]
```

This value is $4 \times 8 = 32$ by default.

Warning

Currently implicit/explicit in a number of the "rc" file and [rc_settings](#). Would probably want the left 32 or the first 32 items in the main window only to be subject to keystroke control. This value is calculated by the [normalize\(\)](#) function, and is *not* part of the "user" configuration file.

10.98.7.47 m_gmute_tracks

```
int seq64::user_settings::m_gmute_tracks [private]
```

This value is *not* part of the "user" configuration file; it is calculated by the [normalize\(\)](#) function.

10.98.7.48 m_max_sequence

```
int seq64::user_settings::m_max_sequence [private]
```

It is a derived value, and not stored in the "user" file.

```
m_max_sequence = m_seqs_in_set * m_max_sets;
```

10.98.7.49 m_seqarea_x

```
int seq64::user_settings::m_seqarea_x [private]
```

Compare these two constants to `m_seqarea_seq_x(y)`, which was in `mainwid.h`, but is now in this file.

10.98.7.50 m_seqarea_y

```
int seq64::user_settings::m_seqarea_y [private]
```

10.98.7.51 m_seqarea_seq_x

```
int seq64::user_settings::m_seqarea_seq_x [private]
```

Doesn't look at all like it is based on the size of characters. These values are used only in the `mainwid` module.

10.98.7.52 m_seqarea_seq_y

```
int seq64::user_settings::m_seqarea_seq_y [private]
```

10.98.7.53 m_mainwid_x

```
int seq64::user_settings::m_mainwid_x [private]
```

Affected by the `m_mainwid_border` and `m_mainwid_spacing` values, as well a `m_window_scale`. Replaces `c_↔mainwid_x`.

10.98.7.54 m_mainwid_y

```
int seq64::user_settings::m_mainwid_y [private]
```

Affected by the `m_mainwid_border` and `m_control_height` values, as well a `m_window_scale`. Replaces `c_↔mainwid_y`.

10.98.7.55 m_mainwnd_x

```
int seq64::user_settings::m_mainwnd_x [private]
```

If `m_window_scale` is significantly different from 1.0, then the accessor will scale this value.

10.98.7.56 m_mainwnd_y

```
int seq64::user_settings::m_mainwnd_y [private]
```

Like `m_mainwnd_x`, this value is scaled by the accessor, however, only if less than 1.0; otherwise, the top buttons expand way too much.

10.98.7.57 m_save_user_config

```
bool seq64::user_settings::m_save_user_config [private]
```

Normally, this state is not saved. It is not saved because there is currently no user-interface for editing it, and because it can pick up some command-line options, and it is not right to have them written to the "user" configuration file.

(The "rc" configuration file is a different case, having historically always been saved, and having a number of command-line options, such as JACK settings that should generally be permanent on a given system.)

Anyway, this flag can be set by the `-user-save` option. This setting is never saved. But note that, if no "user" configuration file is found, it is then saved anyway.

10.98.7.58 mc_min_zoom

```
const int seq64::user_settings::mc_min_zoom [private]
```

It's value is 1.

10.98.7.59 mc_max_zoom

```
const int seq64::user_settings::mc_max_zoom [private]
```

It's value was 32, but is now 512, to allow for better presentation of high PPQN valued sequences.

10.98.7.60 mc_baseline_ppqn

```
const int seq64::user_settings::mc_baseline_ppqn [private]
```

This value is necessary in order to keep user-interface elements stable when different PPQNs are used. It is set to DEFAULT_PPQN.

10.98.7.61 m_user_option_daemonize

```
bool seq64::user_settings::m_user_option_daemonize [private]
```

All options that begin with "option_" are options specific to a particular version of Sequencer64. We don't anticipate having a lot of such options, so there's no need for a separate class to handle them. These options are flagged on the command-line by the strings "-o" or "--option".

10.98.7.62 m_user_use_logfile

```
bool seq64::user_settings::m_user_use_logfile [private]
```

10.98.7.63 m_user_option_logfile

```
std::string seq64::user_settings::m_user_option_logfile [private]
```

In other words, stdout and stderr will go to a log file instead. Unless a full path is provided, this filename will be a base filename, with the path given by `rc().config_directory()` prepended to it. That path is normally `~/.config/sequencer64`, but can be modified on the command line via the `-H` (`--home`) option.

This file can also be specified by the `"-o log=filename"` option.

10.98.7.64 m_work_around_play_image

```
bool seq64::user_settings::m_work_around_play_image [private]
```

We can't duplicate on the developer's machines, so while we try to figure how to avoid the issue, this flag is provided to simply leave the play-button image alone.

10.98.7.65 m_work_around_transpose_image

```
bool seq64::user_settings::m_work_around_transpose_image [private]
```

10.98.7.66 m_user_ui_key_height

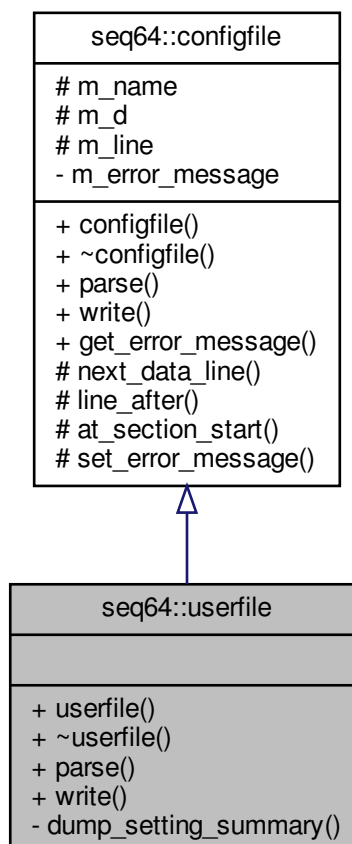
```
int seq64::user_settings::m_user_ui_key_height [private]
```

Defaults to 12 pixels (8 is actually a bit nicer IMHO). Will eventually affect the Gtkmm-2.4 user-interface as well.

10.99 seq64::userfile Class Reference

Supports the user's `~/ .config/sequencer64/sequencer64.usr` and `~/ .seq24usr` configuration file.

Inheritance diagram for seq64::userfile:



Public Member Functions

- [userfile](#) (const std::string &a_name)
Principal constructor.
- [~userfile](#) ()
A rote destructor needed for a derived class.
- bool [parse](#) ([perform](#) &a_perf)
Parses a "usr" file, filling in the given perform object.
- bool [write](#) (const [perform](#) &a_perf)
This function just returns false, as there is no "perform" information in the user-file yet.

Private Member Functions

- void [dump_setting_summary](#) ()
Provides a debug dump of basic information to help debug a surprisingly intractable problem with all busses having the name and values of the last buss in the configuration.

Additional Inherited Members

10.99.1 Constructor & Destructor Documentation

10.99.1.1 userfile()

```
seq64::userfile::userfile (
    const std::string & name )
```

Parameters

<i>name</i>	Provides the full file path specification to the configuration file.
-------------	--

10.99.1.2 ~userfile()

```
seq64::userfile::~~userfile ( )
```

10.99.2 Member Function Documentation

10.99.2.1 parse()

```
bool seq64::userfile::parse (
    perform & a_perf ) [virtual]
```

This function opens the file as a text file (line-oriented).

Parameters

<i>a_perf</i>	The performance object, currently unused.
---------------	---

Returns

Returns true if the parsing succeeded.

Implements [seq64::configfile](#).

10.99.2.2 write()

```
bool seq64::userfile::write (
    const perform & a_perf ) [virtual]
```

Parameters

<i>a_perf</i>	The performance object, currently unused.
---------------	---

Returns

Returns true if the writing succeeded.

Implements [seq64::configfile](#).

10.99.2.3 dump_setting_summary()

```
void seq64::userfile::dump_setting_summary ( ) [private]
```

Does work only if PLATFORM_DEBUG is defined; see the [user_settings](#) class.

Index

- ~FruityPerfInput
 - seq64::FruityPerfInput, [264](#)
- ~Seq24PerfInput
 - seq64::Seq24PerfInput, [901](#)
- ~automutex
 - seq64::automutex, [126](#)
- ~busarray
 - seq64::busarray, [128](#)
- ~businfo
 - seq64::businfo, [138](#)
- ~configfile
 - seq64::configfile, [151](#)
- ~editable_event
 - seq64::editable_event, [160](#)
- ~editable_events
 - seq64::editable_events, [174](#)
- ~event
 - seq64::event, [185](#)
- ~event_list
 - seq64::event_list, [211](#)
- ~eventedit
 - seq64::eventedit, [226](#)
- ~eventslots
 - seq64::eventslots, [242](#)
- ~gui_assistant
 - seq64::gui_assistant, [277](#)
- ~gui_assistant_gtk2
 - seq64::gui_assistant_gtk2, [280](#)
- ~gui_drawingarea_gtk2
 - seq64::gui_drawingarea_gtk2, [285](#)
- ~gui_palette_gtk2
 - seq64::gui_palette_gtk2, [305](#)
- ~gui_window_gtk2
 - seq64::gui_window_gtk2, [318](#)
- ~jack_assistant
 - seq64::jack_assistant, [325](#)
- ~keys_perform
 - seq64::keys_perform, [357](#)
- ~keys_perform_gtk2
 - seq64::keys_perform_gtk2, [384](#)
- ~lfownd
 - seq64::lfownd, [402](#)
- ~maintime
 - seq64::maintime, [407](#)
- ~mainwid
 - seq64::mainwid, [414](#)
- ~mainwnd
 - seq64::mainwnd, [434](#)
- ~mastermidibase
 - seq64::mastermidibase, [463](#)
- ~mastermidibus
 - seq64::mastermidibus, [481](#)
- ~midi_alsa
 - seq64::midi_alsa, [487](#)
- ~midi_alsa_info
 - seq64::midi_alsa_info, [493](#)
- ~midi_api
 - seq64::midi_api, [499](#)
- ~midi_container
 - seq64::midi_container, [509](#)
- ~midi_in_jack
 - seq64::midi_in_jack, [525](#)
- ~midi_info
 - seq64::midi_info, [529](#)
- ~midi_jack
 - seq64::midi_jack, [539](#)
- ~midi_jack_data
 - seq64::midi_jack_data, [547](#)
- ~midi_jack_info
 - seq64::midi_jack_info, [550](#)
- ~midi_list
 - seq64::midi_list, [557](#)
- ~midi_out_jack
 - seq64::midi_out_jack, [568](#)
- ~midi_queue
 - seq64::midi_queue, [572](#)
- ~midi_splitter
 - seq64::midi_splitter, [576](#)
- ~midi_vector
 - seq64::midi_vector, [586](#)
- ~midibase
 - seq64::midibase, [593](#)
- ~midibus
 - seq64::midibus, [611](#)
- ~midifile
 - seq64::midifile, [618](#)
- ~optionsfile
 - seq64::optionsfile, [651](#)
- ~perfedit
 - seq64::perfedit, [660](#)
- ~perfnames
 - seq64::perfnames, [676](#)
- ~perform
 - seq64::perform, [700](#)
- ~perfroll
 - seq64::perfroll, [806](#)
- ~perftime
 - seq64::perftime, [826](#)

- ~rterror
 - seq64::rterror, 869
- ~rtmidi
 - seq64::rtmidi, 872
- ~rtmidi_in
 - seq64::rtmidi_in, 880
- ~rtmidi_info
 - seq64::rtmidi_info, 889
- ~rtmidi_out
 - seq64::rtmidi_out, 898
- ~seqdata
 - seq64::seqdata, 911
- ~seqedit
 - seq64::seqedit, 929
- ~seqevent
 - seq64::seqevent, 962
- ~seqkeys
 - seq64::seqkeys, 977
- ~seqmenu
 - seq64::seqmenu, 990
- ~seqroll
 - seq64::seqroll, 1005
- ~seqtime
 - seq64::seqtime, 1035
- ~sequence
 - seq64::sequence, 1053
- ~triggers
 - seq64::triggers, 1132
- ~userfile
 - seq64::userfile, 1205
- about_dialog
 - seq64::mainwnd, 445
- action
 - seq64::midi_control, 517
- action_t
 - seq64::perform, 699
- activate
 - seq64::businfo, 140
 - seq64::jack_assistant, 330
 - seq64::mastermidibase, 474
 - seq64::mastermidibus, 482
 - seq64::perform, 769
- activate_adding
 - seq64::FruityPerfInput, 265
 - seq64::Seq24PerfInput, 901
 - seq64::perfroll, 813
- active
 - seq64::businfo, 139
 - seq64::midi_control, 518
- add
 - seq64::busarray, 128, 129
 - seq64::editable_events, 177
 - seq64::event_list, 212
 - seq64::midi_jack_info, 553
 - seq64::midi_port_info, 569
 - seq64::midi_queue, 573
 - seq64::triggers, 1136
- add_bus
 - seq64::midi_info, 534
 - seq64::rtmidi_info, 891
 - seq64::user_settings, 1171
- add_chord
 - seq64::sequence, 1076
- add_clock
 - seq64::perform, 780
- add_event
 - seq64::midi_container, 513
 - seq64::sequence, 1076, 1077
- add_ex_event
 - seq64::midi_container, 513
- add_extended_keys_page
 - seq64::options, 647
- add_input
 - seq64::perform, 780
 - seq64::rtmidi_info, 890
- add_instrument
 - seq64::user_settings, 1171
- add_jack_sync_page
 - seq64::options, 648
- add_keyboard_page
 - seq64::options, 647
- add_long
 - seq64::midi_container, 512
- add_midi_clock_page
 - seq64::options, 647
- add_midi_input_page
 - seq64::options, 647
- add_mouse_page
 - seq64::options, 647
- add_note
 - seq64::seqroll, 1006
 - seq64::sequence, 1075
- add_or_delete_trigger
 - seq64::perform, 750
- add_output
 - seq64::rtmidi_info, 890
- add_recent_file
 - seq64::rc_settings, 846
- add_sequence
 - seq64::perform, 715
- add_short
 - seq64::midi_container, 512
- add_trigger
 - seq64::midifile, 626
 - seq64::perform, 749
 - seq64::sequence, 1078
- add_variable
 - seq64::midi_container, 512
- adding
 - seq64::seqroll, 1022
- adj_callback_bpm
 - seq64::mainwnd, 436
- adj_callback_ss
 - seq64::mainwnd, 436
- adj_callback_wid
 - seq64::mainwnd, 435

- adjust_note_off
 - seq64::event, [201](#)
- adjust_offset
 - seq64::sequence, [1112](#)
 - seq64::triggers, [1144](#)
- adjust_offsets_to_length
 - seq64::triggers, [1136](#)
- adjust_timestamp
 - seq64::sequence, [1094](#)
- adjust_trigger_offsets_to_length
 - seq64::sequence, [1112](#)
- adjustment_dummy
 - seq64, [93](#)
- alias
 - seq64::user_midi_bus_t, [1158](#)
- align_selection
 - seq64::seqroll, [1019](#)
- all_notes_off
 - seq64::perform, [770](#)
- allocate
 - seq64::midi_queue, [574](#)
- allow_click_edit
 - seq64::rc_settings, [843](#), [850](#)
- allow_mod4_mode
 - seq64::rc_settings, [842](#), [849](#)
- allow_snap_split
 - seq64::rc_settings, [842](#), [850](#)
- allow_two_perfeditis
 - seq64::user_settings, [1180](#), [1187](#)
- analyze
 - seq64::editable_event, [166](#)
- announce_bus_exists
 - seq64::mastermidibase, [464](#)
- any_group_unmutes
 - seq64::perform, [765](#)
- any_selected_notes
 - seq64::event_list, [218](#)
 - seq64::sequence, [1054](#)
- api_clock
 - seq64::mastermidibase, [475](#)
 - seq64::midi_alsa, [489](#)
 - seq64::midi_api, [502](#)
 - seq64::midi_jack, [544](#)
 - seq64::midibase, [606](#)
 - seq64::midibus, [613](#)
 - seq64::rtmidi, [873](#)
- api_connect
 - seq64::midi_api, [500](#)
 - seq64::midi_info, [532](#)
 - seq64::midi_jack, [541](#)
 - seq64::midi_jack_info, [551](#)
 - seq64::midibus, [612](#)
 - seq64::rtmidi, [872](#)
 - seq64::rtmidi_info, [895](#)
- api_continue_from
 - seq64::mastermidibase, [474](#)
 - seq64::midi_alsa, [489](#)
 - seq64::midi_api, [502](#)
- seq64::midi_jack, [543](#)
- seq64::midibase, [606](#)
- seq64::midibus, [613](#)
- seq64::rtmidi, [872](#)
- api_data
 - seq64::rtmidi_in_data, [885](#)
- api_deinit_in
 - seq64::midi_alsa, [488](#)
 - seq64::midi_api, [501](#)
 - seq64::midi_jack, [542](#)
 - seq64::midibase, [605](#)
 - seq64::midibus, [612](#)
 - seq64::rtmidi, [874](#)
- api_flush
 - seq64::mastermidibase, [475](#)
 - seq64::mastermidibus, [483](#)
 - seq64::midi_alsa, [488](#)
 - seq64::midi_alsa_info, [495](#)
 - seq64::midi_api, [502](#)
 - seq64::midi_info, [532](#)
 - seq64::midi_jack, [543](#)
 - seq64::midi_jack_info, [552](#)
 - seq64::midibase, [605](#)
 - seq64::rtmidi, [875](#)
 - seq64::rtmidi_info, [894](#)
- api_get_bus_name
 - seq64::midi_api, [503](#)
- api_get_midi_event
 - seq64::mastermidibase, [476](#)
 - seq64::mastermidibus, [482](#)
 - seq64::midi_alsa, [488](#)
 - seq64::midi_alsa_info, [494](#)
 - seq64::midi_api, [501](#)
 - seq64::midi_in_jack, [525](#)
 - seq64::midi_info, [531](#)
 - seq64::midi_jack, [542](#)
 - seq64::midi_jack_info, [551](#)
 - seq64::midibase, [604](#)
 - seq64::midibus, [613](#)
 - seq64::rtmidi, [874](#)
 - seq64::rtmidi_info, [894](#)
- api_get_port_name
 - seq64::midi_api, [503](#)
 - seq64::midi_jack, [544](#)
- api_init
 - seq64::mastermidibase, [474](#)
 - seq64::mastermidibus, [482](#)
- api_init_clock
 - seq64::mastermidibase, [474](#)
- api_init_in
 - seq64::midi_alsa, [487](#)
 - seq64::midi_api, [500](#)
 - seq64::midi_jack, [542](#)
 - seq64::midibase, [606](#)
 - seq64::midibus, [612](#)
 - seq64::rtmidi, [874](#)
- api_init_in_sub
 - seq64::midi_alsa, [488](#)

- seq64::midi_api, 501
- seq64::midi_jack, 542
- seq64::midibase, 605
- seq64::midibus, 612
- seq64::rtmidi, 874
- api_init_out
 - seq64::midi_alsa, 487
 - seq64::midi_api, 500
 - seq64::midi_jack, 542
 - seq64::midibase, 606
 - seq64::midibus, 612
 - seq64::rtmidi, 873
- api_init_out_sub
 - seq64::midi_alsa, 487
 - seq64::midi_api, 500
 - seq64::midi_jack, 542
 - seq64::midibase, 605
 - seq64::midibus, 612
 - seq64::rtmidi, 874
- api_play
 - seq64::midi_alsa, 488
 - seq64::midi_api, 501
 - seq64::midi_jack, 543
 - seq64::midibase, 605
 - seq64::midibus, 614
 - seq64::rtmidi, 872
- api_poll_for_midi
 - seq64::mastermidibase, 476
 - seq64::mastermidibus, 482
 - seq64::midi_alsa_info, 494
 - seq64::midi_api, 500
 - seq64::midi_in_jack, 525
 - seq64::midi_info, 532
 - seq64::midi_jack_info, 551
 - seq64::midibase, 604
 - seq64::midibus, 613
 - seq64::rtmidi, 874
 - seq64::rtmidi_info, 894
- api_port_start
 - seq64::mastermidibase, 475
 - seq64::mastermidibus, 483
 - seq64::midi_alsa_info, 494
 - seq64::midi_info, 531
 - seq64::midi_jack_info, 552
 - seq64::rtmidi_info, 893
- api_set_beats_per_minute
 - seq64::mastermidibase, 475
 - seq64::mastermidibus, 483
 - seq64::midi_alsa, 489
 - seq64::midi_alsa_info, 494
 - seq64::midi_api, 503
 - seq64::midi_info, 531
 - seq64::midi_jack, 544
 - seq64::midi_jack_info, 551
 - seq64::rtmidi, 873
 - seq64::rtmidi_info, 893
- api_set_ppqn
 - seq64::mastermidibase, 475
- seq64::mastermidibus, 482
- seq64::midi_alsa, 489
- seq64::midi_alsa_info, 494
- seq64::midi_api, 503
- seq64::midi_info, 531
- seq64::midi_jack, 544
- seq64::midi_jack_info, 551
- seq64::rtmidi, 873
- seq64::rtmidi_info, 893
- api_start
 - seq64::mastermidibase, 474
 - seq64::midi_alsa, 489
 - seq64::midi_api, 502
 - seq64::midi_jack, 543
 - seq64::midibase, 606
 - seq64::midibus, 613
 - seq64::rtmidi, 873
- api_stop
 - seq64::mastermidibase, 475
 - seq64::midi_alsa, 489
 - seq64::midi_api, 502
 - seq64::midi_jack, 544
 - seq64::midibase, 606
 - seq64::midibus, 613
 - seq64::rtmidi, 873
- api_sysex
 - seq64::midi_alsa, 488
 - seq64::midi_api, 501
 - seq64::midi_jack, 543
 - seq64::midibase, 605
 - seq64::rtmidi, 875
- app_client_name
 - seq64::rc_settings, 848
- app_name
 - seq64::midi_info, 531
 - seq64::rtmidi_info, 893
- append
 - seq64::event_list, 213
- append_comment_line
 - seq64::rc_settings, 842
 - seq64::user_settings, 1174
- append_event
 - seq64::sequence, 1077
- append_meta_data
 - seq64::event, 195, 196
- append_recent_file
 - seq64::rc_settings, 847
- append_sysex
 - seq64::event, 195
- application_name
 - seq64::rc_settings, 848
- apply_length
 - seq64::seqedit, 934
 - seq64::sequence, 1064, 1065
- apply_song_transpose
 - seq64::mainwnd, 442
 - seq64::perform, 720
 - seq64::sequence, 1061

- are_any_armed
 - seq64::perform, [754](#)
- armed_saved
 - seq64::perform, [725](#)
- array
 - seq64::midi_message, [562](#)
- at_bpm_dn
 - seq64::keys_perform, [372](#)
- at_bpm_up
 - seq64::keys_perform, [372](#)
- at_end
 - seq64::midifile, [621](#)
- at_event_edit
 - seq64::keys_perform, [376](#)
- at_fast_forward
 - seq64::keys_perform, [375](#)
- at_follow_transport
 - seq64::keys_perform, [374](#)
- at_group_learn
 - seq64::keys_perform, [374](#)
- at_group_off
 - seq64::keys_perform, [374](#)
- at_group_on
 - seq64::keys_perform, [373](#)
- at_keep_queue
 - seq64::keys_perform, [373](#)
- at_menu_mode
 - seq64::keys_perform, [374](#)
- at_oneshot_queue
 - seq64::keys_perform, [375](#)
- at_pattern_edit
 - seq64::keys_perform, [375](#)
- at_pattern_shift
 - seq64::keys_perform, [376](#)
- at_pause
 - seq64::keys_perform, [374](#)
- at_pointer_position
 - seq64::keys_perform, [375](#)
- at_queue
 - seq64::keys_perform, [373](#)
- at_replace
 - seq64::keys_perform, [372](#)
- at_rewind
 - seq64::keys_perform, [375](#)
- at_screenset_dn
 - seq64::keys_perform, [373](#)
- at_screenset_up
 - seq64::keys_perform, [373](#)
- at_section_start
 - seq64::configfile, [152](#)
- at_set_playing_screenset
 - seq64::keys_perform, [373](#)
- at_show_ui_sequence_key
 - seq64::keys_perform, [376](#)
- at_show_ui_sequence_number
 - seq64::keys_perform, [376](#)
- at_snapshot_1
 - seq64::keys_perform, [373](#)
- at_snapshot_2
 - seq64::keys_perform, [373](#)
- at_song_mode
 - seq64::keys_perform, [374](#)
- at_song_record
 - seq64::keys_perform, [375](#)
- at_start
 - seq64::keys_perform, [374](#)
- at_stop
 - seq64::keys_perform, [376](#)
- at_tap_bpm
 - seq64::keys_perform, [375](#)
- at_toggle_jack
 - seq64::keys_perform, [374](#)
- at_toggle_mutes
 - seq64::keys_perform, [375](#)
- at_trigger_transition
 - seq64::trigger, [1126](#)
- auto_option_save
 - seq64::rc_settings, [842](#), [849](#)
- automutex
 - seq64::automutex, [125](#), [126](#)
- background_sequence
 - seq64::sequence, [1108](#)
- baseline_ppqn
 - seq64::user_settings, [1186](#)
- beat_log2
 - seq64, [61](#)
- beat_pow2
 - seq64, [60](#)
- beat_width
 - seq64::midi_timing, [582](#)
- beats
 - seq64::midi_measures, [560](#)
- beats_per_measure
 - seq64::midi_timing, [582](#)
- beats_per_minute
 - seq64::midi_timing, [581](#)
- begin
 - seq64::editable_events, [175](#), [176](#)
 - seq64::event_list, [212](#)
- bg_color
 - seq64::gui_palette_gtk2, [310](#)
- black
 - seq64::gui_palette_gtk2, [307](#)
- black_key_paint
 - seq64::gui_palette_gtk2, [310](#)
- black_paint
 - seq64::gui_palette_gtk2, [309](#)
- block_columns
 - seq64::user_settings, [1181](#), [1188](#)
- block_independent
 - seq64::user_settings, [1181](#), [1188](#)
- block_rows
 - seq64::user_settings, [1181](#), [1188](#)
- blue
 - seq64::gui_palette_gtk2, [309](#)
- bpm

- seq64::midi_info, 531
- seq64::midibase, 595
- seq64::perform, 701
- seq64::rtmidi_info, 893
- bpm_dn
 - seq64::keys_perform, 358
- bpm_from_bytes
 - seq64, 64
- bpm_from_tempo_us
 - seq64, 64
- bpm_page_increment
 - seq64::user_settings, 1185, 1190
- bpm_precision
 - seq64::user_settings, 1185, 1190
- bpm_step_increment
 - seq64::user_settings, 1185, 1190
- bpm_up
 - seq64::keys_perform, 358
- build_details
 - seq64, 74
- build_info_dialog
 - seq64::mainwnd, 445
- bus
 - seq64::busarray, 130
 - seq64::businfo, 139, 140
 - seq64::user_settings, 1171
- bus_container
 - seq64::midi_info, 535
- bus_count
 - seq64::user_settings, 1171
- bus_instrument
 - seq64::user_settings, 1172
- bus_name
 - seq64::midibase, 594, 604
 - seq64::user_settings, 1172
- busarray
 - seq64::busarray, 128
 - seq64::businfo, 142
- businfo
 - seq64::businfo, 137, 138
- BussConstIterator
 - seq64::user_settings, 1169
- BussIterator
 - seq64::user_settings, 1169
- bussbyte
 - seq64, 44
- Busses
 - seq64::user_settings, 1169
- button
 - seq64::click, 146
 - seq64::options, 643
- button_press
 - seq64::seqroll, 1019
- button_press_initial
 - seq64::seqroll, 1019
- button_release
 - seq64::seqroll, 1020
- c_backsequence
 - seq64, 111
- c_bpmtag
 - seq64, 109
- c_chord_number
 - seq64, 119
- c_chord_size
 - seq64, 120
- c_chord_table
 - seq64, 120
- c_chord_table_text
 - seq64, 120
- c_chord_text
 - seq64, 119
- c_controller_names
 - seq64, 99
- c_gap_A
 - seq64, 110
- c_gap_B
 - seq64, 110
- c_gap_C
 - seq64, 110
- c_gap_D
 - seq64, 110
- c_gap_E
 - seq64, 110
- c_gap_F
 - seq64, 110
- c_interval_text
 - seq64, 119
- c_key_text
 - seq64, 119
- c_max_busses
 - seq64, 120
- c_max_instruments
 - seq64, 120
- c_midi_control_bpm_dn
 - seq64, 113
- c_midi_control_bpm_page_dn
 - seq64, 115
- c_midi_control_bpm_page_up
 - seq64, 115
- c_midi_control_bpm_up
 - seq64, 113
- c_midi_control_mod_glearn
 - seq64, 114
- c_midi_control_mod_gmute
 - seq64, 114
- c_midi_control_mod_queue
 - seq64, 114
- c_midi_control_mod_replace
 - seq64, 113
- c_midi_control_mod_snapshot
 - seq64, 113
- c_midi_control_play_ss
 - seq64, 114
- c_midi_control_playback
 - seq64, 114
- c_midi_control_quan_record

- seq64, [115](#)
- c_midi_control_record
 - seq64, [115](#)
- c_midi_control_reset_seq
 - seq64, [115](#)
- c_midi_control_solo
 - seq64, [114](#)
- c_midi_control_song_record
 - seq64, [114](#)
- c_midi_control_ss_dn
 - seq64, [113](#)
- c_midi_control_ss_set
 - seq64, [115](#)
- c_midi_control_ss_up
 - seq64, [113](#)
- c_midi_control_thru
 - seq64, [115](#)
- c_midi_controls
 - seq64, [114](#)
- c_midi_controls_extended
 - seq64, [115](#)
- c_midi_track_ctrl
 - seq64, [112](#)
- c_midibus
 - seq64, [107](#)
- c_midibus_input_size
 - seq64, [107](#)
- c_midibus_output_size
 - seq64, [107](#)
- c_midibus_sysex_chunk
 - seq64, [107](#)
- c_midich
 - seq64, [109](#)
- c_midiclocks
 - seq64, [109](#)
- c_midictrl
 - seq64, [110](#)
- c_music_scales
 - seq64, [48](#)
- c_musickey
 - seq64, [111](#)
- c_musicscale
 - seq64, [111](#)
- c_mutegroups
 - seq64, [110](#)
- c_notes
 - seq64, [109](#)
- c_perf_bp_mes
 - seq64, [111](#)
- c_perf_bw
 - seq64, [111](#)
- c_reserved_1
 - seq64, [112](#)
- c_reserved_2
 - seq64, [112](#)
- c_scales_policy
 - seq64, [117](#)
- c_scales_text
 - seq64, [119](#)
- c_scales_transpose_dn
 - seq64, [118](#)
- c_scales_transpose_up
 - seq64, [117](#)
- c_seq_color
 - seq64, [112](#)
- c_seq_edit_mode
 - seq64, [112](#)
- c_status_oneshot
 - seq64, [117](#)
- c_status_queue
 - seq64, [116](#)
- c_status_replace
 - seq64, [116](#)
- c_status_snapshot
 - seq64, [116](#)
- c_tempo_map
 - seq64, [111](#)
- c_tempo_track
 - seq64, [112](#)
- c_timesig
 - seq64, [109](#)
- c_transpose
 - seq64, [111](#)
- c_triggers
 - seq64, [109](#)
- c_triggers_new
 - seq64, [109](#)
- calculate_base_sizes
 - seq64::mainwid, [418](#)
- calculate_measures
 - seq64::eventslots, [251](#)
 - seq64::sequence, [1058](#)
- calculated_height
 - seq64::rect, [866](#)
- calculated_width
 - seq64::rect, [866](#)
- cancel_callback
 - seq64::midi_api, [504](#)
 - seq64::rtmidi_in, [881](#)
- casecompare
 - seq64, [91](#)
- category
 - seq64::editable_event, [162](#), [163](#)
- category_string
 - seq64::editable_event, [162](#)
- category_t
 - seq64::editable_event, [158](#)
- cc_match
 - seq64::event, [193](#)
- change_event_data_lfo
 - seq64::sequence, [1098](#)
- change_event_data_range
 - seq64::sequence, [1097](#)
- change_focus
 - seq64::eventedit, [229](#)
 - seq64::seqedit, [942](#)

- change_horz
 - seq64::perfroll, [811](#)
 - seq64::perftime, [827](#)
 - seq64::seqdata, [914](#)
 - seq64::seqevent, [966](#)
 - seq64::seqroll, [1017](#)
 - seq64::seqtime, [1036](#)
- change_vert
 - seq64::eventslots, [249](#)
 - seq64::perfnames, [677](#)
 - seq64::perfroll, [811](#)
 - seq64::seqkeys, [980](#)
 - seq64::seqroll, [1017](#)
- channel_count
 - seq64::user_midi_bus, [1156](#)
- channel_match
 - seq64::sequence, [1109](#)
- channel_max
 - seq64::user_midi_bus, [1156](#)
- channel_string
 - seq64::editable_event, [166](#)
- channels_match
 - seq64::sequence, [1113](#)
- char_height
 - seq64::font, [260](#)
- char_width
 - seq64::font, [260](#)
- CharList
 - seq64::midi_list, [556](#)
- CharVector
 - seq64::midi_vector, [585](#)
- check_channel
 - seq64::event, [187](#)
- check_daemonize
 - seq64, [75](#)
- check_handles
 - seq64::Seq24PerfInput, [902](#)
- check_queued_tick
 - seq64::sequence, [1067](#)
- checklen
 - seq64::midifile, [625](#)
- choose_file
 - seq64::mainwnd, [447](#)
- choose_ppqn
 - seq64, [88](#)
- clamp
 - seq64, [98](#)
- clamp_group
 - seq64::perform, [778](#)
- clamp_track
 - seq64::perform, [777](#)
- clear
 - seq64::editable_events, [178](#)
 - seq64::event_list, [214](#)
 - seq64::midi_container, [511](#)
 - seq64::midi_info, [530](#)
 - seq64::midi_list, [557](#)
 - seq64::midi_port_info, [570](#)
 - seq64::midi_vector, [587](#)
 - seq64::rect, [863](#)
 - seq64::rtmidi_info, [890](#)
 - seq64::triggers, [1143](#)
- clear_all
 - seq64::perform, [714](#)
- clear_comments
 - seq64::rc_settings, [841](#)
 - seq64::user_settings, [1173](#)
- clear_current_screenset
 - seq64::perform, [732](#)
- clear_errors
 - seq64::midifile, [621](#)
- clear_flags
 - seq64::seqroll, [1020](#)
- clear_link
 - seq64::event, [198](#)
- clear_links
 - seq64::event_list, [216](#)
- clear_mute_groups
 - seq64::mainwnd, [442](#)
 - seq64::perform, [721](#)
- clear_sequence_triggers
 - seq64::perform, [747](#)
- clear_tempo_links
 - seq64::event_list, [217](#)
- clear_triggers
 - seq64::sequence, [1085](#)
- clear_window
 - seq64::gui_drawingarea_gtk2, [287](#)
- click
 - seq64::click, [144](#)
- client
 - seq64::jack_assistant, [334](#)
- client_handle
 - seq64::midi_jack, [539](#), [540](#)
 - seq64::midi_jack_info, [551](#), [552](#)
- client_name
 - seq64::jack_assistant, [335](#)
- client_open
 - seq64::jack_assistant, [335](#)
- client_uid
 - seq64::jack_assistant, [335](#)
- clip_timestamp
 - seq64::sequence, [1095](#)
- clock
 - seq64::busarray, [130](#)
 - seq64::businfo, [141](#)
 - seq64::mastermidibase, [473](#)
 - seq64::midibase, [602](#)
- clock_callback_disable
 - seq64::options, [645](#)
- clock_callback_mod
 - seq64::options, [645](#)
- clock_callback_off
 - seq64::options, [644](#)
- clock_callback_on
 - seq64::options, [645](#)

- clock_e
 - seq64, [47](#)
- clock_enabled
 - seq64::midibase, [597](#)
- clock_mod_callback
 - seq64::options, [645](#)
- clock_tick_duration_bogus
 - seq64, [66](#)
- clock_ticks_from_ppqn
 - seq64, [66](#)
- clocks_per_metronome
 - seq64::perform, [704](#)
 - seq64::sequence, [1060](#)
- close_client
 - seq64::midi_jack, [541](#)
- close_out
 - seq64::eventedit, [230](#)
- close_port
 - seq64::midi_jack, [541](#)
- collapse
 - seq64::perfedit, [664](#)
 - seq64::perform, [761](#)
- Color
 - seq64::font, [258](#)
 - seq64::gui_palette_gtk2, [304](#)
- color
 - seq64::sequence, [1055](#)
- colorbyte
 - seq64, [44](#)
- combine_bytes
 - seq64::perform, [708](#)
- comments_block
 - seq64::rc_settings, [841](#)
 - seq64::user_settings, [1173](#)
- complete_paste
 - seq64::seqroll, [1011](#)
- condition_var
 - seq64::condition_var, [149](#)
- config_directory
 - seq64::rc_settings, [847](#), [852](#)
- config_filename
 - seq64::rc_settings, [848](#), [853](#)
- config_filename_alt
 - seq64::rc_settings, [848](#), [853](#)
- config_filespec
 - seq64::rc_settings, [841](#)
- configfile
 - seq64::configfile, [151](#)
- connect
 - seq64::midi_jack_info, [552](#)
- connect_name
 - seq64::midi_info, [533](#)
 - seq64::midi_port_info, [571](#)
 - seq64::midibase, [594](#)
- connect_port
 - seq64::midi_jack, [541](#)
- const_iterator
 - seq64::editable_events, [172](#)
 - seq64::event_list, [210](#)
 - seq64::seqmenu, [989](#)
- const_reverse_iterator
 - seq64::editable_events, [172](#)
 - seq64::event_list, [210](#)
- container
 - seq64::midi_message, [562](#)
- continue_from
 - seq64::busarray, [130](#)
 - seq64::businfo, [141](#)
 - seq64::mastermidibase, [466](#)
 - seq64::midibase, [603](#)
- continue_sysex
 - seq64::rtmidi_in_data, [884](#)
- control_height
 - seq64::user_settings, [1178](#), [1184](#)
- controller_active
 - seq64::user_instrument, [1151](#)
 - seq64::user_settings, [1173](#)
- controller_count
 - seq64::user_instrument, [1150](#)
- controller_max
 - seq64::user_instrument, [1150](#)
- controller_name
 - seq64::user_instrument, [1150](#)
 - seq64::user_settings, [1173](#)
- controllers
 - seq64::user_instrument_t, [1153](#)
- controllers_active
 - seq64::user_instrument_t, [1153](#)
- convert_drop_xy
 - seq64::perfroll, [811](#)
 - seq64::seqroll, [1013](#)
- convert_sel_box_to_rect
 - seq64::seqroll, [1015](#)
- convert_t
 - seq64::seqevent, [967](#)
- convert_tn
 - seq64::seqroll, [1014](#)
- convert_tn_box_to_rect
 - seq64::seqroll, [1014](#), [1015](#)
- convert_x
 - seq64::perfroll, [809](#)
 - seq64::seqdata, [914](#)
 - seq64::seqevent, [966](#)
- convert_xy
 - seq64::perfroll, [808](#)
 - seq64::seqroll, [1013](#)
- convert_y
 - seq64::eventslots, [248](#)
 - seq64::perfnames, [676](#)
 - seq64::seqkeys, [979](#)
- copy
 - seq64::perfedit, [664](#)
 - seq64::perform, [761](#)
 - seq64::triggers, [1142](#)
- copy_definitions
 - seq64::user_instrument, [1152](#)

- seq64::user_midi_bus, [1157](#)
- copy_events
 - seq64::sequence, [1108](#)
- copy_selected
 - seq64::sequence, [1091](#)
 - seq64::triggers, [1140](#)
- copy_selected_trigger
 - seq64::sequence, [1083](#)
- copy_triggers
 - seq64::perform, [748](#)
 - seq64::sequence, [1085](#)
- count
 - seq64::busarray, [129](#)
 - seq64::editable_events, [176](#)
 - seq64::event_list, [212](#)
 - seq64::eventslots, [242](#)
 - seq64::midi_message, [562](#)
 - seq64::midi_queue, [573](#)
 - seq64::midi_splitter, [578](#)
 - seq64::triggers, [1133](#)
- count_selected_events
 - seq64::event_list, [218](#)
- count_selected_notes
 - seq64::event_list, [218](#)
- create_jack_client
 - seq64, [83](#)
- create_lash_driver
 - seq64, [86](#)
- create_master_bus
 - seq64::perform, [779](#)
- create_menu_image
 - seq64::seqedit, [941](#)
- create_menus
 - seq64::seqedit, [938](#)
- create_ringbuffer
 - seq64::midi_jack, [541](#)
- create_seqedit
 - seq64::seqmenu, [993](#)
- create_tempo_event
 - seq64, [76](#)
- current_date_time
 - seq64, [71](#)
- current_event
 - seq64::editable_events, [178](#)
- current_index
 - seq64::eventslots, [243](#)
- current_screenset_notepad
 - seq64::perform, [758](#)
- current_seq
 - seq64::seqmenu, [990](#)
- current_x
 - seq64::gui_drawingarea_gtk2, [286](#)
- current_y
 - seq64::gui_drawingarea_gtk2, [286](#)
- cut_selected
 - seq64::sequence, [1091](#)
- cut_selected_trigger
 - seq64::sequence, [1083](#)
- daemonize
 - seq64, [75](#)
- dark_blue
 - seq64::gui_palette_gtk2, [308](#)
- dark_cyan
 - seq64::gui_palette_gtk2, [308](#)
- dark_green
 - seq64::gui_palette_gtk2, [308](#)
- dark_grey_paint
 - seq64::gui_palette_gtk2, [308](#)
- dark_magenta
 - seq64::gui_palette_gtk2, [308](#)
- dark_orange
 - seq64::gui_palette_gtk2, [308](#)
- dark_red
 - seq64::gui_palette_gtk2, [307](#)
- data
 - seq64::event, [199](#)
 - seq64::midi_control, [518](#)
- data_string
 - seq64::editable_event, [166](#)
- deactivate
 - seq64::businfo, [140](#)
- deallocate
 - seq64::midi_queue, [574](#)
- debug_text
 - seq64::mainwnd, [435](#)
- decrement_beats_per_minute
 - seq64::perform, [738](#)
- decrement_bottom
 - seq64::eventslots, [251](#)
- decrement_current
 - seq64::eventslots, [250](#)
- decrement_data1
 - seq64::event, [194](#)
- decrement_data2
 - seq64::event, [195](#)
- decrement_offset
 - seq64::trigger, [1127](#)
- decrement_screenset
 - seq64::perform, [739](#)
- decrement_selected
 - seq64::sequence, [1099](#)
- decrement_tick_end
 - seq64::trigger, [1127](#)
- decrement_tick_start
 - seq64::trigger, [1126](#)
- decrement_top
 - seq64::eventslots, [250](#)
- default_name
 - seq64::sequence, [1062](#)
- deinit
 - seq64::jack_assistant, [329](#)
- deinit_in
 - seq64::midibase, [601](#)
- deinit_jack_transport
 - seq64::perform, [773](#)
- delete_api

- seq64::rtmidi, 877
- seq64::rtmidi_info, 895
- delete_current_event
 - seq64::eventslots, 245
- delete_current_sequence
 - seq64::seqmenu, 992
- delete_lash_driver
 - seq64, 87
- delete_selected_triggers
 - seq64::sequence, 1083
- delete_sequence
 - seq64::perform, 716
- delete_trigger
 - seq64::perform, 749
 - seq64::sequence, 1079
- delta_time_us_to_ticks
 - seq64, 65
- device_ignore
 - seq64::rc_settings, 845, 851
- device_ignore_num
 - seq64::rc_settings, 845, 851
- disconnect
 - seq64::midi_jack_info, 552
- display_name
 - seq64::midibase, 594, 604
- divisions
 - seq64::midi_measures, 560
- do_action
 - seq64::seqedit, 942
- do_input
 - seq64::rtmidi_in_data, 883, 884
- done
 - seq64::midi_container, 511
 - seq64::midi_list, 557
 - seq64::midi_vector, 586
- double_ticks_from_ppqn
 - seq64, 67
- draw_all
 - seq64::perfroll, 807
- draw_area
 - seq64::seqkeys, 979
- draw_background
 - seq64::perftime, 827
 - seq64::seqevent, 964
- draw_background_on
 - seq64::perfroll, 810
- draw_background_on_pixmap
 - seq64::seqroll, 1009
- draw_drawable
 - seq64::gui_drawingarea_gtk2, 296
- draw_drawable_row
 - seq64::perfroll, 810
- draw_event
 - seq64::eventslots, 248
- draw_events
 - seq64::eventslots, 249
- draw_events_on
 - seq64::seqdata, 914
- seq64::seqevent, 966
- seq64::seqroll, 1016
- draw_events_on_pixmap
 - seq64::seqdata, 915
 - seq64::seqevent, 964
 - seq64::seqroll, 1009
- draw_key
 - seq64::seqkeys, 979
- draw_line
 - seq64::gui_drawingarea_gtk2, 288–290
- draw_line_on_pixmap
 - seq64::gui_drawingarea_gtk2, 288, 289
- draw_line_on_window
 - seq64::seqdata, 912
- draw_marker_on_sequence
 - seq64::mainwid, 416
- draw_normal_rectangle_on_pixmap
 - seq64::gui_drawingarea_gtk2, 295
- draw_pixmap_on_window
 - seq64::mainwid, 415
 - seq64::perftime, 829
 - seq64::seqdata, 915
 - seq64::seqevent, 964
 - seq64::seqtime, 1035
- draw_progress
 - seq64::perfroll, 808
- draw_progress_on_window
 - seq64::perftime, 827
 - seq64::seqroll, 1010
 - seq64::seqtime, 1035
- draw_rectangle
 - seq64::gui_drawingarea_gtk2, 291–294
- draw_rectangle_on_pixmap
 - seq64::gui_drawingarea_gtk2, 294, 295
- draw_selection_on_window
 - seq64::seqevent, 964
 - seq64::seqroll, 1009
- draw_sequence
 - seq64::perfnames, 676
 - seq64::perfroll, 810
- draw_sequence_on
 - seq64::perfroll, 810
- draw_sequence_on_pixmap
 - seq64::mainwid, 417
- draw_sequence_pixmap_on_window
 - seq64::mainwid, 418
- draw_sequences
 - seq64::perfedit, 665
 - seq64::perfnames, 676
- draw_sequences_on_pixmap
 - seq64::mainwid, 418
- draw_type_t
 - seq64, 49
- dref
 - seq64::editable_events, 176
 - seq64::event_list, 215
- drop_action
 - seq64::perfroll, 814

seq64::seqroll, 1022

drop_event
seq64::seqevent, 966

drop_x
seq64::gui_drawingarea_gtk2, 286

drop_y
seq64::gui_drawingarea_gtk2, 286

dump_midi_input
seq64::mastermidibase, 469

dump_setting_summary
seq64::userfile, 1206

dump_summary
seq64::user_settings, 1184

EVENT_AFTERTOUCHE
seq64, 100

EVENT_ANY
seq64, 100

EVENT_CHANNEL_PRESSURE
seq64, 100

EVENT_CLEAR_CHAN_MASK
seq64, 106

EVENT_CONTROL_CHANGE
seq64, 100

EVENT_CTRL_BALANCE
seq64, 101

EVENT_CTRL_EXPRESSION
seq64, 101

EVENT_CTRL_PAN
seq64, 101

EVENT_CTRL_VOLUME
seq64, 101

EVENT_GET_CHAN_MASK
seq64, 106

EVENT_META_COPYRIGHT
seq64, 104

EVENT_META_CUE_POINT
seq64, 105

EVENT_META_END_OF_TRACK
seq64, 105

EVENT_META_ILLEGAL
seq64, 106

EVENT_META_INSTRUMENT
seq64, 104

EVENT_META_KEY_SIGNATURE
seq64, 106

EVENT_META_LYRIC
seq64, 104

EVENT_META_MARKER
seq64, 105

EVENT_META_MIDI_CHANNEL
seq64, 105

EVENT_META_MIDI_PORT
seq64, 105

EVENT_META_SEQ_NUMBER
seq64, 104

EVENT_META_SEQSPEC
seq64, 106

EVENT_META_SET_TEMPO
seq64, 105

EVENT_META_SMPTE_OFFSET
seq64, 105

EVENT_META_TEXT_EVENT
seq64, 104

EVENT_META_TIME_SIGNATURE
seq64, 105

EVENT_META_TRACK_NAME
seq64, 104

EVENT_MIDI_ACTIVE_SENSE
seq64, 103

EVENT_MIDI_CLOCK
seq64, 103

EVENT_MIDI_CONTINUE
seq64, 103

EVENT_MIDI_META
seq64, 104

EVENT_MIDI_QUARTER_FRAME
seq64, 102

EVENT_MIDI_REALTIME
seq64, 101

EVENT_MIDI_RESET
seq64, 103

EVENT_MIDI_SONG_F4
seq64, 102

EVENT_MIDI_SONG_F5
seq64, 102

EVENT_MIDI_SONG_F9
seq64, 103

EVENT_MIDI_SONG_FD
seq64, 103

EVENT_MIDI_SONG_POS
seq64, 102

EVENT_MIDI_SONG_SELECT
seq64, 102

EVENT_MIDI_START
seq64, 103

EVENT_MIDI_STOP
seq64, 103

EVENT_MIDI_SYSEX_CONTINUE
seq64, 102

EVENT_MIDI_SYSEX_END
seq64, 102

EVENT_MIDI_SYSEX
seq64, 101

EVENT_MIDI_TUNE_SELECT
seq64, 102

EVENT_NOTE_OFF
seq64, 100

EVENT_NOTE_ON
seq64, 100

EVENT_NULL_CHANNEL
seq64, 106

EVENT_PITCH_WHEEL
seq64, 101

EVENT_PROGRAM_CHANGE
seq64, 100

EVENT_STATUS_BIT

- seq64, [99](#)
- EVENTS_ALL
 - seq64, [106](#)
- EVENTS_UNSELECTED
 - seq64, [107](#)
- edit_action_t
 - seq64, [50](#)
- edit_callback_notepad
 - seq64::mainwnd, [437](#)
- edit_field_has_focus
 - seq64::mainwnd, [447](#)
- edit_mode
 - seq64::sequence, [1056](#)
- edit_mode_t
 - seq64, [49](#)
- edit_tempo_track_number
 - seq64::options, [645](#)
- editable_event
 - seq64::editable_event, [159](#), [160](#)
- editable_events
 - seq64::editable_events, [172–174](#)
 - seq64::event_list, [219](#)
- emit_clock
 - seq64::mastermidibase, [467](#)
- empty
 - seq64::event_list, [212](#)
 - seq64::midi_message, [563](#)
 - seq64::midi_queue, [573](#)
- end
 - seq64::editable_events, [176](#)
 - seq64::event_list, [212](#)
- enqueue_draw
 - seq64::eventedit, [226](#)
 - seq64::eventslots, [248](#)
 - seq64::perfedit, [660](#)
 - seq64::perfnames, [676](#)
 - seq64::perffroll, [811](#)
 - seq64::perftime, [827](#)
- enregister
 - seq64::perform, [707](#)
- enregister_peer
 - seq64::perfedit, [661](#)
- enregister_perfects
 - seq64::mainwnd, [442](#)
- error
 - seq64::midi_api, [504](#)
 - seq64::midi_info, [534](#)
- error_is_fatal
 - seq64::midifile, [620](#)
- error_message
 - seq64, [91](#)
 - seq64::jack_assistant, [337](#)
 - seq64::midifile, [620](#)
 - seq64::optionsfile, [654](#)
- event
 - seq64::event, [185](#)
- event_count
 - seq64::eventslots, [242](#)
- seq64::sequence, [1056](#)
- event_edit
 - seq64::keys_perform, [364](#)
- event_in_range
 - seq64::sequence, [1110](#)
- event_key
 - seq64::event_list::event_key, [206](#)
- event_length
 - seq64::editable_event::meta_length_t, [484](#)
- event_list
 - seq64::event_list, [211](#)
- event_name
 - seq64::editable_event::name_value_t, [642](#)
- event_value
 - seq64::editable_event::meta_length_t, [484](#)
 - seq64::editable_event::name_value_t, [642](#)
- EventStack
 - seq64::sequence, [1052](#)
- eventedit
 - seq64::eventedit, [224](#)
 - seq64::eventslots, [254](#)
- Events
 - seq64::editable_events, [172](#)
 - seq64::event_list, [210](#)
- events
 - seq64::editable_events, [175](#)
 - seq64::event_list, [219](#)
 - seq64::sequence, [1053](#)
- EventsPair
 - seq64::editable_events, [172](#)
 - seq64::event_list, [210](#)
- eventslots
 - seq64::editable_events, [179](#)
 - seq64::eventedit, [232](#)
 - seq64::eventslots, [242](#)
- ex_data_string
 - seq64::editable_event, [165](#)
- exact_split
 - seq64::triggers, [1137](#)
- exact_split_trigger
 - seq64::sequence, [1079](#)
- expand
 - seq64::perfedit, [664](#)
 - seq64::perform, [761](#)
- extend
 - seq64::sequence, [1065](#)
- extract_bus_name
 - seq64, [70](#)
- extract_names
 - seq64::midi_jack_info, [553](#)
- extract_port_name
 - seq64, [70](#)
- extract_port_names
 - seq64, [70](#)
- extract_timing_numbers
 - seq64, [52](#)
- FF_RW_timeout
 - seq64, [98](#)

- seq64::perform, 708
- FF_rewind
 - seq64::perform, 708
- fast_forward
 - seq64::keys_perform, 366
 - seq64::perfedit, 662
 - seq64::perform, 710
- ff_rw_button_t
 - seq64::perform, 700
- ff_rw_type
 - seq64::perform, 710
- fg_color
 - seq64::gui_palette_gtk2, 310, 311
- file_access
 - seq64, 76
- file_accessible
 - seq64, 77
- file_executable
 - seq64, 78
- file_exists
 - seq64, 76
- file_exit
 - seq64::mainwnd, 446
- file_extension
 - seq64, 80
- file_extension_match
 - seq64, 81
- file_import_dialog
 - seq64::mainwnd, 444
- file_is_directory
 - seq64, 78
- file_new
 - seq64::mainwnd, 443
- file_open
 - seq64::mainwnd, 443
- file_readable
 - seq64, 77
- file_save
 - seq64::mainwnd, 444
- file_save_as
 - seq64::mainwnd, 445
- file_writable
 - seq64, 77
- filename
 - seq64::rc_settings, 845
- fill
 - seq64::midi_container, 509
- fill_background_pixmap
 - seq64::perfroll, 807
- fill_background_window
 - seq64::mainwid, 415
- fill_meta_track_end
 - seq64::midi_container, 514
- fill_proprietary
 - seq64::midi_container, 514
- fill_seq_name
 - seq64::midi_container, 514
- fill_seq_number
 - seq64::midi_container, 513
- fill_top_bar
 - seq64::seqedit, 938
- filter_by_channel
 - seq64::mastermidibase, 464
 - seq64::perform, 706
 - seq64::rc_settings, 844, 851
- filter_callback
 - seq64::options, 646
- finalize_sequence
 - seq64::midifile, 621
- finish
 - seq64::perform, 715
- first_message
 - seq64::rtmidi_in_data, 884
- flush
 - seq64::mastermidibase, 468
 - seq64::midibase, 602
- follow_change_callback
 - seq64::seqedit, 936
- follow_progress
 - seq64::perform, 709
 - seq64::perfroll, 807
 - seq64::seqroll, 1011
- follow_transport
 - seq64::keys_perform, 365, 366
- font
 - seq64::font, 259
- font_render
 - seq64, 93
- force_draw
 - seq64::gui_drawingarea_gtk2, 287
 - seq64::seqevent, 965
 - seq64::seqkeys, 978
 - seq64::seqroll, 1012
- format_timestamp
 - seq64::editable_event, 165
- front
 - seq64::midi_queue, 574
- FruityPerfInput
 - seq64::FruityPerfInput, 264
 - seq64::triggers, 1146
- FruitySeqEventInput
 - seq64::FruitySeqEventInput, 269
- FruitySeqRollInput
 - seq64::FruitySeqRollInput, 273
 - seq64::seqkeys, 983
- full
 - seq64::midi_queue, 573
- full_port_count
 - seq64::midi_info, 530
 - seq64::rtmidi, 876
 - seq64::rtmidi_info, 891
- g_midi_control_limit
 - seq64, 116
- g_rc_settings
 - seq64, 122
- g_user_settings

- seq64, [122](#)
- get
 - seq64::midi_container, [511](#)
 - seq64::midi_list, [557](#)
 - seq64::midi_vector, [587](#)
 - seq64::rect, [862](#)
- get_32nds_per_quarter
 - seq64::perform, [704](#)
 - seq64::sequence, [1060](#)
- get_all_port_info
 - seq64::midi_alsa_info, [495](#)
 - seq64::midi_info, [534](#)
 - seq64::midi_jack_info, [552](#)
 - seq64::rtmidi_info, [892](#)
- get_api
 - seq64::rtmidi, [876](#), [877](#)
- get_api_info
 - seq64::rtmidi_info, [894](#)
- get_bank_name
 - seq64::perform, [756](#)
- get_beat_width
 - seq64::jack_assistant, [327](#)
 - seq64::perform, [703](#)
 - seq64::sequence, [1060](#)
- get_beats_per_bar
 - seq64::perform, [703](#)
 - seq64::sequence, [1059](#)
- get_beats_per_measure
 - seq64::jack_assistant, [327](#)
- get_beats_per_minute
 - seq64::jack_assistant, [328](#)
 - seq64::mastermidibase, [464](#)
 - seq64::perform, [721](#)
- get_bpm
 - seq64::mastermidibase, [465](#)
- get_bus_id
 - seq64::midi_info, [532](#)
 - seq64::midi_port_info, [570](#)
 - seq64::midibase, [595](#)
 - seq64::rtmidi, [875](#)
 - seq64::rtmidi_info, [891](#)
- get_bus_index
 - seq64::midibase, [595](#)
- get_bus_name
 - seq64::midi_info, [532](#)
 - seq64::midi_port_info, [570](#)
 - seq64::rtmidi, [875](#)
 - seq64::rtmidi_info, [891](#)
- get_channel
 - seq64::event, [187](#)
- get_client
 - seq64::midi_alsa, [487](#)
- get_clipboard_box
 - seq64::sequence, [1093](#)
- get_clock
 - seq64::busarray, [132](#)
 - seq64::mastermidibase, [472](#)
 - seq64::midibase, [597](#)
 - seq64::perform, [780](#)
- get_clock_mod
 - seq64::midibase, [600](#)
- get_color
 - seq64::gui_drawingarea_gtk2, [286](#)
 - seq64::gui_palette_gtk2, [305](#)
- get_color_ex
 - seq64::gui_palette_gtk2, [305](#)
- get_compiled_api
 - seq64::rtmidi_info, [890](#)
- get_compound_option
 - seq64, [89](#)
- get_current_directory
 - seq64, [79](#)
- get_current_jack_position
 - seq64, [85](#)
 - seq64::jack_assistant, [340](#)
 - seq64::perform, [785](#)
- get_current_sequence
 - seq64::seqmenu, [991](#)
- get_data
 - seq64::event, [194](#)
- get_editing
 - seq64::sequence, [1063](#)
- get_error_message
 - seq64::configfile, [153](#)
- get_expanded_record
 - seq64::seqroll, [1011](#)
- get_file_pos
 - seq64::midifile, [621](#)
- get_follow_transport
 - seq64::jack_assistant, [334](#)
 - seq64::perform, [709](#)
- get_full_path
 - seq64, [79](#)
- get_group_mute_state
 - seq64::perform, [728](#)
- get_hold_undo
 - seq64::sequence, [1057](#)
- get_input
 - seq64::busarray, [134](#)
 - seq64::mastermidibase, [471](#)
 - seq64::midi_info, [533](#)
 - seq64::midi_port_info, [570](#)
 - seq64::midibase, [597](#)
 - seq64::perform, [781](#)
 - seq64::rtmidi_info, [892](#)
- get_jack_client_info
 - seq64::jack_assistant, [336](#)
- get_jack_mode
 - seq64::jack_assistant, [333](#)
- get_jack_pos
 - seq64::jack_assistant, [333](#)
- get_jack_stop_tick
 - seq64::jack_assistant, [333](#)
- get_jack_tick
 - seq64::jack_assistant, [333](#)
 - seq64::perform, [717](#)

- get_key_count
 - seq64::keys_perform, 369
 - seq64::perform, 732
- get_key_events
 - seq64::keys_perform, 369
 - seq64::perform, 732
- get_key_events_rev
 - seq64::keys_perform, 369
 - seq64::perform, 732
- get_key_groups
 - seq64::keys_perform, 369
 - seq64::perform, 732
- get_key_groups_rev
 - seq64::keys_perform, 369
 - seq64::perform, 733
- get_keys
 - seq64::keys_perform, 358
- get_last_tick
 - seq64::sequence, 1065
- get_left_tick
 - seq64::perform, 718
- get_length
 - seq64::editable_events, 177
 - seq64::event_list, 212
 - seq64::eventslots, 242
 - seq64::sequence, 1065
- get_linked
 - seq64::event, 198
- get_loop_reset
 - seq64::sequence, 1110
- get_max_trigger
 - seq64::perform, 751
 - seq64::sequence, 1084
- get_maximum
 - seq64::triggers, 1141
- get_measures
 - seq64::seqedit, 933
- get_message
 - seq64::rterror, 869
- get_midi_bus
 - seq64::sequence, 1086
- get_midi_bus_name
 - seq64::busarray, 132
- get_midi_channel
 - seq64::sequence, 1073
- get_midi_event
 - seq64::busarray, 135
 - seq64::mastermidibase, 470
 - seq64::midibase, 600
- get_midi_in_bus_name
 - seq64::mastermidibase, 469
- get_midi_out_bus_name
 - seq64::mastermidibase, 469
- get_minmax_note_events
 - seq64::sequence, 1104
- get_next_event
 - seq64::sequence, 1104
- get_next_event_ex
 - seq64::sequence, 1105
- get_next_note_event
 - seq64::sequence, 1103
- get_next_trigger
 - seq64::sequence, 1105
- get_note
 - seq64::event, 199
- get_note_velocity
 - seq64::event, 200
- get_num_in_buses
 - seq64::mastermidibase, 464
- get_num_measures
 - seq64::sequence, 1064
- get_num_out_buses
 - seq64::mastermidibase, 464
- get_num_selected_events
 - seq64::sequence, 1090
- get_num_selected_notes
 - seq64::sequence, 1090
- get_onsets_selected_box
 - seq64::sequence, 1093
- get_overwrite_rec
 - seq64::sequence, 1109
- get_playing
 - seq64::sequence, 1066
- get_playing_screenset
 - seq64::perform, 753
- get_port
 - seq64::midi_alsa, 487
- get_port_count
 - seq64::midi_info, 532
 - seq64::midi_port_info, 570
 - seq64::rtmidi, 876
 - seq64::rtmidi_info, 891
- get_port_id
 - seq64::midi_info, 533
 - seq64::midi_port_info, 570
 - seq64::midibase, 595
 - seq64::rtmidi, 876
 - seq64::rtmidi_info, 892
- get_port_name
 - seq64::midi_info, 533
 - seq64::midi_port_info, 570
 - seq64::rtmidi, 876
 - seq64::rtmidi_info, 892
- get_port_statuses
 - seq64::mastermidibase, 473
- get_ppqn
 - seq64::jack_assistant, 327
 - seq64::mastermidibase, 465
 - seq64::sequence, 1059
- get_progress_follow
 - seq64::seqroll, 1012
- get_quantized_rec
 - seq64::sequence, 1069
- get_queue_number
 - seq64::midi_port_info, 571
- get_queued

- seq64::sequence, 1067
- get_queued_tick
 - seq64::sequence, 1067
- get_raise
 - seq64::sequence, 1063
- get_rank
 - seq64::event, 203
- get_recording
 - seq64::sequence, 1068
- get_right_tick
 - seq64::perform, 720
- get_screenset_notepad
 - seq64::perform, 765
- get_selected_box
 - seq64::seqroll, 1016
 - seq64::sequence, 1092
- get_selected_end
 - seq64::triggers, 1141
- get_selected_start
 - seq64::triggers, 1141
- get_sequence
 - seq64::mastermidibase, 465
 - seq64::perform, 740
 - seq64::seqmenu, 991
- get_sequence_color
 - seq64::gui_drawingarea_gtk2, 286
 - seq64::perform, 756
- get_snap_tick
 - seq64::sequence, 1068
- get_song_mute
 - seq64::sequence, 1061
- get_start_tick
 - seq64::perform, 718
- get_state
 - seq64::triggers, 1138
- get_state_name
 - seq64::jack_assistant, 326
- get_status
 - seq64::event, 193
- get_sysex
 - seq64::event, 197
- get_sysex_size
 - seq64::event, 197
- get_system
 - seq64::midi_info, 533
 - seq64::midi_port_info, 571
 - seq64::rtmidi_info, 892
- get_tempo_track_number
 - seq64::perform, 704
- get_thru
 - seq64::sequence, 1069
- get_tick
 - seq64::perform, 717
- get_timestamp
 - seq64::event, 187
- get_toggle_jack
 - seq64::perfedit, 661
 - seq64::perform, 708
- get_transposable
 - seq64::sequence, 1062
- get_transpose
 - seq64::perform, 721
- get_trigger_offset
 - seq64::sequence, 1086
- get_trigger_paste_tick
 - seq64::sequence, 1055
 - seq64::triggers, 1144
- get_trigger_state
 - seq64::perform, 749
 - seq64::sequence, 1080
- get_triggers
 - seq64::sequence, 1080
- get_unit_measure
 - seq64::sequence, 1109
- get_version
 - seq64::rtmidi_info, 890
- get_virtual
 - seq64::midi_info, 533
 - seq64::midi_port_info, 571
 - seq64::rtmidi_info, 892
- getType
 - seq64::rterror, 869
- global_queue
 - seq64::midi_info, 534
 - seq64::rtmidi_info, 893
- global_seq_feature
 - seq64::user_settings, 1178
- gmute_tracks
 - seq64::user_settings, 1175
- grab_input_stream
 - seq64::midifile, 621
- green
 - seq64::gui_palette_gtk2, 309
- grey_paint
 - seq64::gui_palette_gtk2, 308
- grid_brackets
 - seq64::user_settings, 1175, 1181
- grid_is_black
 - seq64::user_settings, 1175
- grid_is_normal
 - seq64::user_settings, 1174
- grid_is_white
 - seq64::user_settings, 1174
- grid_style
 - seq64::user_settings, 1174, 1182
- group_learn
 - seq64::keys_perform, 362
- group_max
 - seq64::keys_perform, 372
 - seq64::perform, 702
- group_off
 - seq64::keys_perform, 362
- group_on
 - seq64::keys_perform, 362
- grow
 - seq64::perfedit, 664

- seq64::triggers, [1137](#)
- grow_edit_t
 - seq64::triggers, [1132](#)
- grow_selected
 - seq64::sequence, [1099](#)
- grow_selected_notes
 - seq64::seqroll, [1018](#)
- grow_trigger
 - seq64::sequence, [1079](#)
- growing
 - seq64::perfroll, [814](#)
 - seq64::seqroll, [1022](#)
- gs_mainwid_pointer
 - seq64, [123](#)
- gs_perfedit_pointer_0
 - seq64, [123](#)
- gs_perfedit_pointer_1
 - seq64, [123](#)
- Gtk, [27](#)
- gtk_drawarea_init
 - seq64::gui_drawingarea_gtk2, [297](#)
- gui
 - seq64::perform, [705](#)
- gui_assistant
 - seq64::gui_assistant, [277](#)
- gui_assistant_gtk2
 - seq64::gui_assistant_gtk2, [280](#)
- gui_drawingarea_gtk2
 - seq64::gui_drawingarea_gtk2, [284](#), [285](#)
- gui_palette_gtk2
 - seq64::gui_palette_gtk2, [305](#)
- gui_window_gtk2
 - seq64::gui_window_gtk2, [317](#)
- half_split
 - seq64::triggers, [1137](#)
- half_split_trigger
 - seq64::sequence, [1078](#)
- handle_cancel
 - seq64::eventedit, [231](#)
- handle_close
 - seq64::eventedit, [230](#)
 - seq64::seqedit, [943](#)
- handle_config
 - seq64::lash, [398](#)
- handle_delete
 - seq64::eventedit, [230](#)
- handle_event
 - seq64::lash, [398](#)
- handle_insert
 - seq64::eventedit, [230](#)
- handle_midi_control
 - seq64::perform, [763](#)
- handle_midi_control_event
 - seq64::perform, [765](#)
- handle_midi_control_ex
 - seq64::perform, [764](#)
- handle_modify
 - seq64::eventedit, [230](#)
- handle_motion_key
 - seq64::FruityPerfInput, [265](#)
 - seq64::Seq24PerfInput, [901](#)
 - seq64::perfroll, [813](#)
- handle_save
 - seq64::eventedit, [230](#)
- handle_signal
 - seq64::mainwnd, [435](#)
- handle_size
 - seq64::sequence, [1110](#)
- has_tempo
 - seq64::event_list, [214](#)
- has_time_signature
 - seq64::event_list, [214](#)
- have_redo
 - seq64::perform, [757](#)
 - seq64::sequence, [1057](#)
- have_undo
 - seq64::perform, [756](#)
 - seq64::sequence, [1057](#)
- height
 - seq64::gui_drawingarea_gtk2, [286](#)
 - seq64::rect, [865](#)
- height_incr
 - seq64::rect, [866](#)
- help_check
 - seq64, [71](#)
 - seq64::rc_settings, [855](#)
- highlight
 - seq64::perform, [739](#)
- home_config_directory
 - seq64::rc_settings, [847](#)
- horizontal_adjust
 - seq64::perfroll, [812](#)
 - seq64::seqedit, [932](#)
 - seq64::seqroll, [1012](#)
- horizontal_set
 - seq64::perfroll, [812](#)
 - seq64::seqedit, [933](#)
- idle_progress
 - seq64::maintime, [407](#)
 - seq64::perftime, [829](#)
 - seq64::seqroll, [1017](#)
 - seq64::seqtime, [1037](#)
- idle_redraw
 - seq64::seqdata, [912](#)
 - seq64::seqevent, [965](#)
 - seq64::seqroll, [1016](#)
- ignore_flags
 - seq64::rtmidi_in_data, [883](#)
- in_range
 - seq64::midi_control, [520](#)
- inc_draw_marker
 - seq64::sequence, [1102](#)
- increment
 - seq64::midi_splitter, [577](#)
- increment_beats_per_minute
 - seq64::perform, [738](#)

- increment_bottom
 - seq64::eventslots, [251](#)
- increment_current
 - seq64::eventslots, [251](#)
- increment_data1
 - seq64::event, [194](#)
- increment_data2
 - seq64::event, [194](#)
- increment_offset
 - seq64::trigger, [1127](#)
- increment_screenset
 - seq64::perform, [739](#)
- increment_selected
 - seq64::sequence, [1098](#)
- increment_size
 - seq64::perfroll, [807](#)
 - seq64::perftime, [827](#)
- increment_tick_end
 - seq64::trigger, [1126](#)
- increment_tick_start
 - seq64::trigger, [1126](#)
- increment_top
 - seq64::eventslots, [250](#)
- independent
 - seq64::mainwnd, [436](#)
- info_message
 - seq64, [91](#)
 - seq64::jack_assistant, [337](#)
- init
 - seq64::font, [259](#)
 - seq64::jack_assistant, [328](#)
 - seq64::lash, [398](#)
 - seq64::mastermidibase, [463](#)
- init_before_show
 - seq64::perfedit, [660](#)
 - seq64::perfroll, [807](#)
- init_clock
 - seq64::busarray, [130](#)
 - seq64::businfo, [140](#), [141](#)
 - seq64::mastermidibase, [467](#)
 - seq64::midibase, [603](#)
- init_in
 - seq64::midibase, [600](#)
- init_in_sub
 - seq64::midibase, [601](#)
- init_input
 - seq64::businfo, [140](#)
- init_jack_transport
 - seq64::perform, [773](#)
- init_out
 - seq64::midibase, [600](#)
- init_out_sub
 - seq64::midibase, [601](#)
- initialize
 - seq64::busarray, [129](#)
 - seq64::businfo, [139](#)
 - seq64::gui_palette_gtk2, [305](#)
 - seq64::midi_splitter, [577](#)
- initialize_sequence
 - seq64::midifile, [621](#)
- initialized
 - seq64::businfo, [139](#)
- inner_start
 - seq64::perform, [777](#)
- inner_stop
 - seq64::perform, [777](#)
- input
 - seq64::mastermidibase, [474](#)
- input_callback
 - seq64::options, [646](#)
- input_data
 - seq64::midi_api, [505](#)
- input_func
 - seq64::perform, [726](#)
- input_ports
 - seq64::midi_info, [530](#)
- input_thread_func
 - seq64, [88](#)
 - seq64::perform, [782](#)
- insert_event
 - seq64::eventslots, [244](#), [245](#)
- install_sequence
 - seq64::perform, [776](#)
- install_signal_handlers
 - seq64::mainwnd, [447](#)
- instrument
 - seq64::user_instrument_t, [1153](#)
 - seq64::user_midi_bus, [1156](#)
 - seq64::user_midi_bus_t, [1158](#)
 - seq64::user_settings, [1171](#)
- instrument_controller_active
 - seq64::user_settings, [1173](#)
- instrument_controller_name
 - seq64::user_settings, [1173](#)
- instrument_count
 - seq64::user_settings, [1172](#)
- instrument_name
 - seq64::user_settings, [1172](#)
- InstrumentConstIterator
 - seq64::user_settings, [1169](#)
- InstrumentIterator
 - seq64::user_settings, [1169](#)
- Instruments
 - seq64::user_settings, [1169](#)
- interaction_method
 - seq64::rc_settings, [845](#), [852](#)
- interaction_method_t
 - seq64, [48](#)
- intersect
 - seq64::triggers, [1139](#), [1140](#)
- intersect_events
 - seq64::sequence, [1082](#)
- intersect_notes
 - seq64::sequence, [1082](#)
- intersect_triggers
 - seq64::perform, [751](#)

- seq64::sequence, 1081
- invalid_key
 - seq64, 86
- inverse_active
 - seq64::midi_control, 518
- inverse_colors
 - seq64::user_settings, 1180, 1188
- is
 - seq64::keystroke, 394
- is_active
 - seq64::perform, 720
- is_adding
 - seq64::perfull, 813
- is_adding_pressed
 - seq64::perfull, 813
- is_black_key
 - seq64::seqkeys, 980
- is_channel_msg
 - seq64::event, 188
- is_control_status
 - seq64::perform, 702
- is_ctrl_key
 - seq64, 93–95
- is_ctrl_shift_key
 - seq64, 96
- is_current_seq_active
 - seq64::seqmenu, 991
- is_current_seq_in_edit
 - seq64::seqmenu, 991
- is_default_mainwid_size
 - seq64::user_settings, 1175
- is_default_name
 - seq64::sequence, 1062
- is_delete
 - seq64::keystroke, 394
- is_desired_cc_or_not_cc
 - seq64::event, 190
- is_dirty_edit
 - seq64::perform, 752
 - seq64::sequence, 1072
- is_dirty_main
 - seq64::perform, 751
 - seq64::sequence, 1071
- is_dirty_names
 - seq64::perform, 752
 - seq64::sequence, 1072
- is_dirty_perf
 - seq64::perform, 752
 - seq64::sequence, 1072
- is_drag_motion
 - seq64, 97
- is_dumping
 - seq64::mastermidibase, 465
- is_edit_sequence
 - seq64::perform, 703
 - seq64::seqmenu, 991
- is_ex_data
 - seq64::event, 202
- is_exportable
 - seq64::perform, 753
- is_group_learning
 - seq64::perform, 760
- is_input_port
 - seq64::midi_api, 499
 - seq64::midibase, 596
- is_input_system_port
 - seq64::mastermidibase, 472
 - seq64::perform, 781
- is_inverse
 - seq64::gui_palette_gtk2, 307
- is_jack_master
 - seq64::perform, 707
- is_jack_running
 - seq64::perform, 707
- is_keep_queue
 - seq64::perform, 724
- is_key_signature
 - seq64::event, 203
- is_left
 - seq64::click, 145
- is_left_drag
 - seq64, 97
- is_letter
 - seq64::keystroke, 392
- is_linked
 - seq64::event, 198
- is_marked
 - seq64::event, 199
- is_master
 - seq64::jack_assistant, 327
- is_meta
 - seq64::event, 202
- is_middle
 - seq64::click, 145
- is_modified
 - seq64::event_list, 213
 - seq64::perform, 701, 773
 - seq64::seqmenu, 990, 991
- is_more_input
 - seq64::mastermidibase, 470
- is_mseq_valid
 - seq64::perform, 776
- is_no_modifier
 - seq64, 94
- is_note
 - seq64::event, 201
- is_note_msg
 - seq64::event, 189
- is_note_off
 - seq64::event, 200
- is_note_off_recorded
 - seq64::event, 201
- is_note_off_velocity
 - seq64, 76
- is_note_on
 - seq64::event, 200

- is_null_midipulse
 - seq64, 87
- is_one_byte
 - seq64::event, 201
- is_one_byte_msg
 - seq64::event, 188
- is_output_port
 - seq64::midibase, 596
- is_painted
 - seq64::event, 198
- is_pattern_playing
 - seq64::perform, 706
- is_port_open
 - seq64::midi_api, 503
 - seq64::rtmidi, 875
- is_press
 - seq64::click, 145
 - seq64::keystroke, 392
- is_realized
 - seq64::gui_window_gtk2, 318
- is_right
 - seq64::click, 145
- is_running
 - seq64::jack_assistant, 327
 - seq64::perform, 706, 775
- is_save
 - seq64::mainwnd, 447
- is_screenset_valid
 - seq64::perform, 774
- is_selected
 - seq64::event, 199
- is_seq_valid
 - seq64::perform, 775
- is_sequence_in_edit
 - seq64::perform, 716
- is_shift_key
 - seq64, 94, 95
- is_smf_0
 - seq64::perform, 740
 - seq64::sequence, 1073
- is_strict_note_msg
 - seq64::event, 189
- is_super_key
 - seq64, 96
- is_sysex
 - seq64::event, 202
 - seq64::midi_message, 563
- is_sysex_msg
 - seq64::event, 189
- is_sysex_special_id
 - seq64::midifile, 637
- is_system_port
 - seq64::busarray, 134
 - seq64::midi_api, 500
 - seq64::midibase, 596
- is_tempo
 - seq64::event, 202
- is_time_signature
 - seq64::event, 203
- is_two_byte_msg
 - seq64::event, 188
- is_two_bytes
 - seq64::event, 202
- is_valid
 - seq64::user_instrument, 1150
 - seq64::user_midi_bus, 1155
- is_valid_iterator
 - seq64::editable_events, 178
- is_variset
 - seq64::user_settings, 1175
- is_virtual_port
 - seq64::midi_api, 499
 - seq64::midibase, 595, 596
- iterator
 - seq64::editable_events, 172
 - seq64::event_list, 210
 - seq64::seqmenu, 989
- jack_assistant
 - seq64::jack_assistant, 324
 - seq64::perform, 781
- jack_data
 - seq64::midi_jack, 540
- jack_dialog
 - seq64::mainwnd, 445
- jack_dummy_callback
 - seq64, 91
- jack_frame_rate
 - seq64::jack_assistant, 333
- jack_idle_connect
 - seq64::gui_assistant, 277
 - seq64::gui_assistant_gtk2, 280
- jack_process_io
 - seq64::midi_jack_info, 553
- jack_session_callback
 - seq64, 85
 - seq64::jack_assistant, 340
- jack_session_uuid
 - seq64::rc_settings, 846, 852
- jack_shutdown
 - seq64::perform, 784
- jack_shutdown_callback
 - seq64, 81
 - seq64::jack_assistant, 338
- jack_sync_callback
 - seq64::perform, 783
- jack_timebase_callback
 - seq64, 81
 - seq64::jack_assistant, 338
 - seq64::perform, 784
- jack_transport_callback
 - seq64, 82
 - seq64::jack_assistant, 337
 - seq64::perform, 783
- jf_bit
 - seq64::jack_status_pair_t, 346
- jf_meaning

- seq64::jack_status_pair_t, 346
- js_clock_tick
 - seq64::jack_scratchpad, 344
- js_current_tick
 - seq64::jack_scratchpad, 344
- js_delta_tick_frac
 - seq64::jack_scratchpad, 346
- js_dumping
 - seq64::jack_scratchpad, 345
- js_init_clock
 - seq64::jack_scratchpad, 345
- js_jack_stopped
 - seq64::jack_scratchpad, 345
- js_looping
 - seq64::jack_scratchpad, 345
- js_playback_mode
 - seq64::jack_scratchpad, 345
- js_ticks_converted
 - seq64::jack_scratchpad, 345
- js_ticks_converted_last
 - seq64::jack_scratchpad, 345
- js_ticks_delta
 - seq64::jack_scratchpad, 345
- js_total_tick
 - seq64::jack_scratchpad, 344
- keep_queue
 - seq64::keys_perform, 359
- Key
 - seq64::editable_events, 171
- key
 - seq64::keystroke, 394
- key_height
 - seq64::user_settings, 1187, 1189
- key_name
 - seq64::keys_perform, 371
 - seq64::keys_perform_gtk2, 384
 - seq64::perform, 732
- key_press_event
 - seq64::perftime, 830
- keybindentry
 - seq64::keybindentry, 348
 - seq64::perform, 781
- keyboard_control_press
 - seq64::perform, 745
- keyboard_group_action
 - seq64::perform, 746
- keyboard_group_c_status_press
 - seq64::perform, 745
- keyboard_group_c_status_release
 - seq64::perform, 745
- keyboard_group_press
 - seq64::perform, 746
- keyboard_group_release
 - seq64::perform, 746
- keys
 - seq64::gui_assistant, 278
 - seq64::perform, 705
- keys_perform
 - seq64::keys_perform, 357
- keys_perform_gtk2
 - seq64::keys_perform_gtk2, 384
- keystroke
 - seq64::keystroke, 391, 392
- keyval_name
 - seq64, 97
- keyval_normalize
 - seq64, 86
- kpt_bpm_dn
 - seq64::keys_perform_transfer, 386
- kpt_bpm_up
 - seq64::keys_perform_transfer, 386
- kpt_event_edit
 - seq64::keys_perform_transfer, 388
- kpt_fast_forward
 - seq64::keys_perform_transfer, 389
- kpt_follow_transport
 - seq64::keys_perform_transfer, 389
- kpt_group_learn
 - seq64::keys_perform_transfer, 387
- kpt_group_off
 - seq64::keys_perform_transfer, 387
- kpt_group_on
 - seq64::keys_perform_transfer, 387
- kpt_keep_queue
 - seq64::keys_perform_transfer, 387
- kpt_menu_mode
 - seq64::keys_perform_transfer, 389
- kpt_oneshot_queue
 - seq64::keys_perform_transfer, 390
- kpt_pattern_edit
 - seq64::keys_perform_transfer, 388
- kpt_pattern_shift
 - seq64::keys_perform_transfer, 388
- kpt_pause
 - seq64::keys_perform_transfer, 389
- kpt_pointer_position
 - seq64::keys_perform_transfer, 389
- kpt_queue
 - seq64::keys_perform_transfer, 387
- kpt_replace
 - seq64::keys_perform_transfer, 387
- kpt_rewind
 - seq64::keys_perform_transfer, 389
- kpt_screenset_dn
 - seq64::keys_perform_transfer, 386
- kpt_screenset_up
 - seq64::keys_perform_transfer, 386
- kpt_set_playing_screenset
 - seq64::keys_perform_transfer, 386
- kpt_show_ui_sequence_key
 - seq64::keys_perform_transfer, 388
- kpt_show_ui_sequence_number
 - seq64::keys_perform_transfer, 388
- kpt_snapshot_1
 - seq64::keys_perform_transfer, 387
- kpt_snapshot_2

- seq64::keys_perform_transfer, 387
- kpt_song_mode
 - seq64::keys_perform_transfer, 389
- kpt_song_record
 - seq64::keys_perform_transfer, 390
- kpt_start
 - seq64::keys_perform_transfer, 388
- kpt_stop
 - seq64::keys_perform_transfer, 388
- kpt_tap_bpm
 - seq64::keys_perform_transfer, 388
- kpt_toggle_jack
 - seq64::keys_perform_transfer, 389
- kpt_toggle_mutes
 - seq64::keys_perform_transfer, 390
- lash
 - seq64::lash, 397
- lash_driver
 - seq64, 87
- lash_support
 - seq64::rc_settings, 842, 849
- lash_support_callback
 - seq64::options, 647
- lash_timeout_connect
 - seq64::gui_assistant, 278
 - seq64::gui_assistant_gtk2, 280
- last_used_dir
 - seq64::rc_settings, 846
- launch
 - seq64::perform, 714
- launch_input_thread
 - seq64::perform, 772
- launch_output_thread
 - seq64::perform, 772
- learn_toggle
 - seq64::mainwnd, 441
 - seq64::perform, 738
- left_right_size
 - seq64::perform, 720
- legacy_format
 - seq64::rc_settings, 842, 849
- length
 - seq64::trigger, 1125
- lfownd
 - seq64::lfownd, 402
 - seq64::seqdata, 918
- light_grey_paint
 - seq64::gui_palette_gtk2, 309
- line_after
 - seq64::configfile, 152
- line_color
 - seq64::gui_palette_gtk2, 307
- line_count
 - seq64::eventslots, 243
- line_increment
 - seq64::eventslots, 243
- line_maximum
 - seq64::eventslots, 243
- link
 - seq64::event, 197
- link_new
 - seq64::event_list, 216
 - seq64::sequence, 1101
- link_tempos
 - seq64::event_list, 216
 - seq64::sequence, 1101
- List
 - seq64::triggers, 1131
- load_events
 - seq64::editable_events, 175
 - seq64::eventslots, 243
- load_inverse_palette
 - seq64::gui_palette_gtk2, 307
- load_mute_group
 - seq64::perform, 768
- load_recent_file
 - seq64::mainwnd, 443
- lock
 - seq64::mutex, 641
- log2_time_sig_value
 - seq64, 59
- log_current_tempo
 - seq64::perform, 779
- log_main_sequence
 - seq64::midi_splitter, 577
- log_screenset
 - seq64::mainwid, 414
- log_tempo_track_number
 - seq64::options, 645
- long_options
 - seq64, 120
- lookup_keyevent_key
 - seq64::keys_perform, 369
 - seq64::perform, 733
- lookup_keyevent_seq
 - seq64::keys_perform, 370
 - seq64::perform, 734
- lookup_keygroup_group
 - seq64::keys_perform, 370
 - seq64::perform, 735
- lookup_keygroup_key
 - seq64::keys_perform, 370
 - seq64::perform, 735
- lookup_slot_key
 - seq64::perform, 734
- loop_record_t
 - seq64, 50
- m_32nds_per_quarter
 - seq64::perform, 793
 - seq64::sequence, 1122
- m_4bar_offset
 - seq64::perfrill, 820
 - seq64::perftime, 831
- m_FF_RW_button_type
 - seq64::perform, 787
- m_active

- seq64::businfo, [142](#)
- seq64::midi_control, [520](#)
- m_adding
 - seq64::perfroll, [818](#)
 - seq64::seqevent, [972](#)
 - seq64::seqroll, [1028](#)
- m_adding_pressed
 - seq64::perfroll, [818](#)
- m_adjust_bpm
 - seq64::mainwnd, [456](#)
- m_adjust_load_offset
 - seq64::mainwnd, [456](#)
- m_adjust_ss
 - seq64::mainwnd, [453](#)
- m_allow_click_edit
 - seq64::rc_settings, [856](#)
- m_allow_mod4_mode
 - seq64::rc_settings, [856](#)
- m_allow_snap_split
 - seq64::rc_settings, [856](#)
- m_allow_two_perfedits
 - seq64::user_settings, [1196](#)
- m_alsa_seq
 - seq64::midi_alsa_info, [495](#)
- m_api_data
 - seq64::rtmidi_in_data, [886](#)
- m_app_client_name
 - seq64::rc_settings, [859](#)
- m_app_name
 - seq64::midi_info, [536](#)
- m_application_name
 - seq64::rc_settings, [859](#)
- m_armed_progress_color
 - seq64::mainwid, [422](#)
- m_armed_saved
 - seq64::perform, [787](#)
- m_armed_statuses
 - seq64::perform, [787](#)
- m_auto_option_save
 - seq64::rc_settings, [855](#)
- m_b_on_c_pixmap
 - seq64::font, [262](#)
- m_b_on_y_pixmap
 - seq64::font, [262](#)
- m_back
 - seq64::midi_queue, [574](#)
- m_background
 - seq64::gui_drawingarea_gtk2, [298](#)
- m_background_sequence
 - seq64::seqroll, [1031](#)
 - seq64::sequence, [1123](#)
- m_background_x
 - seq64::perfroll, [819](#)
- m_bar_width
 - seq64::maintime, [408](#)
- m_beat_length
 - seq64::perfroll, [820](#)
- m_beat_width
 - seq64::jack_assistant, [343](#)
 - seq64::maintime, [408](#)
 - seq64::midi_timing, [583](#)
 - seq64::perform, [792](#)
- m_beats
 - seq64::midi_measures, [561](#)
- m_beats_per_bar
 - seq64::perform, [792](#)
- m_beats_per_measure
 - seq64::jack_assistant, [343](#)
 - seq64::midi_timing, [583](#)
- m_beats_per_minute
 - seq64::jack_assistant, [343](#)
 - seq64::mastermidibase, [478](#)
 - seq64::midi_timing, [583](#)
- m_bg_color
 - seq64::gui_palette_gtk2, [315](#)
- m_bgsequence
 - seq64::seqedit, [947](#)
- m_black
 - seq64::gui_palette_gtk2, [311](#)
- m_black_pixmap
 - seq64::font, [261](#)
- m_blk_key_paint
 - seq64::gui_palette_gtk2, [314](#)
- m_blk_paint
 - seq64::gui_palette_gtk2, [314](#)
- m_blue
 - seq64::gui_palette_gtk2, [312](#)
- m_bottbox
 - seq64::eventedit, [234](#)
- m_bottom_iterator
 - seq64::eventslots, [256](#)
- m_box_height
 - seq64::maintime, [409](#)
- m_box_less_pill
 - seq64::maintime, [409](#)
- m_box_width
 - seq64::maintime, [408](#)
- m_bpm
 - seq64::midi_info, [536](#)
 - seq64::midibase, [608](#)
 - seq64::perfedit, [672](#)
 - seq64::perform, [792](#)
- m_bpm_page_increment
 - seq64::user_settings, [1200](#)
- m_bpm_precision
 - seq64::user_settings, [1200](#)
- m_bpm_step_increment
 - seq64::user_settings, [1200](#)
- m_bus
 - seq64::businfo, [142](#)
 - seq64::sequence, [1116](#)
- m_bus_announce
 - seq64::mastermidibase, [477](#)
- m_bus_container
 - seq64::midi_info, [536](#)
- m_bus_id

- seq64::midibase, 607
- m_bus_index
 - seq64::midibase, 607
- m_bus_name
 - seq64::midibase, 608
- m_button
 - seq64::click, 147
- m_button_bpm
 - seq64::perfedit, 671
 - seq64::seqedit, 955
- m_button_bus
 - seq64::seqedit, 953
- m_button_bw
 - seq64::perfedit, 671
 - seq64::seqedit, 956
- m_button_cancel
 - seq64::eventedit, 234
- m_button_channel
 - seq64::seqedit, 953
- m_button_chord
 - seq64::seqedit, 955
- m_button_collapse
 - seq64::perfedit, 670
- m_button_copy
 - seq64::perfedit, 670
- m_button_data
 - seq64::seqedit, 955
- m_button_del
 - seq64::eventedit, 234
- m_button_down
 - seq64::mainwid, 422
- m_button_expand
 - seq64::perfedit, 670
- m_button_follow
 - seq64::perfedit, 671
- m_button_grow
 - seq64::perfedit, 670
- m_button_ins
 - seq64::eventedit, 234
- m_button_jack
 - seq64::mainwnd, 455
 - seq64::perfedit, 670
- m_button_jack_connect
 - seq64::options, 649
- m_button_jack_disconnect
 - seq64::options, 649
- m_button_jack_master
 - seq64::options, 648
- m_button_jack_master_cond
 - seq64::options, 648
- m_button_jack_transport
 - seq64::options, 648
- m_button_key
 - seq64::seqedit, 954
- m_button_learn
 - seq64::mainwnd, 454
- m_button_length
 - seq64::seqedit, 954
- m_button_lfo
 - seq64::seqedit, 951
- m_button_loop
 - seq64::perfedit, 670
- m_button_minidata
 - seq64::seqedit, 955
- m_button_modify
 - seq64::eventedit, 234
- m_button_note_length
 - seq64::seqedit, 953
- m_button_ok
 - seq64::options, 648
- m_button_panic
 - seq64::mainwnd, 454
- m_button_perfedit
 - seq64::mainwnd, 455
- m_button_play
 - seq64::mainwnd, 454
 - seq64::perfedit, 669
- m_button_quantize
 - seq64::seqedit, 952
- m_button_queue
 - seq64::mainwnd, 456
- m_button_rec_type
 - seq64::seqedit, 956
- m_button_rec_vol
 - seq64::seqedit, 956
- m_button_redo
 - seq64::perfedit, 670
 - seq64::seqedit, 952
- m_button_save
 - seq64::eventedit, 234
- m_button_scale
 - seq64::seqedit, 954
- m_button_sequence
 - seq64::seqedit, 952
- m_button_snap
 - seq64::perfedit, 669
 - seq64::seqedit, 953
- m_button_song_record
 - seq64::mainwnd, 455
- m_button_song_snap
 - seq64::mainwnd, 455
- m_button_stop
 - seq64::mainwnd, 454
 - seq64::perfedit, 669
- m_button_tempo_log
 - seq64::mainwnd, 454
- m_button_tempo_record
 - seq64::mainwnd, 454
- m_button_time_type
 - seq64::mainwnd, 456
- m_button_tools
 - seq64::seqedit, 952
- m_button_undo
 - seq64::perfedit, 670
 - seq64::seqedit, 952
- m_button_xpose

- seq64::perfedit, 669
- m_button_zoom
 - seq64::seqedit, 954
- m_bw
 - seq64::perfedit, 672
- m_bytes
 - seq64::midi_message, 563
- m_c_on_b_pixmap
 - seq64::font, 262
- m_call_seq_edit
 - seq64::mainwnd, 457
- m_call_seq_eventedit
 - seq64::mainwnd, 457
- m_call_seq_shift
 - seq64::mainwnd, 457
- m_can_add
 - seq64::FruitySeqRollInput, 275
- m_category
 - seq64::editable_event, 168
- m_cc
 - seq64::seqdata, 919
 - seq64::seqevent, 973
 - seq64::seqroll, 1032
- m_cell_h
 - seq64::font, 260
- m_cell_w
 - seq64::font, 260
- m_channel
 - seq64::event, 204
- m_channel_count
 - seq64::user_midi_bus, 1157
- m_channel_match
 - seq64::sequence, 1116
- m_char_list
 - seq64::midi_list, 558
 - seq64::midifile, 638
- m_char_vector
 - seq64::midi_vector, 587
- m_char_w
 - seq64::eventslots, 255
 - seq64::perfnames, 680
- m_chord
 - seq64::seqedit, 946
 - seq64::seqroll, 1028
- m_client
 - seq64::lash, 399
- m_client_name
 - seq64::midi_in_jack, 526
 - seq64::midi_port_info::port_info_t, 833
- m_client_number
 - seq64::midi_port_info::port_info_t, 833
- m_clip_mask
 - seq64::font, 262
- m_clipboard
 - seq64::triggers, 1147
- m_clock_mod
 - seq64::midibase, 607
- m_clock_type
 - seq64::midibase, 607
- m_clocks_per_metronome
 - seq64::perform, 793
 - seq64::sequence, 1122
- m_comments_block
 - seq64::rc_settings, 855
 - seq64::user_settings, 1193
- m_cond
 - seq64::condition_var, 149
- m_condition_var
 - seq64::perform, 797
- m_config_directory
 - seq64::rc_settings, 859
- m_config_filename
 - seq64::rc_settings, 859
- m_config_filename_alt
 - seq64::rc_settings, 859
- m_connected
 - seq64::midi_api, 505
- m_container
 - seq64::busarray, 136
- m_continue_sysex
 - seq64::rtmidi_in_data, 887
- m_control_height
 - seq64::user_settings, 1194
- m_control_status
 - seq64::perform, 796
- m_controller_count
 - seq64::user_instrument, 1152
- m_current_event
 - seq64::editable_events, 179
- m_current_index
 - seq64::eventslots, 256
- m_current_iterator
 - seq64::eventslots, 257
- m_current_screenset
 - seq64::mainwnd, 453
- m_current_seq
 - seq64::seqmenu, 998
- m_current_tick
 - seq64::perform, 792
- m_current_x
 - seq64::gui_drawingarea_gtk2, 299
- m_current_y
 - seq64::gui_drawingarea_gtk2, 299
- m_current_zoom
 - seq64::user_settings, 1195
- m_cyan
 - seq64::gui_palette_gtk2, 312
- m_d
 - seq64::configfile, 154
- m_data
 - seq64::event, 204
 - seq64::midi_control, 520
 - seq64::midifile, 638
- m_dest_addr_client
 - seq64::midi_alsa, 490
- m_dest_addr_port

- seq64::midi_alsa, [490](#)
- m_device_ignore
 - seq64::rc_settings, [858](#)
- m_device_ignore_num
 - seq64::rc_settings, [858](#)
- m_dirty_edit
 - seq64::sequence, [1119](#)
- m_dirty_main
 - seq64::sequence, [1119](#)
- m_dirty_names
 - seq64::sequence, [1120](#)
- m_dirty_perf
 - seq64::sequence, [1119](#)
- m_disable_reported
 - seq64::midifile, [638](#)
- m_display_name
 - seq64::midibase, [608](#)
- m_divisions
 - seq64::midi_measures, [561](#)
- m_divs_per_beat
 - seq64::perroll, [819](#)
- m_dk_black
 - seq64::gui_palette_gtk2, [312](#)
- m_dk_blue
 - seq64::gui_palette_gtk2, [313](#)
- m_dk_cyan
 - seq64::gui_palette_gtk2, [313](#)
- m_dk_green
 - seq64::gui_palette_gtk2, [312](#)
- m_dk_grey
 - seq64::gui_palette_gtk2, [314](#)
- m_dk_grey_paint
 - seq64::gui_palette_gtk2, [314](#)
- m_dk_magenta
 - seq64::gui_palette_gtk2, [313](#)
- m_dk_orange
 - seq64::gui_palette_gtk2, [313](#)
- m_dk_pink
 - seq64::gui_palette_gtk2, [314](#)
- m_dk_red
 - seq64::gui_palette_gtk2, [312](#)
- m_dk_white
 - seq64::gui_palette_gtk2, [313](#)
- m_dk_yellow
 - seq64::gui_palette_gtk2, [312](#)
- m_do_input
 - seq64::rtmidi_in_data, [886](#)
- m_dont_reset_ticks
 - seq64::perform, [795](#)
- m_drag_handle
 - seq64::seqdata, [920](#)
- m_drag_paste_start_pos
 - seq64::FruitySeqRollInput, [275](#)
- m_dragging
 - seq64::seqdata, [920](#)
- m_drawing_background_seq
 - seq64::seqroll, [1031](#)
- m_drop_sequence
 - seq64::perroll, [821](#)
- m_drop_tick
 - seq64::perroll, [821](#)
- m_drop_tick_offset
 - seq64::perroll, [821](#)
- m_drop_x
 - seq64::gui_drawingarea_gtk2, [299](#)
- m_drop_y
 - seq64::gui_drawingarea_gtk2, [300](#)
- m_dumping_input
 - seq64::mastermidibase, [479](#)
- m_edit_sequence
 - seq64::perform, [797](#)
- m_editbox
 - seq64::eventedit, [233](#)
- m_editing
 - seq64::sequence, [1120](#)
- m_editing_cc
 - seq64::seqedit, [957](#)
- m_editing_status
 - seq64::seqedit, [957](#)
- m_effective_tick
 - seq64::Seq24PerfInput, [904](#)
- m_entry_bpm
 - seq64::perfedit, [671](#)
 - seq64::seqedit, [956](#)
- m_entry_bus
 - seq64::seqedit, [953](#)
- m_entry_bw
 - seq64::perfedit, [671](#)
 - seq64::seqedit, [956](#)
- m_entry_channel
 - seq64::seqedit, [953](#)
- m_entry_chord
 - seq64::seqedit, [955](#)
- m_entry_data
 - seq64::seqedit, [955](#)
- m_entry_ev_data_0
 - seq64::eventedit, [236](#)
- m_entry_ev_data_1
 - seq64::eventedit, [236](#)
- m_entry_ev_name
 - seq64::eventedit, [236](#)
- m_entry_ev_timestamp
 - seq64::eventedit, [236](#)
- m_entry_key
 - seq64::seqedit, [954](#)
- m_entry_length
 - seq64::seqedit, [954](#)
- m_entry_name
 - seq64::seqedit, [957](#)
- m_entry_note_length
 - seq64::seqedit, [954](#)
- m_entry_notes
 - seq64::mainwnd, [457](#)
- m_entry_scale
 - seq64::seqedit, [955](#)
- m_entry_seqnumber

- seq64::seqedit, 957
- m_entry_sequence
 - seq64::seqedit, 953
- m_entry_snap
 - seq64::perfedit, 669
 - seq64::seqedit, 953
- m_entry_xpose
 - seq64::perfedit, 669
- m_entry_zoom
 - seq64::seqedit, 954
- m_erase_painting
 - seq64::FruitySeqRollInput, 275
- m_error_callback
 - seq64::midi_api, 506
- m_error_callback_user_data
 - seq64::midi_api, 506
- m_error_is_fatal
 - seq64::midifile, 638
- m_error_message
 - seq64::configfile, 153
 - seq64::midifile, 637
- m_error_string
 - seq64::midi_api, 505
 - seq64::midi_info, 536
- m_event_container
 - seq64::eventslots, 254
- m_event_count
 - seq64::eventslots, 255
- m_eventedit
 - seq64::seqmenu, 998
- m_events
 - seq64::editable_events, 179
 - seq64::event_list, 220
 - seq64::sequence, 1115
- m_events_clipboard
 - seq64::sequence, 1114
- m_events_redo
 - seq64::sequence, 1116
- m_events_undo
 - seq64::sequence, 1115
- m_events_undo_hold
 - seq64::sequence, 1115
- m_eventslots
 - seq64::eventedit, 233
- m_excell_FF_RW
 - seq64::perform, 786
- m_expanded_recording
 - seq64::seqroll, 1031
- m_fg_color
 - seq64::gui_palette_gtk2, 315
- m_file_size
 - seq64::midifile, 637
- m_filename
 - seq64::rc_settings, 858
- m_filter_by_channel
 - seq64::mastermidibase, 479
 - seq64::perform, 793
 - seq64::rc_settings, 857
- m_first_error_occurred
 - seq64::midi_api, 506
- m_first_event
 - seq64::seqedit, 957
- m_first_event_name
 - seq64::seqedit, 958
- m_first_message
 - seq64::rtmidi_in_data, 886
- m_flash_height
 - seq64::maintime, 409
- m_flash_width
 - seq64::maintime, 409
- m_flash_x
 - seq64::maintime, 409
- m_follow_transport
 - seq64::jack_assistant, 343
- m_font_h
 - seq64::font, 261
- m_font_w
 - seq64::font, 261
- m_foreground
 - seq64::gui_drawingarea_gtk2, 299
- m_format_timestamp
 - seq64::editable_event, 168
- m_front
 - seq64::midi_queue, 574
- m_gc
 - seq64::gui_drawingarea_gtk2, 298
- m_global_bgsequence
 - seq64::midifile, 639
- m_global_queue
 - seq64::midi_info, 536
- m_global_seq_feature_save
 - seq64::user_settings, 1195
- m_gmute_tracks
 - seq64::user_settings, 1201
- m_green
 - seq64::gui_palette_gtk2, 311
- m_grey
 - seq64::gui_palette_gtk2, 313
- m_grey_paint
 - seq64::gui_palette_gtk2, 314
- m_grid_brackets
 - seq64::user_settings, 1193
- m_grid_style
 - seq64::user_settings, 1193
- m_group_max
 - seq64::keys_perform, 378
- m_grow_direction
 - seq64::perfroll, 822
- m_growing
 - seq64::perfroll, 822
 - seq64::seqevent, 972
 - seq64::seqroll, 1029
- m_gui_support
 - seq64::perform, 798
- m_h_page_increment
 - seq64::perfroll, 818

- m_h_perf_page_increment
 - seq64::user_settings, [1196](#)
- m_hadjust
 - seq64::gui_drawingarea_gtk2, [298](#)
 - seq64::perfedit, [668](#)
 - seq64::seqedit, [950](#)
- m_has_link
 - seq64::event, [204](#)
- m_has_tempo
 - seq64::event_list, [220](#)
- m_has_time_signature
 - seq64::event_list, [220](#)
- m_have_button_press
 - seq64::perfroll, [820](#)
- m_have_focus
 - seq64::eventedit, [237](#)
 - seq64::seqedit, [958](#)
- m_have_redo
 - seq64::perform, [798](#)
 - seq64::sequence, [1115](#)
- m_have_undo
 - seq64::perform, [798](#)
 - seq64::sequence, [1115](#)
- m_hbox
 - seq64::lfownd, [403](#)
 - seq64::perfedit, [671](#)
 - seq64::seqedit, [952](#)
- m_hbox2
 - seq64::seqedit, [952](#)
- m_height
 - seq64::rect, [867](#)
- m_hint_key
 - seq64::seqkeys, [984](#)
- m_hint_state
 - seq64::seqkeys, [984](#)
- m_hlbox
 - seq64::perfedit, [671](#)
- m_horizontal_adjust
 - seq64::seqroll, [1027](#)
- m_hscroll
 - seq64::perfedit, [668](#)
- m_hscroll_new
 - seq64::seqedit, [950](#)
- m_htopbox
 - seq64::eventedit, [233](#)
- m_ignore_flags
 - seq64::rtmidi_in_data, [886](#)
- m_image_mousemode
 - seq64::seqedit, [957](#)
- m_image_play
 - seq64::mainwnd, [454](#)
 - seq64::perfedit, [669](#)
- m_image_transpose
 - seq64::seqedit, [948](#)
- m_in_thread
 - seq64::perform, [790](#)
- m_in_thread_launched
 - seq64::perform, [790](#)
- m_inbus_array
 - seq64::mastermidibase, [478](#)
- m_info_api
 - seq64::rtmidi_info, [896](#)
- m_init_clock
 - seq64::businfo, [142](#)
- m_init_input
 - seq64::businfo, [142](#)
- m_initial_chord
 - seq64::seqedit, [946](#)
- m_initial_note_length
 - seq64::seqedit, [945](#)
- m_initial_snap
 - seq64::seqedit, [945](#)
- m_initial_zoom
 - seq64::seqedit, [946](#)
- m_initialized
 - seq64::businfo, [142](#)
- m_input
 - seq64::midi_info, [535](#)
- m_input_data
 - seq64::midi_api, [505](#)
- m_input_port_name
 - seq64::midi_alsa, [491](#)
- m_inputting
 - seq64::midibase, [608](#)
 - seq64::perform, [791](#)
- m_instrument_def
 - seq64::user_instrument, [1152](#)
- m_instruments
 - seq64::user_settings, [1193](#)
- m_interaction_method
 - seq64::rc_settings, [858](#)
- m_inverse_active
 - seq64::midi_control, [520](#)
- m_inverse_colors
 - seq64::user_settings, [1196](#)
- m_is_drag_pasting
 - seq64::FruitySeqEventInput, [271](#)
 - seq64::seqroll, [1029](#)
- m_is_drag_pasting_start
 - seq64::FruitySeqEventInput, [271](#)
 - seq64::seqroll, [1029](#)
- m_is_input
 - seq64::midi_port_info::port_info_t, [834](#)
- m_is_input_port
 - seq64::midibase, [609](#)
- m_is_inverse
 - seq64::gui_palette_gtk2, [311](#)
- m_is_lash_supported
 - seq64::lash, [399](#)
- m_is_modified
 - seq64::event_list, [220](#)
 - seq64::perform, [797](#)
- m_is_pattern_playing
 - seq64::perform, [791](#)
- m_is_press
 - seq64::click, [146](#)

- seq64::keystroke, [396](#)
- m_is_realized
 - seq64::gui_window_gtk2, [320](#)
- m_is_running
 - seq64::mainwnd, [457](#)
 - seq64::perfedit, [672](#)
 - seq64::perform, [791](#)
- m_is_snap_recording
 - seq64::mainwnd, [455](#)
- m_is_song_recording
 - seq64::mainwnd, [455](#)
- m_is_system
 - seq64::midi_port_info::port_info_t, [834](#)
- m_is_system_port
 - seq64::midibase, [609](#)
- m_is_tempo_recording
 - seq64::mainwnd, [455](#)
- m_is_valid
 - seq64::user_instrument, [1152](#)
 - seq64::user_midi_bus, [1157](#)
- m_is_virtual
 - seq64::midi_port_info::port_info_t, [834](#)
- m_is_virtual_port
 - seq64::midibase, [609](#)
- m_iterator_draw
 - seq64::sequence, [1116](#)
- m_iterator_draw_trigger
 - seq64::triggers, [1147](#)
- m_iterator_play_trigger
 - seq64::triggers, [1147](#)
- m_jack_asst
 - seq64::perform, [798](#)
- m_jack_buffmessage
 - seq64::midi_jack_data, [547](#)
- m_jack_buffsize
 - seq64::midi_jack_data, [547](#)
- m_jack_client
 - seq64::jack_assistant, [341](#)
 - seq64::midi_jack_data, [547](#)
 - seq64::midi_jack_info, [553](#)
- m_jack_client_2
 - seq64::midi_jack_info, [554](#)
- m_jack_client_name
 - seq64::jack_assistant, [341](#)
- m_jack_client_uuid
 - seq64::jack_assistant, [341](#)
- m_jack_data
 - seq64::midi_jack, [545](#)
- m_jack_frame_current
 - seq64::jack_assistant, [341](#)
- m_jack_frame_last
 - seq64::jack_assistant, [341](#)
- m_jack_frame_rate
 - seq64::jack_assistant, [342](#)
- m_jack_info
 - seq64::midi_jack, [545](#)
- m_jack_lasttime
 - seq64::midi_jack_data, [548](#)
- m_jack_master
 - seq64::jack_assistant, [342](#)
- m_jack_parent
 - seq64::jack_assistant, [341](#)
- m_jack_port
 - seq64::midi_jack_data, [547](#)
- m_jack_ports
 - seq64::midi_jack_info, [553](#)
- m_jack_pos
 - seq64::jack_assistant, [341](#)
- m_jack_rtmidiin
 - seq64::midi_jack_data, [548](#)
- m_jack_running
 - seq64::jack_assistant, [342](#)
- m_jack_session_uuid
 - seq64::rc_settings, [858](#)
- m_jack_stop_tick
 - seq64::jack_assistant, [343](#)
- m_jack_tick
 - seq64::jack_assistant, [342](#)
 - seq64::perform, [794](#)
- m_jack_transport_state
 - seq64::jack_assistant, [341](#)
- m_jack_transport_state_last
 - seq64::jack_assistant, [342](#)
- m_session_ev
 - seq64::jack_assistant, [342](#)
- m_justselected_one
 - seq64::FruitySeqEventInput, [271](#)
 - seq64::seqroll, [1030](#)
- m_key
 - seq64::keybindentry, [349](#)
 - seq64::keystroke, [396](#)
 - seq64::seqedit, [946](#)
 - seq64::seqkeys, [984](#)
 - seq64::seqroll, [1028](#)
- m_key_bpm_dn
 - seq64::keys_perform, [378](#)
- m_key_bpm_up
 - seq64::keys_perform, [378](#)
- m_key_event_edit
 - seq64::keys_perform, [381](#)
- m_key_events
 - seq64::keys_perform, [377](#)
- m_key_events_rev
 - seq64::keys_perform, [378](#)
- m_key_fast_forward
 - seq64::keys_perform, [381](#)
- m_key_follow_transport
 - seq64::keys_perform, [380](#)
- m_key_group_learn
 - seq64::keys_perform, [380](#)
- m_key_group_off
 - seq64::keys_perform, [379](#)
- m_key_group_on
 - seq64::keys_perform, [379](#)
- m_key_groups
 - seq64::keys_perform, [377](#)

`m_key_groups_rev`
 `seq64::keys_perform`, 378

`m_key_keep_queue`
 `seq64::keys_perform`, 379

`m_key_menu_mode`
 `seq64::keys_perform`, 380

`m_key_oneshot_queue`
 `seq64::keys_perform`, 382

`m_key_pattern_edit`
 `seq64::keys_perform`, 381

`m_key_pattern_shift`
 `seq64::keys_perform`, 381

`m_key_pause`
 `seq64::keys_perform`, 380

`m_key_pointer_position`
 `seq64::keys_perform`, 381

`m_key_queue`
 `seq64::keys_perform`, 378

`m_key_replace`
 `seq64::keys_perform`, 378

`m_key_rewind`
 `seq64::keys_perform`, 380

`m_key_screenset_dn`
 `seq64::keys_perform`, 379

`m_key_screenset_up`
 `seq64::keys_perform`, 379

`m_key_set_playing_screenset`
 `seq64::keys_perform`, 379

`m_key_show_ui_sequence_key`
 `seq64::keys_perform`, 377

`m_key_show_ui_sequence_number`
 `seq64::keys_perform`, 377

`m_key_snapshot_1`
 `seq64::keys_perform`, 379

`m_key_snapshot_2`
 `seq64::keys_perform`, 379

`m_key_song_mode`
 `seq64::keys_perform`, 380

`m_key_song_record`
 `seq64::keys_perform`, 382

`m_key_start`
 `seq64::keys_perform`, 380

`m_key_stop`
 `seq64::keys_perform`, 381

`m_key_tap_bpm`
 `seq64::keys_perform`, 381

`m_key_toggle_jack`
 `seq64::keys_perform`, 380

`m_key_toggle_mutes`
 `seq64::keys_perform`, 381

`m_keying`
 `seq64::seqkeys`, 984

`m_keying_note`
 `seq64::seqkeys`, 984

`m_keys_perform`
 `seq64::gui_assistant`, 278

`m_label_category`
 `seq64::eventedit`, 236

`m_label_channel`
 `seq64::eventedit`, 235

`m_label_ev_count`
 `seq64::eventedit`, 235

`m_label_modified`
 `seq64::eventedit`, 235

`m_label_ppqn`
 `seq64::eventedit`, 235

`m_label_right`
 `seq64::eventedit`, 236

`m_label_seq_length`
 `seq64::eventedit`, 235

`m_label_seq_name`
 `seq64::eventedit`, 235

`m_label_spacer`
 `seq64::eventedit`, 235

`m_label_time_fmt`
 `seq64::eventedit`, 236

`m_label_time_sig`
 `seq64::eventedit`, 235

`m_lash_args`
 `seq64::lash`, 399

`m_lash_support`
 `seq64::rc_settings`, 856

`m_last_max_timestamp`
 `seq64::eventslots`, 255

`m_last_tick`
 `seq64::sequence`, 1120

`m_last_tick_x`
 `seq64::mainwid`, 423

`m_last_used_dir`
 `seq64::rc_settings`, 858

`m_lasttick`
 `seq64::midibase`, 609

`m_left_marker_tick`
 `seq64::perftime`, 832

`m_left_tick`
 `seq64::perform`, 794

`m_legacy_format`
 `seq64::rc_settings`, 856

`m_length`
 `seq64::sequence`, 1121
 `seq64::triggers`, 1148

`m_lfo_wnd`
 `seq64::seqedit`, 951

`m_line`
 `seq64::configfile`, 154

`m_line_color`
 `seq64::gui_palette_gtk2`, 315

`m_line_count`
 `seq64::eventslots`, 256

`m_line_maximum`
 `seq64::eventslots`, 256

`m_line_overlap`
 `seq64::eventslots`, 256

`m_linked`
 `seq64::event`, 204

`m_local_addr_client`

- seq64::midi_alsa, 490
- m_local_addr_port
 - seq64::midi_alsa, 490
- m_loop_reset
 - seq64::sequence, 1119
- m_looping
 - seq64::perform, 791
- m_lt_grey_paint
 - seq64::gui_palette_gtk2, 314
- m_magenta
 - seq64::gui_palette_gtk2, 312
- m_main_cursor
 - seq64::mainwnd, 454
- m_main_time
 - seq64::mainwnd, 453
- m_main_wid
 - seq64::mainwnd, 452
- m_mainperf
 - seq64::gui_drawingarea_gtk2, 299
 - seq64::gui_window_gtk2, 320
 - seq64::options, 648
 - seq64::seqmenu, 997
- m_mainwid_adjustors
 - seq64::mainwnd, 451
- m_mainwid_block_cols
 - seq64::user_settings, 1197
- m_mainwid_block_independent
 - seq64::user_settings, 1197
- m_mainwid_block_rows
 - seq64::user_settings, 1197
- m_mainwid_blocks
 - seq64::mainwnd, 452
- m_mainwid_border
 - seq64::user_settings, 1194
- m_mainwid_border_x
 - seq64::mainwid, 424
- m_mainwid_border_y
 - seq64::mainwid, 424
- m_mainwid_columns
 - seq64::mainwnd, 452
- m_mainwid_count
 - seq64::mainwnd, 452
- m_mainwid_frames
 - seq64::mainwnd, 451
- m_mainwid_grid
 - seq64::mainwnd, 451
- m_mainwid_independent
 - seq64::mainwnd, 452
- m_mainwid_rows
 - seq64::mainwnd, 452
- m_mainwid_spacing
 - seq64::mainwid, 424
 - seq64::user_settings, 1194
- m_mainwid_spinners
 - seq64::mainwnd, 452
- m_mainwid_x
 - seq64::mainwid, 423
 - seq64::user_settings, 1202
- m_mainwid_y
 - seq64::mainwid, 424
 - seq64::user_settings, 1202
- m_mainwnd_cols
 - seq64::mainwid, 423
 - seq64::user_settings, 1194
- m_mainwnd_rows
 - seq64::mainwid, 423
 - seq64::user_settings, 1194
- m_mainwnd_x
 - seq64::user_settings, 1202
- m_mainwnd_y
 - seq64::user_settings, 1202
- m_manual_alsa_ports
 - seq64::rc_settings, 857
- m_marked
 - seq64::event, 205
- m_master_bus
 - seq64::perform, 793
 - seq64::sequence, 1117
- m_master_clocks
 - seq64::mastermidibase, 478
 - seq64::perform, 793
- m_master_info
 - seq64::midi_api, 505
 - seq64::midibus, 614
- m_master_inputs
 - seq64::mastermidibase, 478
 - seq64::perform, 794
- m_max_busses
 - seq64::mastermidibase, 477
- m_max_groups
 - seq64::perform, 787
- m_max_sequence
 - seq64::user_settings, 1201
- m_max_sets
 - seq64::mainwid, 424
 - seq64::perform, 796
 - seq64::user_settings, 1194
- m_max_value
 - seq64::midi_control, 521
- m_maxbeats
 - seq64::sequence, 1121
- m_measure_length
 - seq64::perfroll, 820
 - seq64::perftime, 832
- m_measures
 - seq64::eventslots, 255
 - seq64::midi_measures, 560
 - seq64::seqedit, 947
- m_menu
 - seq64::seqmenu, 997
- m_menu_bpm
 - seq64::perfeddit, 671
 - seq64::seqedit, 950
- m_menu_bw
 - seq64::perfeddit, 672
 - seq64::seqedit, 950

- m_menu_chords
 - seq64::seqedit, [949](#)
- m_menu_data
 - seq64::seqedit, [949](#)
- m_menu_edit
 - seq64::mainwnd, [451](#)
- m_menu_file
 - seq64::mainwnd, [450](#)
- m_menu_help
 - seq64::mainwnd, [451](#)
- m_menu_key
 - seq64::seqedit, [949](#)
- m_menu_length
 - seq64::seqedit, [948](#)
- m_menu_midibus
 - seq64::seqedit, [949](#)
- m_menu_midich
 - seq64::seqedit, [949](#)
- m_menu_minidata
 - seq64::seqedit, [949](#)
- m_menu_mode
 - seq64::mainwnd, [457](#)
- m_menu_note_length
 - seq64::seqedit, [948](#)
- m_menu_rec_type
 - seq64::seqedit, [950](#)
- m_menu_rec_vol
 - seq64::seqedit, [950](#)
- m_menu_recent
 - seq64::mainwnd, [450](#)
- m_menu_scale
 - seq64::seqedit, [949](#)
- m_menu_sequences
 - seq64::seqedit, [949](#)
- m_menu_snap
 - seq64::perfedit, [668](#)
 - seq64::seqedit, [948](#)
- m_menu_tools
 - seq64::seqedit, [948](#)
- m_menu_view
 - seq64::mainwnd, [451](#)
- m_menu_xpose
 - seq64::perfedit, [669](#)
- m_menu_zoom
 - seq64::seqedit, [948](#)
- m_menubar
 - seq64::mainwnd, [450](#)
 - seq64::seqedit, [947](#)
- m_message
 - seq64::rterror, [869](#)
 - seq64::rtmidi_in_data, [886](#)
- m_midi_api
 - seq64::rtmidi, [877](#)
- m_midi_beat_width
 - seq64::user_settings, [1199](#)
- m_midi_beats_per_measure
 - seq64::user_settings, [1198](#)
- m_midi_beats_per_minute
 - seq64::user_settings, [1199](#)
- m_midi_bpm_maximum
 - seq64::user_settings, [1199](#)
- m_midi_bpm_minimum
 - seq64::user_settings, [1199](#)
- m_midi_bus_def
 - seq64::user_midi_bus, [1158](#)
- m_midi_buses
 - seq64::user_settings, [1193](#)
- m_midi_buss_override
 - seq64::user_settings, [1199](#)
- m_midi_cc_off
 - seq64::perform, [796](#)
- m_midi_cc_on
 - seq64::perform, [795](#)
- m_midi_cc_toggle
 - seq64::perform, [795](#)
- m_midi_channel
 - seq64::sequence, [1116](#)
- m_midi_handle
 - seq64::midi_info, [536](#)
- m_midi_info
 - seq64::rtmidi, [877](#)
- m_midi_master
 - seq64::mastermidibus, [484](#)
- m_midi_mode_input
 - seq64::midi_info, [535](#)
- m_midi_mute_group_present
 - seq64::perform, [788](#)
- m_midi_parameters
 - seq64::editable_events, [179](#)
- m_midi_ppqn
 - seq64::user_settings, [1198](#)
- m_midiclockpos
 - seq64::perform, [795](#)
- m_midiclockrunning
 - seq64::perform, [795](#)
- m_midiclocktick
 - seq64::perform, [795](#)
- m_min_value
 - seq64::midi_control, [520](#)
- m_mode_group
 - seq64::perform, [788](#)
- m_mode_group_learn
 - seq64::perform, [788](#)
- m_modified
 - seq64::seqmenu, [998](#)
- m_modifier
 - seq64::click, [147](#)
 - seq64::keystroke, [396](#)
- m_move_delta_x
 - seq64::seqroll, [1030](#)
- m_move_delta_y
 - seq64::seqroll, [1030](#)
- m_move_snap_offset_x
 - seq64::seqevent, [973](#)
 - seq64::seqroll, [1030](#)
- m_moving

- seq64::mainwid, [422](#)
- seq64::perfroll, [822](#)
- seq64::seqevent, [972](#)
- seq64::seqroll, [1029](#)
- m_moving_init
 - seq64::seqevent, [972](#)
 - seq64::seqroll, [1029](#)
- m_moving_seq
 - seq64::mainwid, [422](#)
- m_musical_key
 - seq64::sequence, [1123](#)
- m_musical_scale
 - seq64::sequence, [1123](#)
- m_mute_group
 - seq64::perform, [787](#)
- m_mute_group_rc
 - seq64::perform, [787](#)
- m_mute_group_saving
 - seq64::rc_settings, [858](#)
- m_mute_group_selected
 - seq64::perform, [788](#)
- m_mutex
 - seq64::mastermidibase, [479](#)
 - seq64::midibase, [609](#)
 - seq64::midifile, [637](#)
 - seq64::sequence, [1123](#)
- m_mutex_lock
 - seq64::mutex, [641](#)
- m_name
 - seq64::configfile, [153](#)
 - seq64::midifile, [638](#)
 - seq64::sequence, [1120](#)
- m_name_category
 - seq64::editable_event, [168](#)
- m_name_channel
 - seq64::editable_event, [169](#)
- m_name_data
 - seq64::editable_event, [169](#)
- m_name_meta
 - seq64::editable_event, [169](#)
- m_name_seqspect
 - seq64::editable_event, [169](#)
- m_name_status
 - seq64::editable_event, [168](#)
- m_name_timestamp
 - seq64::editable_event, [168](#)
- m_namebox_w
 - seq64::perfnames, [680](#)
- m_names_chars
 - seq64::perfnames, [680](#)
- m_names_x
 - seq64::perfnames, [680](#)
- m_names_y
 - seq64::perfnames, [680](#)
 - seq64::perfroll, [819](#)
- m_new_format
 - seq64::midifile, [638](#)
- m_note_length
 - seq64::seqedit, [946](#)
 - seq64::seqroll, [1028](#)
- m_note_off_margin
 - seq64::sequence, [1123](#)
- m_note_off_velocity
 - seq64::sequence, [1123](#)
- m_note_on_velocity
 - seq64::sequence, [1122](#)
- m_notebook
 - seq64::options, [649](#)
- m_notes_on
 - seq64::sequence, [1117](#)
- m_notify
 - seq64::perform, [798](#)
- m_num_poll_descriptors
 - seq64::midi_alsa_info, [495](#)
- m_number_h
 - seq64::seqdata, [919](#)
- m_number_offset_y
 - seq64::seqdata, [919](#)
- m_number_selected
 - seq64::triggers, [1147](#)
- m_number_w
 - seq64::seqdata, [919](#)
- m_off_from_snap
 - seq64::sequence, [1118](#)
- m_offset
 - seq64::font, [261](#)
 - seq64::trigger, [1128](#)
- m_old
 - seq64::seqdata, [920](#)
 - seq64::seqevent, [971](#)
 - seq64::seqroll, [1027](#)
- m_old_progress_ticks
 - seq64::perfroll, [820](#)
- m_old_seq
 - seq64::mainwid, [422](#)
- m_one_measure
 - seq64::perform, [794](#)
- m_one_shot
 - seq64::sequence, [1118](#)
- m_one_shot_tick
 - seq64::sequence, [1118](#)
- m_options
 - seq64::mainwnd, [453](#)
- m_optsbox
 - seq64::eventedit, [234](#)
- m_orange
 - seq64::gui_palette_gtk2, [313](#)
- m_out_thread
 - seq64::perform, [790](#)
- m_out_thread_launched
 - seq64::perform, [790](#)
- m_outbus_array
 - seq64::mastermidibase, [478](#)
- m_output
 - seq64::midi_info, [536](#)
- m_outputing

- seq64::perform, 791
- m_overwrite_recording
 - seq64::sequence, 1119
- m_padded_h
 - seq64::font, 261
- m_page_factor
 - seq64::perfroll, 819
- m_pager_index
 - seq64::eventslots, 257
- m_painted
 - seq64::event, 205
- m_painting
 - seq64::seqevent, 972
 - seq64::seqroll, 1029
- m_palette
 - seq64::gui_palette_gtk2, 311
- m_parent
 - seq64::editable_event, 168
 - seq64::eventslots, 254
 - seq64::perfnames, 679
 - seq64::perfroll, 817
 - seq64::perftime, 831
 - seq64::sequence, 1115
 - seq64::triggers, 1146
- m_parent_bus
 - seq64::midi_api, 505
- m_pass_sysex
 - seq64::rc_settings, 857
- m_paste
 - seq64::seqevent, 973
 - seq64::seqroll, 1029
- m_paste_tick
 - seq64::triggers, 1148
- m_peer_perfedit
 - seq64::perfedit, 667
- m_perf
 - seq64::keybindentry, 349
- m_perf_edit
 - seq64::mainwnd, 453
- m_perf_edit_2
 - seq64::mainwnd, 453
- m_perf_scale_x
 - seq64::perfroll, 819
 - seq64::perftime, 832
- m_perfnames
 - seq64::perfedit, 668
- m_perform
 - seq64::lash, 399
- m_perfroll
 - seq64::perfedit, 668
- m_perftime
 - seq64::perfedit, 668
- m_phase
 - seq64::lfownd, 404
- m_pill_width
 - seq64::maintime, 408
- m_pink
 - seq64::gui_palette_gtk2, 313
- m_pixmap
 - seq64::font, 261
 - seq64::gui_drawingarea_gtk2, 298
- m_playback_mode
 - seq64::perform, 792
- m_playing
 - seq64::sequence, 1117
- m_playing_notes
 - seq64::sequence, 1117
- m_playscreen
 - seq64::perform, 796
- m_playscreen_offset
 - seq64::perform, 796
- m_poll_descriptors
 - seq64::midi_alsa_info, 496
- m_port_container
 - seq64::midi_port_info, 571
- m_port_count
 - seq64::midi_port_info, 571
- m_port_id
 - seq64::midibase, 607
- m_port_name
 - seq64::midi_port_info::port_info_t, 833
 - seq64::midibase, 608
- m_port_number
 - seq64::midi_port_info::port_info_t, 833
- m_pos
 - seq64::midifile, 638
 - seq64::seqroll, 1028
- m_position_for_get
 - seq64::midi_container, 516
- m_pp_eighth
 - seq64::seqedit, 947
- m_pp_sixteenth
 - seq64::seqedit, 947
- m_pp_whole
 - seq64::seqedit, 947
- m_ppqn
 - seq64::jack_assistant, 343
 - seq64::maintime, 409
 - seq64::mainwnd, 451
 - seq64::mastermidibase, 478
 - seq64::midi_info, 536
 - seq64::midi_splitter, 579
 - seq64::midi_timing, 583
 - seq64::midibase, 608
 - seq64::midifile, 639
 - seq64::perfedit, 672
 - seq64::perform, 792
 - seq64::perfroll, 818
 - seq64::perftime, 831
 - seq64::seqedit, 947
 - seq64::sequence, 1121
 - seq64::triggers, 1148
- m_print_keys
 - seq64::rc_settings, 858
- m_priority
 - seq64::rc_settings, 856

- m_progress_bar_colored
 - seq64::user_settings, [1196](#)
- m_progress_bar_thick
 - seq64::user_settings, [1196](#)
- m_progress_color
 - seq64::gui_palette_gtk2, [315](#)
- m_progress_follow
 - seq64::seqroll, [1031](#)
- m_progress_height
 - seq64::mainwid, [425](#)
- m_progress_x
 - seq64::seqroll, [1030](#)
- m_quantized_rec
 - seq64::sequence, [1117](#)
- m_queue
 - seq64::mastermidibase, [478](#)
 - seq64::midibase, [608](#)
 - seq64::rtmidi_in_data, [886](#)
- m_queue_number
 - seq64::midi_port_info::port_info_t, [834](#)
- m_queued
 - seq64::sequence, [1118](#)
- m_queued_replace_slot
 - seq64::perform, [790](#)
- m_queued_tick
 - seq64::sequence, [1120](#)
- m_raise
 - seq64::sequence, [1120](#)
- m_range
 - seq64::lfownd, [404](#)
- m_rank
 - seq64::event_list::event_key, [207](#)
- m_rec_vol
 - seq64::sequence, [1122](#)
- m_recent_files
 - seq64::rc_settings, [860](#)
- m_recording
 - seq64::sequence, [1117](#)
- m_red
 - seq64::gui_palette_gtk2, [311](#)
- m_redo_stack
 - seq64::triggers, [1147](#)
- m_redo_vect
 - seq64::perform, [798](#)
- m_redraw_period_ms
 - seq64::gui_window_gtk2, [320](#)
- m_remote_port_name
 - seq64::midi_jack, [545](#)
- m_reposition
 - seq64::perform, [786](#)
- m_resume_note_ons
 - seq64::perform, [792](#)
- m_reveal_alsa_ports
 - seq64::rc_settings, [857](#)
- m_right_marker_tick
 - seq64::perftime, [832](#)
- m_right_tick
 - seq64::perform, [794](#)
- m_rightbox
 - seq64::eventedit, [234](#)
- m_ring
 - seq64::midi_queue, [575](#)
- m_ring_size
 - seq64::midi_queue, [574](#)
- m_roll_length_ticks
 - seq64::perfroll, [821](#)
- m_rt_midi
 - seq64::midibus, [614](#)
- m_safety_mutex
 - seq64::automutex, [126](#)
- m_save_user_config
 - seq64::user_settings, [1202](#)
- m_scale
 - seq64::seqedit, [946](#)
 - seq64::seqkeys, [984](#)
 - seq64::seqroll, [1028](#)
- m_scale_phase
 - seq64::lfownd, [403](#)
- m_scale_range
 - seq64::lfownd, [403](#)
- m_scale_speed
 - seq64::lfownd, [403](#)
- m_scale_value
 - seq64::lfownd, [403](#)
- m_scale_wave
 - seq64::lfownd, [403](#)
- m_screensets
 - seq64::mainwid, [422](#)
 - seq64::perform, [796](#)
- m_screensets_notepad
 - seq64::perform, [795](#)
- m_screensets_offset
 - seq64::mainwid, [425](#)
 - seq64::perform, [796](#)
- m_screensets_slots
 - seq64::mainwid, [424](#)
- m_screensets_state
 - seq64::perform, [790](#)
- m_scroll_offset_key
 - seq64::seqkeys, [983](#)
 - seq64::seqroll, [1030](#)
- m_scroll_offset_ticks
 - seq64::seqdata, [919](#)
 - seq64::seqevent, [971](#)
 - seq64::seqroll, [1030](#)
 - seq64::seqtime, [1038](#)
- m_scroll_offset_x
 - seq64::seqdata, [919](#)
 - seq64::seqevent, [972](#)
 - seq64::seqroll, [1030](#)
 - seq64::seqtime, [1038](#)
- m_scroll_offset_y
 - seq64::seqkeys, [984](#)
 - seq64::seqroll, [1031](#)
- m_scroll_page
 - seq64::perfroll, [820](#)

- seq64::seqroll, 1031
- m_sel_paint
 - seq64::gui_palette_gtk2, 315
- m_selected
 - seq64::event, 205
 - seq64::seqevent, 971
 - seq64::seqroll, 1027
 - seq64::trigger, 1128
- m_selecting
 - seq64::seqevent, 972
 - seq64::seqroll, 1029
- m_seq
 - seq64::eventedit, 236
 - seq64::eventslots, 254
 - seq64::lfownd, 403
 - seq64::mastermidibase, 479
 - seq64::midi_alsa, 490
 - seq64::seqdata, 918
 - seq64::seqedit, 947
 - seq64::seqevent, 971
 - seq64::seqkeys, 983
 - seq64::seqroll, 1027
 - seq64::seqtime, 1038
- m_seq_color
 - seq64::sequence, 1121
- m_seq_edit_mode
 - seq64::sequence, 1121
- m_seq_number
 - seq64::sequence, 1121
- m_seqarea_seq_x
 - seq64::mainwid, 423
 - seq64::user_settings, 1201
- m_seqarea_seq_y
 - seq64::mainwid, 423
 - seq64::user_settings, 1201
- m_seqarea_x
 - seq64::mainwid, 423
 - seq64::user_settings, 1201
- m_seqarea_y
 - seq64::mainwid, 423
 - seq64::user_settings, 1201
- m_seqchars_x
 - seq64::user_settings, 1198
- m_seqchars_y
 - seq64::user_settings, 1198
- m_seqdata
 - seq64::lfownd, 403
- m_seqdata_wid
 - seq64::seqedit, 951
 - seq64::seqevent, 972
- m_seqedit
 - seq64::seqmenu, 997
- m_seqedit_bgsequence
 - seq64::user_settings, 1195
- m_seqedit_key
 - seq64::user_settings, 1195
- m_seqedit_scale
 - seq64::user_settings, 1195
- m_seqevent_wid
 - seq64::seqedit, 951
- m_seqkeys_wid
 - seq64::seqedit, 951
 - seq64::seqroll, 1027
- m_seqroll_wid
 - seq64::seqedit, 951
- m_seqs
 - seq64::perform, 788
- m_seqs_active
 - seq64::perform, 789
- m_seqs_in_set
 - seq64::perfnames, 681
 - seq64::perform, 787
 - seq64::user_settings, 1200
- m_seqtime_wid
 - seq64::seqedit, 951
- m_sequence
 - seq64::editable_events, 179
 - seq64::midi_container, 515
- m_sequence_active
 - seq64::perfnames, 681
 - seq64::perfroll, 821
- m_sequence_count
 - seq64::perform, 797
- m_sequence_high
 - seq64::perform, 797
- m_sequence_max
 - seq64::perfnames, 681
 - seq64::perform, 797
 - seq64::perfroll, 821
- m_sequence_offset
 - seq64::perfnames, 681
 - seq64::perfroll, 821
- m_sequence_state
 - seq64::perform, 789
- m_setbox_w
 - seq64::eventslots, 255
 - seq64::perfnames, 680
- m_show_midi
 - seq64::rc_settings, 856
- m_show_octave_letters
 - seq64::seqkeys, 984
- m_showbox
 - seq64::eventedit, 233
- m_size
 - seq64::midi_queue, 574
- m_size_box_w
 - seq64::perfroll, 820
- m_slot
 - seq64::keybindentry, 349
- m_slots_chars
 - seq64::eventslots, 254
- m_slots_x
 - seq64::eventslots, 255
- m_slots_y
 - seq64::eventslots, 255
- m_smf0_channels

- seq64::midi_splitter, 579
- m_smf0_channels_count
 - seq64::midi_splitter, 579
- m_smf0_main_sequence
 - seq64::midi_splitter, 579
- m_smf0_seq_number
 - seq64::midi_splitter, 579
- m_smf0_splitter
 - seq64::midifile, 639
- m_snap
 - seq64::perfedit, 672
 - seq64::perftime, 832
 - seq64::seqedit, 946
 - seq64::seqevent, 971
 - seq64::seqroll, 1028
- m_snap_tick
 - seq64::sequence, 1121
- m_snap_x
 - seq64::perffroll, 818
- m_snap_y
 - seq64::perffroll, 818
- m_song_mute
 - seq64::sequence, 1116
- m_song_playback_block
 - seq64::sequence, 1118
- m_song_record_snap
 - seq64::perform, 791
- m_song_record_tick
 - seq64::sequence, 1119
- m_song_recording
 - seq64::perform, 791
 - seq64::sequence, 1118
- m_song_recording_snap
 - seq64::sequence, 1119
- m_song_start_mode
 - seq64::perform, 786
- m_speed
 - seq64::lfownd, 404
- m_spinbutton_bpm
 - seq64::mainwnd, 456
- m_spinbutton_load_offset
 - seq64::mainwnd, 456
- m_spinbutton_ss
 - seq64::mainwnd, 453
- m_standard_bpm
 - seq64::perfedit, 672
- m_start_from_perfedit
 - seq64::perform, 786
- m_starting_tick
 - seq64::perform, 794
- m_stats
 - seq64::rc_settings, 856
- m_status
 - seq64::event, 204
 - seq64::midi_control, 520
 - seq64::seqdata, 919
 - seq64::seqevent, 973
 - seq64::seqroll, 1032
- m_status_label
 - seq64::mainwnd, 451
- m_sysex
 - seq64::event, 204
- m_table
 - seq64::eventedit, 233
 - seq64::perfedit, 667
 - seq64::seqedit, 951
- m_tempo_paint
 - seq64::gui_palette_gtk2, 315
- m_tempo_track_number
 - seq64::perform, 793
 - seq64::rc_settings, 860
- m_text_size_x
 - seq64::mainwid, 424
- m_text_size_y
 - seq64::mainwid, 424
- m_text_x
 - seq64::user_settings, 1197
- m_text_y
 - seq64::user_settings, 1198
- m_thru
 - seq64::sequence, 1118
- m_tick
 - seq64::maintime, 409
 - seq64::perform, 794
- m_tick_end
 - seq64::trigger, 1128
- m_tick_offset
 - seq64::perftime, 831
- m_tick_start
 - seq64::trigger, 1128
- m_tick_time
 - seq64::mainwnd, 456
- m_tick_time_as_bbt
 - seq64::mainwnd, 456
- m_ticks_per_bar
 - seq64::perffroll, 819
- m_time_beat_width
 - seq64::sequence, 1122
- m_time_beats_per_measure
 - seq64::sequence, 1122
- m_timearea_y
 - seq64::perftime, 832
- m_timeout_connect
 - seq64::mainwnd, 457
- m_timestamp
 - seq64::event, 203
 - seq64::event_list::event_key, 207
 - seq64::midi_message, 564
- m_toggle_follow
 - seq64::seqedit, 956
- m_toggle_jack
 - seq64::jack_assistant, 342
- m_toggle_play
 - seq64::seqedit, 956
- m_toggle_q_rec
 - seq64::seqedit, 957

- m_toggle_record
 - seq64::seqedit, [956](#)
- m_toggle_thru
 - seq64::seqedit, [957](#)
- m_toggle_transpose
 - seq64::seqedit, [948](#)
- m_tooltips
 - seq64::options, [648](#)
 - seq64::seqedit, [955](#)
- m_top_index
 - seq64::eventslots, [256](#)
- m_top_iterator
 - seq64::eventslots, [256](#)
- m_total_seqs
 - seq64::user_settings, [1200](#)
- m_tracks_mute_state
 - seq64::perform, [788](#)
- m_trans_button_press
 - seq64::seqroll, [1031](#)
- m_transport_follow
 - seq64::seqroll, [1031](#)
- m_transposable
 - seq64::sequence, [1116](#)
- m_transpose
 - seq64::perform, [790](#)
- m_trigger_copied
 - seq64::triggers, [1147](#)
- m_trigger_offset
 - seq64::sequence, [1120](#)
- m_triggers
 - seq64::sequence, [1115](#)
 - seq64::triggers, [1147](#)
- m_type
 - seq64::keybindentry, [349](#)
 - seq64::rterror, [869](#)
- m_undo_stack
 - seq64::triggers, [1147](#)
- m_undo_vect
 - seq64::perform, [798](#)
- m_unit_measure
 - seq64::sequence, [1119](#)
- m_us_per_quarter_note
 - seq64::perform, [793](#)
 - seq64::sequence, [1122](#)
- m_use_default_ppqn
 - seq64::midi_splitter, [579](#)
 - seq64::midifile, [639](#)
- m_use_jack_polling
 - seq64::mastermidibus, [484](#)
- m_use_more_icons
 - seq64::user_settings, [1197](#)
- m_use_new_font
 - seq64::font, [260](#)
 - seq64::user_settings, [1196](#)
- m_usemidiclock
 - seq64::perform, [795](#)
- m_user_callback
 - seq64::rtmidi_in_data, [887](#)
- m_user_data
 - seq64::rtmidi_in_data, [887](#)
- m_user_filename
 - seq64::rc_settings, [859](#)
- m_user_filename_alt
 - seq64::rc_settings, [859](#)
- m_user_option_daemonize
 - seq64::user_settings, [1203](#)
- m_user_option_logfile
 - seq64::user_settings, [1203](#)
- m_user_ui_key_height
 - seq64::user_settings, [1204](#)
- m_user_use_logfile
 - seq64::user_settings, [1203](#)
- m_using_callback
 - seq64::rtmidi_in_data, [887](#)
- m_v_page_increment
 - seq64::perfroll, [818](#)
- m_v_perf_page_increment
 - seq64::user_settings, [1196](#)
- m_vadjust
 - seq64::eventedit, [233](#)
 - seq64::gui_drawingarea_gtk2, [298](#)
 - seq64::perfedit, [668](#)
 - seq64::seqedit, [950](#)
- m_value
 - seq64::lfownd, [404](#)
- m_vbox
 - seq64::seqedit, [952](#)
- m_vector_sequence
 - seq64::mastermidibase, [479](#)
- m_velocity_override
 - seq64::user_settings, [1199](#)
- m_verbose_option
 - seq64::rc_settings, [855](#)
- m_vertical_adjust
 - seq64::seqroll, [1027](#)
- m_vscroll
 - seq64::eventedit, [233](#)
 - seq64::perfedit, [668](#)
- m_vscroll_new
 - seq64::seqedit, [950](#)
- m_w_scale_x
 - seq64::perfroll, [819](#)
- m_was_active_edit
 - seq64::perform, [789](#)
- m_was_active_main
 - seq64::perform, [789](#)
- m_was_active_names
 - seq64::perform, [789](#)
- m_was_active_perf
 - seq64::perform, [789](#)
- m_was_playing
 - seq64::sequence, [1117](#)
- m_wave
 - seq64::lfownd, [404](#)
- m_wave_name
 - seq64::lfownd, [404](#)

- m_white
 - seq64::gui_palette_gtk2, 312
- m_white_pixmap
 - seq64::font, 261
- m_wht_key_paint
 - seq64::gui_palette_gtk2, 315
- m_wht_paint
 - seq64::gui_palette_gtk2, 314
- m_width
 - seq64::rect, 867
- m_window
 - seq64::gui_drawingarea_gtk2, 298
- m_window_redraw_rate_ms
 - seq64::user_settings, 1197
- m_window_scale
 - seq64::user_settings, 1194
- m_window_x
 - seq64::gui_drawingarea_gtk2, 299
 - seq64::gui_window_gtk2, 320
- m_window_y
 - seq64::gui_drawingarea_gtk2, 299
 - seq64::gui_window_gtk2, 320
- m_with_jack_master
 - seq64::rc_settings, 857
- m_with_jack_master_cond
 - seq64::rc_settings, 857
- m_with_jack_midi
 - seq64::rc_settings, 857
- m_with_jack_transport
 - seq64::rc_settings, 857
- m_work_around_play_image
 - seq64::user_settings, 1203
- m_work_around_transpose_image
 - seq64::user_settings, 1203
- m_x
 - seq64::click, 147
 - seq64::rect, 866
- m_xy_offset
 - seq64::perfnames, 680
- m_y
 - seq64::click, 147
 - seq64::rect, 866
- m_y_on_b_pixmap
 - seq64::font, 262
- m_yellow
 - seq64::gui_palette_gtk2, 311
- m_zoom
 - seq64::perfroll, 819
 - seq64::seqdata, 919
 - seq64::seqedit, 946
 - seq64::seqevent, 971
 - seq64::seqroll, 1028
 - seq64::seqtime, 1038
- magenta
 - seq64::gui_palette_gtk2, 309
- main_window_title
 - seq64::perform, 742
- maintime
 - seq64::maintime, 407
- mainwid
 - seq64::mainwid, 413
- mainwid_border
 - seq64::user_settings, 1177, 1183
- mainwid_border_x
 - seq64::user_settings, 1178
- mainwid_border_y
 - seq64::user_settings, 1178
- mainwid_grid_style_t
 - seq64::user_settings, 1169
- mainwid_height
 - seq64::user_settings, 1190
- mainwid_height_fudge
 - seq64::user_settings, 1191
- mainwid_spacing
 - seq64::user_settings, 1177, 1184
- mainwid_width
 - seq64::user_settings, 1190
- mainwid_width_fudge
 - seq64::user_settings, 1191
- mainwid_x
 - seq64::user_settings, 1177
- mainwid_y
 - seq64::user_settings, 1177
- mainwnd
 - seq64::maintime, 408
 - seq64::mainwid, 421
 - seq64::mainwnd, 433
 - seq64::perform, 781
 - seq64::rc_settings, 854
 - seq64::seqmenu, 996
- mainwnd_cols
 - seq64::user_settings, 1175, 1182
- mainwnd_key_event
 - seq64::perform, 744
- mainwnd_rows
 - seq64::user_settings, 1175, 1182
- mainwnd_x
 - seq64::user_settings, 1177
- mainwnd_y
 - seq64::user_settings, 1177
- make_clock
 - seq64::event, 199
- make_directory
 - seq64, 78
- make_section_name
 - seq64, 92
- manual_alsa_ports
 - seq64::rc_settings, 844, 851
- mark
 - seq64::event, 198
- mark_all
 - seq64::event_list, 217
- mark_out_of_range
 - seq64::event_list, 217
- mark_selected
 - seq64::event_list, 217

- seq64::sequence, [1100](#)
- master_bus
 - seq64::perform, [705](#)
- master_info
 - seq64::midi_api, [503](#), [504](#)
- master_midi_mode
 - seq64::midi_api, [504](#)
- mastermidibase
 - seq64::mastermidibase, [463](#)
- mastermidibus
 - seq64::mastermidibus, [481](#)
 - seq64::midibase, [607](#)
 - seq64::midibus, [614](#)
 - seq64::rtmidi_info, [895](#)
- match
 - seq64::midi_control, [519](#)
 - seq64::midibase, [595](#)
- max_active_set
 - seq64::perform, [772](#)
- max_sequence
 - seq64::user_settings, [1176](#)
- max_sets
 - seq64::perform, [755](#), [774](#)
 - seq64::user_settings, [1176](#), [1182](#)
- max_value
 - seq64::midi_control, [519](#)
- max_zoom
 - seq64::user_settings, [1186](#)
- mc_baseline_ppqn
 - seq64::user_settings, [1203](#)
- mc_max_zoom
 - seq64::user_settings, [1203](#)
- mc_min_zoom
 - seq64::user_settings, [1202](#)
- measures
 - seq64::midi_measures, [559](#)
- measures_to_ticks
 - seq64, [67](#)
 - seq64::sequence, [1060](#)
- measurestring_to_pulses
 - seq64, [55](#)
- menu_mode
 - seq64::keys_perform, [365](#)
- merge
 - seq64::event_list, [214](#)
- message
 - seq64::rtmidi_in_data, [883](#)
- message_concatenate
 - seq64, [90](#)
- meta_event_length
 - seq64::editable_event, [161](#)
- meta_string
 - seq64::editable_event, [165](#)
- midi_alsa
 - seq64::midi_alsa, [486](#)
- midi_alsa_info
 - seq64::mastermidibase, [477](#)
 - seq64::mastermidibus, [483](#)
 - seq64::midi_alsa_info, [493](#)
- midi_api
 - seq64::midi_api, [499](#)
- midi_api_name
 - seq64, [99](#)
- midi_beat_width
 - seq64::user_settings, [1185](#), [1192](#)
- midi_beats_per_bar
 - seq64::user_settings, [1184](#), [1191](#)
- midi_beats_per_minute
 - seq64::user_settings, [1184](#), [1191](#)
- midi_bpm_maximum
 - seq64::user_settings, [1185](#), [1191](#)
- midi_bpm_minimum
 - seq64::user_settings, [1184](#), [1191](#)
- midi_buss_override
 - seq64::user_settings, [1185](#), [1189](#)
- midi_container
 - seq64::midi_container, [509](#)
 - seq64::triggers, [1146](#)
- midi_control
 - seq64::midi_control, [518](#)
- midi_control_event
 - seq64::perform, [762](#)
- midi_control_off
 - seq64::perform, [762](#)
- midi_control_on
 - seq64::perform, [762](#)
- midi_control_record
 - seq64::perform, [763](#)
- midi_control_toggle
 - seq64::perform, [761](#)
- midi_handle
 - seq64::midi_info, [530](#), [534](#)
- midi_in_alsa
 - seq64::midi_in_alsa, [523](#)
- midi_in_jack
 - seq64::midi_in_jack, [525](#)
- midi_info
 - seq64::midi_info, [529](#)
- midi_input_test
 - seq64, [99](#)
- midi_jack
 - seq64::midi_jack, [539](#)
 - seq64::midi_jack_info, [553](#)
- midi_jack_data
 - seq64::midi_jack_data, [546](#)
- midi_jack_info
 - seq64::mastermidibus, [483](#)
 - seq64::midi_jack, [545](#)
 - seq64::midi_jack_info, [550](#)
- midi_list
 - seq64::midi_list, [556](#)
- midi_measures
 - seq64::midi_measures, [559](#)
- midi_measures_to_pulses
 - seq64, [56](#)
- midi_message

- seq64::midi_message, [562](#)
- midi_mode
 - seq64::midi_info, [530](#)
 - seq64::rtmidi_info, [890](#)
- midi_mute_group_present
 - seq64::perform, [702](#), [781](#)
- midi_out_alsa
 - seq64::midi_out_alsa, [566](#)
- midi_out_jack
 - seq64::midi_out_jack, [568](#)
- midi_port_info
 - seq64::midi_port_info, [569](#)
- midi_ppqn
 - seq64::user_settings, [1184](#), [1189](#)
- midi_probe
 - seq64, [99](#)
- midi_queue
 - seq64::midi_queue, [572](#)
- midi_splitter
 - seq64::midi_splitter, [576](#)
- midi_timing
 - seq64::midi_timing, [581](#)
- midi_vector
 - seq64::midi_vector, [585](#)
- midibase
 - seq64::midibase, [592](#)
- midibpm
 - seq64, [44](#)
- midibus
 - seq64::midibus, [611](#)
 - seq64::rtmidi, [877](#)
 - seq64::rtmidi_info, [896](#)
- midibyte
 - seq64, [44](#)
- midifile
 - seq64::event_list, [219](#)
 - seq64::midi_container, [515](#)
 - seq64::midifile, [618](#)
 - seq64::perform, [782](#)
 - seq64::triggers, [1146](#)
 - seq64::user_settings, [1192](#)
- midilong
 - seq64, [44](#)
- midipulse
 - seq64, [44](#)
- midishort
 - seq64, [44](#)
- midistring
 - seq64, [44](#)
- millisleep
 - seq64, [87](#)
- min_value
 - seq64::midi_control, [519](#)
- min_zoom
 - seq64::user_settings, [1185](#)
- mod_control
 - seq64::click, [146](#)
 - seq64::keystroke, [395](#)
- mod_control_shift
 - seq64::click, [146](#)
 - seq64::keystroke, [395](#)
- mod_last_tick
 - seq64::sequence, [1066](#)
- mod_super
 - seq64::click, [146](#)
 - seq64::keystroke, [395](#)
- mod_timestamp
 - seq64::event, [190](#)
- modifier
 - seq64::click, [146](#)
 - seq64::keystroke, [395](#)
- modify
 - seq64::perform, [701](#)
 - seq64::sequence, [1056](#)
- modify_current_event
 - seq64::eventslots, [246](#)
- motion_notify
 - seq64::seqroll, [1020](#)
- mouse_action
 - seq64::seqedit, [942](#)
- mouse_action_e
 - seq64, [50](#)
- mouse_click_edit_callback
 - seq64::options, [647](#)
- mouse_fruity_callback
 - seq64::options, [646](#)
- mouse_mod4_callback
 - seq64::options, [646](#)
- mouse_seq24_callback
 - seq64::options, [646](#)
- mouse_snap_split_callback
 - seq64::options, [646](#)
- move
 - seq64::triggers, [1142](#)
- move_selected
 - seq64::triggers, [1140](#)
- move_selected_notes
 - seq64::seqroll, [1017](#)
 - seq64::sequence, [1095](#)
- move_selection_box
 - seq64::seqroll, [1017](#)
- move_triggers
 - seq64::perform, [747](#)
 - seq64::sequence, [1083](#), [1085](#)
- moving
 - seq64::perfroll, [814](#)
 - seq64::seqroll, [1023](#)
- multi_wid
 - seq64::mainwnd, [435](#)
- multiply_pattern
 - seq64::sequence, [1107](#)
- musical_key
 - seq64::sequence, [1107](#)
- musical_scale
 - seq64::sequence, [1108](#)
- mute_all_tracks

- seq64::perform, 724
- seq64::seqmenu, 996
- mute_group_handling_t
 - seq64, 48
- mute_group_offset
 - seq64::perform, 728
- mute_group_saving
 - seq64::rc_settings, 845, 852
- mute_group_tracks
 - seq64::perform, 766
- mute_op_t
 - seq64::perform, 699
- mute_screenset
 - seq64::perform, 725
- mutex
 - seq64::mutex, 641
- name
 - seq64::sequence, 1062
 - seq64::user_instrument, 1150
 - seq64::user_midi_bus, 1155
- name_change_callback
 - seq64::seqedit, 936
- name_to_value
 - seq64::editable_event, 161
- nc_midi_port_info
 - seq64::midi_info, 535
- need_set_spinner
 - seq64::mainwnd, 436
- new_current_sequence
 - seq64::seqmenu, 991
- new_file
 - seq64::mainwnd, 446
- new_open_error_dialog
 - seq64::mainwnd, 445
- new_sequence
 - seq64::perform, 715
 - seq64::seqmenu, 992
- next
 - seq64::triggers, 1143
- next_data_line
 - seq64::configfile, 151
- next_trigger
 - seq64::triggers, 1144
- nominal_height
 - seq64::mainwid, 415
- nominal_width
 - seq64::mainwid, 415
- non_cc_match
 - seq64::event, 193
- normal_action
 - seq64::seqroll, 1022
- normalize
 - seq64::user_settings, 1170
- normalize_path
 - seq64, 79
- note_off_length
 - seq64::seqroll, 1006
- note_off_margin
 - seq64::sequence, 1109
- note_value_to_tempo
 - seq64, 62
- number
 - seq64::sequence, 1055
- number_selected
 - seq64::triggers, 1133
- off_from_snap
 - seq64::sequence, 1071, 1113
- off_one_shot
 - seq64::sequence, 1069
- off_playing_notes
 - seq64::sequence, 1102
- off_sequences
 - seq64::perform, 770
- offset
 - seq64::trigger, 1127
- offset_sequence
 - seq64::perfroll, 810
- on_button_press_event
 - seq64::FruityPerfInput, 265
 - seq64::FruitySeqEventInput, 269
 - seq64::FruitySeqRollInput, 273
 - seq64::Seq24PerfInput, 902
 - seq64::Seq24SeqEventInput, 906
 - seq64::eventslots, 252
 - seq64::mainwid, 419
 - seq64::perfnames, 678
 - seq64::perfroll, 815
 - seq64::perftime, 830
 - seq64::seqdata, 916
 - seq64::sequevent, 968
 - seq64::seqkeys, 981
 - seq64::seqroll, 1023
 - seq64::seqtime, 1037
- on_button_release_event
 - seq64::FruityPerfInput, 265
 - seq64::FruitySeqEventInput, 270
 - seq64::FruitySeqRollInput, 274
 - seq64::Seq24PerfInput, 903
 - seq64::Seq24SeqEventInput, 906
 - seq64::eventslots, 252
 - seq64::mainwid, 420
 - seq64::perfnames, 678
 - seq64::perfroll, 815
 - seq64::perftime, 830
 - seq64::seqdata, 916
 - seq64::sequevent, 969
 - seq64::seqkeys, 981
 - seq64::seqroll, 1024
 - seq64::seqtime, 1037
- on_delete_event
 - seq64::eventedit, 232
 - seq64::mainwnd, 449
 - seq64::perfeddit, 667
 - seq64::seqedit, 943
- on_enter_notify_event
 - seq64::seqkeys, 982

- seq64::seqroll, 1026
- on_expose_event
 - seq64::eventslots, 252
 - seq64::maintime, 408
 - seq64::mainwid, 419
 - seq64::perfnames, 677
 - seq64::perfroll, 814
 - seq64::perftime, 829
 - seq64::seqdata, 916
 - seq64::seqevent, 968
 - seq64::seqkeys, 981
 - seq64::seqroll, 1023
 - seq64::seqtime, 1037
- on_focus_in_event
 - seq64::eventedit, 231
 - seq64::eventslots, 252
 - seq64::mainwid, 421
 - seq64::perfroll, 816
 - seq64::seqedit, 943
 - seq64::seqevent, 970
 - seq64::seqroll, 1024
- on_focus_out_event
 - seq64::eventedit, 231
 - seq64::eventslots, 252
 - seq64::lfownd, 402
 - seq64::mainwid, 421
 - seq64::perfroll, 816
 - seq64::seqedit, 943
 - seq64::seqevent, 970
 - seq64::seqroll, 1025
- on_frame_down
 - seq64::eventslots, 253
- on_frame_end
 - seq64::eventslots, 254
- on_frame_home
 - seq64::eventslots, 253
- on_frame_up
 - seq64::eventslots, 253
- on_grouplearnchange
 - seq64::mainwnd, 450
 - seq64::performcallback, 800
- on_key_press_event
 - seq64::eventedit, 232
 - seq64::keybindentry, 348
 - seq64::mainwnd, 449
 - seq64::perfedit, 666
 - seq64::perfroll, 816
 - seq64::seqedit, 944
 - seq64::seqevent, 970
 - seq64::seqroll, 1025
- on_key_release_event
 - seq64::mainwnd, 449
 - seq64::perfedit, 667
- on_leave_notify_event
 - seq64::seqdata, 917
 - seq64::seqkeys, 982
 - seq64::seqroll, 1026
- on_left_button_pressed
 - seq64::FruityPerfInput, 266
- on_motion_notify_event
 - seq64::FruityPerfInput, 266
 - seq64::FruitySeqEventInput, 271
 - seq64::FruitySeqRollInput, 274
 - seq64::Seq24PerfInput, 904
 - seq64::Seq24SeqEventInput, 907
 - seq64::mainwid, 420
 - seq64::perfroll, 815
 - seq64::seqdata, 917
 - seq64::seqevent, 969
 - seq64::seqkeys, 982
 - seq64::seqroll, 1024
- on_move_down
 - seq64::eventslots, 253
- on_move_up
 - seq64::eventslots, 253
- on_realize
 - seq64::eventedit, 231
 - seq64::eventslots, 252
 - seq64::gui_drawingarea_gtk2, 298
 - seq64::gui_window_gtk2, 320
 - seq64::mainwid, 419
 - seq64::mainwnd, 449
 - seq64::perfedit, 666
 - seq64::perfnames, 677
 - seq64::perfroll, 814
 - seq64::perftime, 829
 - seq64::seqdata, 916
 - seq64::seqedit, 943
 - seq64::seqevent, 968
 - seq64::seqkeys, 980
 - seq64::seqmenu, 996
 - seq64::seqroll, 1023
 - seq64::seqtime, 1037
- on_right_button_pressed
 - seq64::FruityPerfInput, 267
- on_scroll_event
 - seq64::eventslots, 253
 - seq64::perfnames, 679
 - seq64::perfroll, 815
 - seq64::seqdata, 917
 - seq64::seqedit, 944
 - seq64::seqkeys, 982
 - seq64::seqroll, 1026
- on_set_focus
 - seq64::eventedit, 231
 - seq64::seqedit, 943
- on_size_allocate
 - seq64::eventslots, 253
 - seq64::perfnames, 679
 - seq64::perfroll, 816
 - seq64::perftime, 830
 - seq64::seqdata, 918
 - seq64::seqevent, 970
 - seq64::seqkeys, 983
 - seq64::seqroll, 1026
 - seq64::seqtime, 1037

- on_size_request
 - seq64::perfull, 816
- one_shot
 - seq64::sequence, 1070, 1113
- one_shot_tick
 - seq64::sequence, 1070
- oneshot_queue
 - seq64::keys_perform, 368
- open_client
 - seq64::midi_in_jack, 525
 - seq64::midi_jack, 541
 - seq64::midi_out_jack, 568
- open_client_impl
 - seq64::midi_jack, 540
- open_file
 - seq64::mainwnd, 434
- open_performance_edit
 - seq64::mainwnd, 441
- open_performance_edit_2
 - seq64::mainwnd, 442
- openmidi_api
 - seq64::rtmidi_in, 881
 - seq64::rtmidi_info, 895
 - seq64::rtmidi_out, 898
- operator<
 - seq64::event, 186
 - seq64::event_list::event_key, 206
 - seq64::trigger, 1125
- operator=
 - seq64::automutex, 126
 - seq64::click, 145
 - seq64::editable_event, 162
 - seq64::editable_events, 174
 - seq64::event, 186
 - seq64::event_list, 211
 - seq64::gui_drawingarea_gtk2, 285
 - seq64::keystroke, 392
 - seq64::maintime, 407
 - seq64::rc_settings, 840
 - seq64::sequence, 1053
 - seq64::triggers, 1132
 - seq64::user_instrument, 1150
 - seq64::user_midi_bus, 1155
 - seq64::user_settings, 1170
- operator[]
 - seq64::midi_message, 562
- option_daemonize
 - seq64::user_settings, 1186, 1188
- option_logfile
 - seq64::user_settings, 1186, 1189
- option_use_logfile
 - seq64::user_settings, 1186, 1189
- options
 - seq64::keybindentry, 349
 - seq64::keys_perform, 377
 - seq64::options, 644
 - seq64::perform, 782
 - seq64::rc_settings, 854
- options_dialog
 - seq64::mainwnd, 444
- optionsfile
 - seq64::keys_perform, 377
 - seq64::optionsfile, 651
 - seq64::perform, 782
 - seq64::rc_settings, 854
- orange
 - seq64::gui_palette_gtk2, 309
- output
 - seq64::jack_assistant, 332
- output_func
 - seq64::perform, 725
- output_ports
 - seq64::midi_info, 530
- output_thread_func
 - seq64, 87
 - seq64::perform, 783
- padded_height
 - seq64::font, 260
- page_decrement_beats_per_minute
 - seq64::perform, 738
- page_increment_beats_per_minute
 - seq64::perform, 738
- page_movement
 - seq64::eventslots, 249
- page_topper
 - seq64::eventslots, 250
- pager_index
 - seq64::eventslots, 243
- paint
 - seq64::event, 198
- panic
 - seq64::mainwnd, 441
 - seq64::mastermidibase, 468
 - seq64::perform, 761
- parent
 - seq64::editable_event, 162
 - seq64::jack_assistant, 326
- parent_bus
 - seq64::midi_api, 504
- parse
 - seq64::configfile, 153
 - seq64::midifile, 618
 - seq64::optionsfile, 651
 - seq64::userfile, 1205
- parse_command_line_options
 - seq64, 74
 - seq64::rc_settings, 854
- parse_log_option
 - seq64, 73
- parse_mute_group_section
 - seq64::optionsfile, 653
- parse_mute_groups
 - seq64, 72
- parse_o_options
 - seq64, 73
 - seq64::user_settings, 1192

- parse_options_files
 - seq64, [71](#)
- parse_prop_header
 - seq64::midifile, [624](#)
- parse_proprietary_track
 - seq64::midifile, [625](#)
- parse_smf_0
 - seq64::midifile, [622](#)
- parse_smf_1
 - seq64::midifile, [622](#)
- partial_assign
 - seq64::sequence, [1053](#)
- pass_sysex
 - seq64::rc_settings, [843](#), [850](#)
- paste
 - seq64::triggers, [1140](#)
- paste_or_split_trigger
 - seq64::perform, [750](#)
- paste_selected
 - seq64::sequence, [1092](#)
- paste_trigger
 - seq64::perform, [750](#)
 - seq64::sequence, [1083](#)
- pattern_edit
 - seq64::keys_perform, [363](#), [364](#)
- pattern_shift
 - seq64::keys_perform, [364](#)
- pause
 - seq64::keys_perform, [363](#)
 - seq64::sequence, [1102](#)
- pause_key
 - seq64::perform, [737](#)
- pause_playing
 - seq64::mainwnd, [439](#)
 - seq64::perfedit, [666](#)
 - seq64::perform, [736](#)
- perf
 - seq64::gui_drawingarea_gtk2, [287](#)
 - seq64::gui_window_gtk2, [318](#)
 - seq64::options, [644](#)
- perf_h_page_increment
 - seq64::user_settings, [1180](#), [1187](#)
- perf_modify
 - seq64::eventedit, [228](#)
- perf_v_page_increment
 - seq64::user_settings, [1180](#), [1187](#)
- perfedit
 - seq64::perfedit, [659](#)
 - seq64::perfnames, [679](#)
 - seq64::perform, [782](#)
 - seq64::perfroll, [817](#)
 - seq64::perftime, [831](#)
- perfnames
 - seq64::perfnames, [675](#)
- perform
 - seq64::keys_perform, [377](#)
 - seq64::mastermidibase, [477](#)
 - seq64::perform, [700](#)
 - seq64::sequence, [1114](#)
- perfroll
 - seq64::perform, [782](#)
 - seq64::perfroll, [805](#)
- perfroll_key_event
 - seq64::perform, [746](#)
- perftime
 - seq64::perftime, [825](#)
- pixel_to_tick
 - seq64::perftime, [828](#)
- play
 - seq64::busarray, [131](#)
 - seq64::mastermidibase, [466](#)
 - seq64::midibase, [601](#)
 - seq64::perform, [771](#)
 - seq64::sequence, [1074](#)
 - seq64::triggers, [1134](#)
- play_change_callback
 - seq64::seqedit, [936](#)
- play_note_off
 - seq64::sequence, [1102](#)
- play_note_on
 - seq64::sequence, [1101](#)
- play_queue
 - seq64::sequence, [1074](#)
- playback_key_event
 - seq64::perform, [747](#)
- playback_mode
 - seq64::perform, [760](#)
- pointer_position
 - seq64::keys_perform, [366](#)
- poll_for_midi
 - seq64::busarray, [135](#)
 - seq64::mastermidibase, [470](#)
 - seq64::midibase, [600](#)
- pop
 - seq64::midi_queue, [573](#)
- pop_front
 - seq64::midi_queue, [573](#)
- pop_redo
 - seq64::sequence, [1058](#)
 - seq64::triggers, [1134](#)
- pop_trigger_redo
 - seq64::perform, [748](#)
 - seq64::sequence, [1058](#)
- pop_trigger_undo
 - seq64::perform, [748](#)
 - seq64::sequence, [1058](#)
- pop_undo
 - seq64::sequence, [1057](#)
 - seq64::triggers, [1134](#)
- populate_menu_edit
 - seq64::mainwnd, [448](#)
- populate_menu_file
 - seq64::mainwnd, [448](#)
- populate_menu_help
 - seq64::mainwnd, [448](#)
- populate_menu_view

- seq64::mainwnd, [448](#)
- popup_event_menu
 - seq64::seqedit, [939](#)
- popup_menu
 - seq64::perfedit, [665](#)
 - seq64::seqedit, [939](#)
 - seq64::seqmenu, [992](#)
- popup_midibus_menu
 - seq64::seqedit, [940](#)
- popup_midich_menu
 - seq64::seqedit, [941](#)
- popup_mini_event_menu
 - seq64::seqedit, [940](#)
- popup_record_menu
 - seq64::seqedit, [940](#)
- popup_sequence_menu
 - seq64::seqedit, [940](#)
- popup_tool_menu
 - seq64::seqedit, [941](#)
- port_disabled
 - seq64::midibase, [597](#)
- port_exit
 - seq64::busarray, [133](#)
 - seq64::mastermidibase, [466](#)
- port_handle
 - seq64::midi_jack, [540](#)
- port_list
 - seq64::mastermidibus, [483](#)
 - seq64::midi_info, [534](#)
 - seq64::rtmidi_info, [894](#)
- port_name
 - seq64::midibase, [594](#), [604](#)
- port_start
 - seq64::mastermidibase, [465](#)
- position
 - seq64::jack_assistant, [330](#)
 - seq64::midi_container, [511](#)
- position_increment
 - seq64::midi_container, [512](#)
- position_jack
 - seq64::perform, [769](#)
- position_reset
 - seq64::midi_container, [511](#)
- ppqn
 - seq64::mainwnd, [434](#)
 - seq64::midi_info, [531](#)
 - seq64::midi_splitter, [578](#)
 - seq64::midi_timing, [582](#)
 - seq64::midibase, [595](#)
 - seq64::midifile, [620](#), [621](#)
 - seq64::perform, [701](#), [761](#)
 - seq64::rtmidi_info, [893](#)
- ppqn_is_valid
 - seq64, [63](#)
- preallocate_clocks
 - seq64::perform, [779](#)
- print
 - seq64::busarray, [133](#)
 - seq64::businfo, [141](#)
 - seq64::editable_events, [178](#)
 - seq64::event, [203](#)
 - seq64::event_list, [219](#)
 - seq64::mastermidibase, [468](#)
 - seq64::midibase, [603](#)
 - seq64::sequence, [1073](#)
 - seq64::triggers, [1134](#)
- print_busses
 - seq64::perform, [717](#)
- print_group_unmutes
 - seq64::perform, [765](#)
- print_keys
 - seq64::rc_settings, [845](#), [851](#)
- print_message
 - seq64::rterror, [869](#)
- print_triggers
 - seq64::perform, [747](#)
 - seq64::sequence, [1074](#)
- priority
 - seq64::rc_settings, [843](#), [850](#)
- private_bus
 - seq64::user_settings, [1192](#)
- private_instrument
 - seq64::user_settings, [1192](#)
- process_events
 - seq64::lash, [398](#)
- progress_bar_colored
 - seq64::user_settings, [1180](#), [1187](#)
- progress_bar_thick
 - seq64::user_settings, [1180](#), [1187](#)
- progress_color
 - seq64::gui_palette_gtk2, [307](#)
- prop_item_size
 - seq64::midifile, [635](#)
- pulse_divide
 - seq64, [69](#)
- pulse_length_us
 - seq64, [64](#)
- pulses_per_measure
 - seq64, [67](#)
- pulses_to_measurestring
 - seq64, [53](#)
- pulses_to_midi_measures
 - seq64, [54](#)
- pulses_to_string
 - seq64, [53](#)
- pulses_to_timestring
 - seq64, [54](#), [55](#)
- push
 - seq64::midi_message, [563](#)
- push_back
 - seq64::event_list, [213](#)
- push_quantize
 - seq64::sequence, [1106](#)
- push_trigger_undo
 - seq64::perform, [748](#)
 - seq64::sequence, [1058](#)

- push_undo
 - seq64::sequence, [1057](#)
 - seq64::triggers, [1133](#)
- put
 - seq64::midi_container, [511](#)
 - seq64::midi_list, [557](#)
 - seq64::midi_vector, [586](#)
- put_event_on_bus
 - seq64::sequence, [1111](#)
- q_rec_change_callback
 - seq64::seqedit, [937](#)
- quantize_events
 - seq64::sequence, [1106](#)
- query_save_changes
 - seq64::mainwnd, [445](#)
- queue
 - seq64::keys_perform, [359](#)
 - seq64::rtmidi_in_data, [882](#), [883](#)
- queue_it
 - seq64::mainwnd, [440](#)
- queue_number
 - seq64::midi_info, [533](#)
 - seq64::midibase, [597](#)
 - seq64::rtmidi_info, [892](#)
- quit
 - seq64::gui_assistant, [277](#)
 - seq64::gui_assistant_gtk2, [280](#)
 - seq64::gui_window_gtk2, [318](#)
- rc
 - seq64, [88](#)
- rc_error_dialog
 - seq64::mainwnd, [434](#)
- rc_settings
 - seq64::rc_settings, [840](#)
- read_byte
 - seq64::midifile, [627](#)
- read_byte_array
 - seq64::midifile, [627](#), [628](#)
- read_gap
 - seq64::midifile, [628](#)
- read_long
 - seq64::midifile, [627](#)
- read_seek
 - seq64::midifile, [626](#)
- read_seq_number
 - seq64::midifile, [631](#)
- read_short
 - seq64::midifile, [627](#)
- read_track_name
 - seq64::midifile, [631](#)
- read_varinum
 - seq64::midifile, [627](#)
- recent_file
 - seq64::rc_settings, [848](#)
- recent_file_count
 - seq64::rc_settings, [849](#)
- record_change_callback
 - seq64::seqedit, [936](#)
- record_tempo_op_t
 - seq64::perform, [699](#)
- recording_next_measure
 - seq64::sequence, [1068](#)
- rect
 - seq64::rect, [862](#)
- red
 - seq64::gui_palette_gtk2, [309](#)
- redo
 - seq64::perfedit, [665](#)
- redo_callback
 - seq64::seqedit, [937](#)
- redraw
 - seq64::mainwid, [415](#)
 - seq64::perfnames, [677](#)
 - seq64::seqdata, [911](#)
 - seq64::seqevent, [963](#)
 - seq64::seqmenu, [994](#)
 - seq64::seqroll, [1010](#)
 - seq64::seqtime, [1035](#)
- redraw_dirty_sequences
 - seq64::perfnames, [676](#)
 - seq64::perfroll, [808](#)
- redraw_events
 - seq64::seqroll, [1011](#)
- redraw_period_ms
 - seq64::gui_window_gtk2, [318](#)
- redraw_progress
 - seq64::perfroll, [808](#)
- ref_midi_port_info
 - seq64::midi_info, [535](#)
- register_port
 - seq64::midi_jack, [541](#)
- reload_mute_groups
 - seq64::mainwnd, [442](#)
 - seq64::perform, [721](#)
- remote_port_name
 - seq64::midi_jack, [540](#)
- remove
 - seq64::businfo, [138](#)
 - seq64::editable_events, [178](#)
 - seq64::event_list, [214](#)
 - seq64::sequence, [1112](#)
 - seq64::triggers, [1138](#)
- remove_all
 - seq64::sequence, [1113](#)
- remove_marked
 - seq64::event_list, [218](#)
 - seq64::sequence, [1100](#)
- remove_recent_file
 - seq64::rc_settings, [847](#)
- remove_selected
 - seq64::sequence, [1100](#)
 - seq64::triggers, [1140](#)
- remove_seqedit
 - seq64::seqmenu, [993](#)
- render_digits

- seq64::seqdata, 915
- render_number
 - seq64::seqdata, 915
- render_string
 - seq64::gui_drawingarea_gtk2, 291
- render_string_on_drawable
 - seq64::font, 259
- render_string_on_pixmap
 - seq64::gui_drawingarea_gtk2, 291
- replace
 - seq64::editable_events, 178
 - seq64::keys_perform, 359
- replacement_port
 - seq64::busarray, 135
- repopulate_event_menu
 - seq64::seqedit, 939
- repopulate_midich_menu
 - seq64::seqedit, 941
- repopulate_mini_event_menu
 - seq64::seqedit, 940
- reposition
 - seq64::perform, 711
- reroute_stdio
 - seq64, 75
- reset
 - seq64::mainwid, 414
 - seq64::mainwnd, 437
 - seq64::perftime, 826
 - seq64::seqdata, 911
 - seq64::seqevent, 962
 - seq64::seqkeys, 980
 - seq64::seqroll, 1010
 - seq64::seqtime, 1035
- reset_draw_marker
 - seq64::sequence, 1103
- reset_draw_trigger_marker
 - seq64::sequence, 1103
 - seq64::triggers, 1144
- reset_ex_iterator
 - seq64::sequence, 1103
- reset_loop
 - seq64::sequence, 1111
- reset_sequences
 - seq64::perform, 771
- reset_window
 - seq64::mainwnd, 438
- restart_sysex
 - seq64::event, 196
- restore_playing_state
 - seq64::perform, 730
- resume_note_ons
 - seq64::perform, 755
 - seq64::sequence, 1071
- RevSlotMap
 - seq64::keys_perform, 357
- reveal_alsa_ports
 - seq64::rc_settings, 845, 851
- reverse_iterator
 - seq64::editable_events, 172
 - seq64::event_list, 210
- rewind
 - seq64::keys_perform, 366
 - seq64::perfedit, 661
 - seq64::perform, 710
- rterror
 - seq64::rterror, 868
- rterror_callback
 - seq64, 45
- rtmidi
 - seq64::rtmidi, 872
- rtmidi_api
 - seq64, 51
- rtmidi_callback_t
 - seq64, 45
- rtmidi_in
 - seq64::rtmidi_in, 880
 - seq64::rtmidi_info, 896
- rtmidi_in_data
 - seq64::rtmidi_in_data, 882
- rtmidi_info
 - seq64::midi_info, 535
 - seq64::rc_settings, 854
 - seq64::rtmidi_info, 889
- rtmidi_out
 - seq64::rtmidi_info, 896
 - seq64::rtmidi_out, 898
- s_arg_list
 - seq64, 121
- s_bitness
 - seq64, 122
- s_character_mapping
 - seq64, 122
- s_global_lash_driver
 - seq64, 122
- s_handlesize
 - seq64, 123
- s_help_1a
 - seq64, 121
- s_help_1b
 - seq64, 121
- s_help_2
 - seq64, 121
- s_help_3
 - seq64, 121
- s_help_4
 - seq64, 121
- s_help_5
 - seq64, 122
- s_jitter_amount
 - seq64, 123
- s_status_pairs
 - seq64, 122
- save_clock
 - seq64::mastermidibase, 476
- save_current_screenset
 - seq64::perform, 730

- save_events
 - seq64::editable_events, 175
 - seq64::eventslots, 247
- save_file
 - seq64::mainwnd, 447
- save_input
 - seq64::mastermidibase, 477
- save_mute_group
 - seq64::perform, 768
- save_playing_state
 - seq64::perform, 730
- save_user_config
 - seq64::user_settings, 1181
- SaveOption
 - seq64::mainwnd, 433
- scale_lfo_change
 - seq64::lflownd, 402
- scale_size
 - seq64::user_settings, 1174
- screenshot
 - seq64::perform, 753
- screenshot_dn
 - seq64::keys_perform, 361
- screenshot_offset
 - seq64::perform, 730, 775
- screenshot_up
 - seq64::keys_perform, 360, 361
- scroll_hadjust
 - seq64::gui_drawingarea_gtk2, 296
 - seq64::gui_window_gtk2, 318
- scroll_hset
 - seq64::gui_drawingarea_gtk2, 297
 - seq64::gui_window_gtk2, 319
- scroll_offset_x
 - seq64::seqroll, 1021
- scroll_offset_y
 - seq64::seqroll, 1021
- scroll_vadjust
 - seq64::gui_drawingarea_gtk2, 296
 - seq64::gui_window_gtk2, 319
- scroll_vset
 - seq64::gui_drawingarea_gtk2, 297
 - seq64::gui_window_gtk2, 319
- sel_paint
 - seq64::gui_palette_gtk2, 310
- select
 - seq64::event, 199
 - seq64::triggers, 1138, 1145
- select_action
 - seq64::seqroll, 1022
- select_action_t
 - seq64::sequence, 1052
- select_all
 - seq64::event_list, 219
 - seq64::sequence, 1091
- select_all_notes
 - seq64::sequence, 1090
- select_and_mute_group
 - seq64::perform, 766
- select_even_or_odd_notes
 - seq64::sequence, 1089
- select_event
 - seq64::eventslots, 247
- select_event_handle
 - seq64::sequence, 1088
- select_events
 - seq64::sequence, 1087, 1088
- select_fg_bg_colors
 - seq64::mainwid, 419
- select_group_mute
 - seq64::perform, 767
- select_linked
 - seq64::sequence, 1089
- select_note_events
 - seq64::sequence, 1086
- select_trigger
 - seq64::perform, 755
 - seq64::sequence, 1080
- selected
 - seq64::trigger, 1127
- selected_api
 - seq64::rtmidi_info, 894, 895
- selected_trigger
 - seq64::perform, 713
- selected_trigger_count
 - seq64::sequence, 1054
- selected_trigger_end
 - seq64::sequence, 1084
- selected_trigger_start
 - seq64::sequence, 1084
- selecting
 - seq64::seqroll, 1022
- send_byte
 - seq64::midi_jack, 544
- send_message
 - seq64::midi_jack, 545
- seq
 - seq64::eventslots, 243
 - seq64::midi_alsa_info, 494
- Seq24PerfInput
 - seq64::Seq24PerfInput, 900
 - seq64::triggers, 1146
- Seq24SeqEventInput
 - seq64::Seq24SeqEventInput, 906
- seq64, 27
 - adjustment_dummy, 93
 - beat_log2, 61
 - beat_pow2, 60
 - bpm_from_bytes, 64
 - bpm_from_tempo_us, 64
 - build_details, 74
 - bussbyte, 44
 - c_backsequence, 111
 - c_bpmtag, 109
 - c_chord_number, 119
 - c_chord_size, 120

c_chord_table, 120
c_chord_table_text, 120
c_chord_text, 119
c_controller_names, 99
c_gap_A, 110
c_gap_B, 110
c_gap_C, 110
c_gap_D, 110
c_gap_E, 110
c_gap_F, 110
c_interval_text, 119
c_key_text, 119
c_max_busses, 120
c_max_instruments, 120
c_midi_control_bpm_dn, 113
c_midi_control_bpm_page_dn, 115
c_midi_control_bpm_page_up, 115
c_midi_control_bpm_up, 113
c_midi_control_mod_glearn, 114
c_midi_control_mod_gmute, 114
c_midi_control_mod_queue, 114
c_midi_control_mod_replace, 113
c_midi_control_mod_snapshot, 113
c_midi_control_play_ss, 114
c_midi_control_playback, 114
c_midi_control_quan_record, 115
c_midi_control_record, 115
c_midi_control_reset_seq, 115
c_midi_control_solo, 114
c_midi_control_song_record, 114
c_midi_control_ss_dn, 113
c_midi_control_ss_set, 115
c_midi_control_ss_up, 113
c_midi_control_thru, 115
c_midi_controls, 114
c_midi_controls_extended, 115
c_midi_track_ctrl, 112
c_midibus, 107
c_midibus_input_size, 107
c_midibus_output_size, 107
c_midibus_sysex_chunk, 107
c_midich, 109
c_midiclocks, 109
c_midictrl, 110
c_music_scales, 48
c_musickey, 111
c_musicscale, 111
c_mutegroups, 110
c_notes, 109
c_perf_bp_mes, 111
c_perf_bw, 111
c_reserved_1, 112
c_reserved_2, 112
c_scales_policy, 117
c_scales_text, 119
c_scales_transpose_dn, 118
c_scales_transpose_up, 117
c_seq_color, 112
c_seq_edit_mode, 112
c_status_oneshot, 117
c_status_queue, 116
c_status_replace, 116
c_status_snapshot, 116
c_tempo_map, 111
c_tempo_track, 112
c_timesig, 109
c_transpose, 111
c_triggers, 109
c_triggers_new, 109
casecompare, 91
check_daemonize, 75
choose_ppqn, 88
clamp, 98
clock_e, 47
clock_tick_duration_bogus, 66
clock_ticks_from_ppqn, 66
colorbyte, 44
create_jack_client, 83
create_lash_driver, 86
create_tempo_event, 76
current_date_time, 71
daemonize, 75
delete_lash_driver, 87
delta_time_us_to_ticks, 65
double_ticks_from_ppqn, 67
draw_type_t, 49
EVENT_AFTERTOUCH, 100
EVENT_ANY, 100
EVENT_CHANNEL_PRESSURE, 100
EVENT_CLEAR_CHAN_MASK, 106
EVENT_CONTROL_CHANGE, 100
EVENT_CTRL_BALANCE, 101
EVENT_CTRL_EXPRESSION, 101
EVENT_CTRL_PAN, 101
EVENT_CTRL_VOLUME, 101
EVENT_GET_CHAN_MASK, 106
EVENT_META_COPYRIGHT, 104
EVENT_META_CUE_POINT, 105
EVENT_META_END_OF_TRACK, 105
EVENT_META_ILLEGAL, 106
EVENT_META_INSTRUMENT, 104
EVENT_META_KEY_SIGNATURE, 106
EVENT_META_LYRIC, 104
EVENT_META_MARKER, 105
EVENT_META_MIDI_CHANNEL, 105
EVENT_META_MIDI_PORT, 105
EVENT_META_SEQ_NUMBER, 104
EVENT_META_SEQSPEC, 106
EVENT_META_SET_TEMPO, 105
EVENT_META_SMPTE_OFFSET, 105
EVENT_META_TEXT_EVENT, 104
EVENT_META_TIME_SIGNATURE, 105
EVENT_META_TRACK_NAME, 104
EVENT_MIDI_ACTIVE_SENSE, 103
EVENT_MIDI_CLOCK, 103
EVENT_MIDI_CONTINUE, 103

EVENT_MIDI_META, 104
 EVENT_MIDI_QUARTER_FRAME, 102
 EVENT_MIDI_REALTIME, 101
 EVENT_MIDI_RESET, 103
 EVENT_MIDI_SONG_F4, 102
 EVENT_MIDI_SONG_F5, 102
 EVENT_MIDI_SONG_F9, 103
 EVENT_MIDI_SONG_FD, 103
 EVENT_MIDI_SONG_POS, 102
 EVENT_MIDI_SONG_SELECT, 102
 EVENT_MIDI_START, 103
 EVENT_MIDI_STOP, 103
 EVENT_MIDI_SYSEX_CONTINUE, 102
 EVENT_MIDI_SYSEX_END, 102
 EVENT_MIDI_SYSEX, 101
 EVENT_MIDI_TUNE_SELECT, 102
 EVENT_NOTE_OFF, 100
 EVENT_NOTE_ON, 100
 EVENT_NULL_CHANNEL, 106
 EVENT_PITCH_WHEEL, 101
 EVENT_PROGRAM_CHANGE, 100
 EVENT_STATUS_BIT, 99
 EVENTS_ALL, 106
 EVENTS_UNSELECTED, 107
 edit_action_t, 50
 edit_mode_t, 49
 error_message, 91
 extract_bus_name, 70
 extract_port_name, 70
 extract_port_names, 70
 extract_timing_numbers, 52
 FF_RW_timeout, 98
 file_access, 76
 file_accessible, 77
 file_executable, 78
 file_exists, 76
 file_extension, 80
 file_extension_match, 81
 file_is_directory, 78
 file_readable, 77
 file_writable, 77
 font_render, 93
 g_midi_control_limit, 116
 g_rc_settings, 122
 g_user_settings, 122
 get_compound_option, 89
 get_current_directory, 79
 get_current_jack_position, 85
 get_full_path, 79
 gs_mainwid_pointer, 123
 gs_perfedir_pointer_0, 123
 gs_perfedir_pointer_1, 123
 help_check, 71
 info_message, 91
 input_thread_func, 88
 interaction_method_t, 48
 invalid_key, 86
 is_ctrl_key, 93–95
 is_ctrl_shift_key, 96
 is_drag_motion, 97
 is_left_drag, 97
 is_no_modifier, 94
 is_note_off_velocity, 76
 is_null_midipulse, 87
 is_shift_key, 94, 95
 is_super_key, 96
 jack_dummy_callback, 91
 jack_session_callback, 85
 jack_shutdown_callback, 81
 jack_timebase_callback, 81
 jack_transport_callback, 82
 keyval_name, 97
 keyval_normalize, 86
 lash_driver, 87
 log2_time_sig_value, 59
 long_options, 120
 loop_record_t, 50
 make_directory, 78
 make_section_name, 92
 measures_to_ticks, 67
 measurestring_to_pulses, 55
 message_concatenate, 90
 midi_api_name, 99
 midi_input_test, 99
 midi_measures_to_pulses, 56
 midi_probe, 99
 midibpm, 44
 midibyte, 44
 midilong, 44
 midipulse, 44
 midishort, 44
 midistring, 44
 millisleep, 87
 mouse_action_e, 50
 mute_group_handling_t, 48
 normalize_path, 79
 note_value_to_tempo, 62
 output_thread_func, 87
 parse_command_line_options, 74
 parse_log_option, 73
 parse_mute_groups, 72
 parse_o_options, 73
 parse_options_files, 71
 ppqn_is_valid, 63
 pulse_divide, 69
 pulse_length_us, 64
 pulses_per_measure, 67
 pulses_to_measurestring, 53
 pulses_to_midi_measures, 54
 pulses_to_string, 53
 pulses_to_timestring, 54, 55
 rc, 88
 reroute_stdio, 75
 rterror_callback, 45
 rtmidi_api, 51
 rtmidi_callback_t, 45

- s_arg_list, 121
- s_bitness, 122
- s_character_mapping, 122
- s_global_lash_driver, 122
- s_handlesize, 123
- s_help_1a, 121
- s_help_1b, 121
- s_help_2, 121
- s_help_3, 121
- s_help_4, 121
- s_help_5, 122
- s_jitter_amount, 123
- s_status_pairs, 122
- seq_app_name, 92
- seq_client_name, 92
- seq_event_type_t, 46
- seq_modifier_t, 46
- seq_scroll_direction_t, 47
- seq_version, 92
- set_current_directory, 90
- shorten_file_spec, 58
- show_jack_statuses, 84
- silence_jack_errors, 98
- silence_jack_info, 99
- strcasecompare, 80
- string_is_void, 59
- string_not_void, 58
- string_to_midibyte, 57
- string_to_pulses, 57
- strings_match, 59
- strip_quotes, 80
- swap, 51
- tempo_to_note_value, 62
- tempo_us_from_bpm, 63
- tempo_us_from_bytes, 61
- tempo_us_to_bytes, 61
- test_widget_click, 96
- ticks_to_delta_time_us, 65
- ticks_to_measures, 68
- timestring_to_pulses, 57, 89
- to_string, 76
- tokenize_string, 53
- undaemonize, 75
- update_main_sequences, 98
- update_perfedit_sequences, 98
- usr, 88
- versiontext, 120
- wave_func, 69
- wave_type_name, 52
- wave_type_t, 45
- write_options_files, 74
- zoom_power_of_2, 60
- seq64::FruityPerfInput, 263
 - ~FruityPerfInput, 264
 - activate_adding, 265
 - FruityPerfInput, 264
 - handle_motion_key, 265
 - on_button_press_event, 265
 - on_button_release_event, 265
 - on_left_button_pressed, 266
 - on_motion_notify_event, 266
 - on_right_button_pressed, 267
 - update_mouse_pointer, 265
- seq64::FruitySeqEventInput, 267
 - FruitySeqEventInput, 269
 - m_is_drag_pasting, 271
 - m_is_drag_pasting_start, 271
 - m_justselected_one, 271
 - on_button_press_event, 269
 - on_button_release_event, 270
 - on_motion_notify_event, 271
 - update_mouse_pointer, 269
- seq64::FruitySeqRollInput, 272
 - FruitySeqRollInput, 273
 - m_can_add, 275
 - m_drag_paste_start_pos, 275
 - m_erase_painting, 275
 - on_button_press_event, 273
 - on_button_release_event, 274
 - on_motion_notify_event, 274
 - update_mouse_pointer, 273
- seq64::Seq24PerfInput, 899
 - ~Seq24PerfInput, 901
 - activate_adding, 901
 - check_handles, 902
 - handle_motion_key, 901
 - m_effective_tick, 904
 - on_button_press_event, 902
 - on_button_release_event, 903
 - on_motion_notify_event, 904
 - Seq24PerfInput, 900
- seq64::Seq24SeqEventInput, 905
 - on_button_press_event, 906
 - on_button_release_event, 906
 - on_motion_notify_event, 907
 - Seq24SeqEventInput, 906
- seq64::automutex, 125
 - ~automutex, 126
 - automutex, 125, 126
 - m_safety_mutex, 126
 - operator=, 126
- seq64::busarray, 127
 - ~busarray, 128
 - add, 128, 129
 - bus, 130
 - busarray, 128
 - clock, 130
 - continue_from, 130
 - count, 129
 - get_clock, 132
 - get_input, 134
 - get_midi_bus_name, 132
 - get_midi_event, 135
 - init_clock, 130
 - initialize, 129
 - is_system_port, 134

- m_container, 136
- play, 131
- poll_for_midi, 135
- port_exit, 133
- print, 133
- replacement_port, 135
- set_all_clocks, 132
- set_all_inputs, 134
- set_clock, 131
- set_input, 133
- start, 130
- stop, 130
- sysex, 131
- seq64::businfo, 136
 - ~businfo, 138
 - activate, 140
 - active, 139
 - bus, 139, 140
 - busarray, 142
 - businfo, 137, 138
 - clock, 141
 - continue_from, 141
 - deactivate, 140
 - init_clock, 140, 141
 - init_input, 140
 - initialize, 139
 - initialized, 139
 - m_active, 142
 - m_bus, 142
 - m_init_clock, 142
 - m_init_input, 142
 - m_initialized, 142
 - print, 141
 - remove, 138
 - start, 141
 - stop, 141
 - sysex, 141
- seq64::click, 143
 - button, 146
 - click, 144
 - is_left, 145
 - is_middle, 145
 - is_press, 145
 - is_right, 145
 - m_button, 147
 - m_is_press, 146
 - m_modifier, 147
 - m_x, 147
 - m_y, 147
 - mod_control, 146
 - mod_control_shift, 146
 - mod_super, 146
 - modifier, 146
 - operator=, 145
 - x, 145
 - y, 146
- seq64::condition_var, 147
 - condition_var, 149
 - m_cond, 149
 - signal, 149
 - sm_cond, 149
 - wait, 149
- seq64::configfile, 150
 - ~configfile, 151
 - at_section_start, 152
 - configfile, 151
 - get_error_message, 153
 - line_after, 152
 - m_d, 154
 - m_error_message, 153
 - m_line, 154
 - m_name, 153
 - next_data_line, 151
 - parse, 153
 - set_error_message, 153
 - write, 153
- seq64::editable_event, 154
 - ~editable_event, 160
 - analyze, 166
 - category, 162, 163
 - category_string, 162
 - category_t, 158
 - channel_string, 166
 - data_string, 166
 - editable_event, 159, 160
 - ex_data_string, 165
 - format_timestamp, 165
 - m_category, 168
 - m_format_timestamp, 168
 - m_name_category, 168
 - m_name_channel, 169
 - m_name_data, 169
 - m_name_meta, 169
 - m_name_seqspect, 169
 - m_name_status, 168
 - m_name_timestamp, 168
 - m_parent, 168
 - meta_event_length, 161
 - meta_string, 165
 - name_to_value, 161
 - operator=, 162
 - parent, 162
 - seqspec_string, 165
 - set_status_from_string, 164
 - sm_category_arrays, 167
 - sm_category_names, 167
 - sm_channel_event_names, 167
 - sm_meta_event_names, 167
 - sm_meta_lengths, 168
 - sm_prop_event_names, 167
 - sm_system_event_names, 167
 - status_string, 165
 - stock_event_string, 165
 - time_as_measures, 164
 - time_as_minutes, 164
 - time_as_pulses, 164

- timestamp, 163
- timestamp_format_t, 159
- timestamp_string, 163
- value_to_name, 161
- seq64::editable_event::meta_length_t, 484
 - event_length, 484
 - event_value, 484
- seq64::editable_event::name_value_t, 641
 - event_name, 642
 - event_value, 642
- seq64::editable_events, 169
 - ~editable_events, 174
 - add, 177
 - begin, 175, 176
 - clear, 178
 - const_iterator, 172
 - const_reverse_iterator, 172
 - count, 176
 - current_event, 178
 - dref, 176
 - editable_events, 172–174
 - end, 176
 - Events, 172
 - events, 175
 - EventsPair, 172
 - eventslots, 179
 - get_length, 177
 - is_valid_iterator, 178
 - iterator, 172
 - Key, 171
 - load_events, 175
 - m_current_event, 179
 - m_events, 179
 - m_midi_parameters, 179
 - m_sequence, 179
 - operator=, 174
 - print, 178
 - remove, 178
 - replace, 178
 - reverse_iterator, 172
 - save_events, 175
 - sort, 178
 - string_to_pulses, 175
 - timing, 174
- seq64::event, 180
 - ~event, 185
 - adjust_note_off, 201
 - append_meta_data, 195, 196
 - append_sysex, 195
 - cc_match, 193
 - check_channel, 187
 - clear_link, 198
 - data, 199
 - decrement_data1, 194
 - decrement_data2, 195
 - event, 185
 - get_channel, 187
 - get_data, 194
 - get_linked, 198
 - get_note, 199
 - get_note_velocity, 200
 - get_rank, 203
 - get_status, 193
 - get_sysex, 197
 - get_sysex_size, 197
 - get_timestamp, 187
 - increment_data1, 194
 - increment_data2, 194
 - is_channel_msg, 188
 - is_desired_cc_or_not_cc, 190
 - is_ex_data, 202
 - is_key_signature, 203
 - is_linked, 198
 - is_marked, 199
 - is_meta, 202
 - is_note, 201
 - is_note_msg, 189
 - is_note_off, 200
 - is_note_off_recorded, 201
 - is_note_on, 200
 - is_one_byte, 201
 - is_one_byte_msg, 188
 - is_painted, 198
 - is_selected, 199
 - is_strict_note_msg, 189
 - is_sysex, 202
 - is_sysex_msg, 189
 - is_tempo, 202
 - is_time_signature, 203
 - is_two_byte_msg, 188
 - is_two_bytes, 202
 - link, 197
 - m_channel, 204
 - m_data, 204
 - m_has_link, 204
 - m_linked, 204
 - m_marked, 205
 - m_painted, 205
 - m_selected, 205
 - m_status, 204
 - m_sysex, 204
 - m_timestamp, 203
 - make_clock, 199
 - mark, 198
 - mod_timestamp, 190
 - non_cc_match, 193
 - operator<, 186
 - operator=, 186
 - paint, 198
 - print, 203
 - restart_sysex, 196
 - select, 199
 - set_channel, 192
 - set_data, 193
 - set_meta_status, 192
 - set_note, 199

- set_note_velocity, 200
- set_status, 191
- set_status_keep_channel, 192
- set_sysex, 196
- set_sysex_size, 197
- set_tempo, 202
- set_timestamp, 187
- SysexContainer, 185
- tempo, 202
- transpose_note, 200
- unmark, 199
- unpaint, 198
- unselect, 199
- seq64::event_list, 207
 - ~event_list, 211
 - add, 212
 - any_selected_notes, 218
 - append, 213
 - begin, 212
 - clear, 214
 - clear_links, 216
 - clear_tempo_links, 217
 - const_iterator, 210
 - const_reverse_iterator, 210
 - count, 212
 - count_selected_events, 218
 - count_selected_notes, 218
 - dref, 215
 - editable_events, 219
 - empty, 212
 - end, 212
 - event_list, 211
 - Events, 210
 - events, 219
 - EventsPair, 210
 - get_length, 212
 - has_tempo, 214
 - has_time_signature, 214
 - is_modified, 213
 - iterator, 210
 - link_new, 216
 - link_tempos, 216
 - m_events, 220
 - m_has_tempo, 220
 - m_has_time_signature, 220
 - m_is_modified, 220
 - mark_all, 217
 - mark_out_of_range, 217
 - mark_selected, 217
 - merge, 214
 - midifile, 219
 - operator=, 211
 - print, 219
 - push_back, 213
 - remove, 214
 - remove_marked, 218
 - reverse_iterator, 210
 - select_all, 219
 - sequence, 220
 - sort, 215
 - unmark_all, 217
 - unmodify, 214
 - unpaint_all, 218
 - unselect_all, 219
 - verify_and_link, 216
- seq64::event_list::event_key, 205
 - event_key, 206
 - m_rank, 207
 - m_timestamp, 207
 - operator<, 206
- seq64::eventedit, 221
 - ~eventedit, 226
 - change_focus, 229
 - close_out, 230
 - enqueue_draw, 226
 - eventedit, 224
 - eventslots, 232
 - handle_cancel, 231
 - handle_close, 230
 - handle_delete, 230
 - handle_insert, 230
 - handle_modify, 230
 - handle_save, 230
 - m_bottbox, 234
 - m_button_cancel, 234
 - m_button_del, 234
 - m_button_ins, 234
 - m_button_modify, 234
 - m_button_save, 234
 - m_editbox, 233
 - m_entry_ev_data_0, 236
 - m_entry_ev_data_1, 236
 - m_entry_ev_name, 236
 - m_entry_ev_timestamp, 236
 - m_eventslots, 233
 - m_have_focus, 237
 - m_htopbox, 233
 - m_label_category, 236
 - m_label_channel, 235
 - m_label_ev_count, 235
 - m_label_modified, 235
 - m_label_ppqn, 235
 - m_label_right, 236
 - m_label_seq_length, 235
 - m_label_seq_name, 235
 - m_label_spacer, 235
 - m_label_time_fmt, 236
 - m_label_time_sig, 235
 - m_optsbox, 234
 - m_rightbox, 234
 - m_seq, 236
 - m_showbox, 233
 - m_table, 233
 - m_vadjust, 233
 - m_vscroll, 233
 - on_delete_event, 232

- on_focus_in_event, [231](#)
- on_focus_out_event, [231](#)
- on_key_press_event, [232](#)
- on_realize, [231](#)
- on_set_focus, [231](#)
- perf_modify, [228](#)
- set_dirty, [228](#)
- set_event_category, [227](#)
- set_event_data_0, [228](#)
- set_event_data_1, [228](#)
- set_event_name, [228](#)
- set_event_timestamp, [227](#)
- set_seq_count, [227](#)
- set_seq_length, [227](#)
- set_seq_ppqn, [227](#)
- set_seq_time_sig, [226](#)
- set_seq_title, [226](#)
- v_adjustment, [229](#)
- seq64::eventslots, [237](#)
 - ~eventslots, [242](#)
 - calculate_measures, [251](#)
 - change_vert, [249](#)
 - convert_y, [248](#)
 - count, [242](#)
 - current_index, [243](#)
 - decrement_bottom, [251](#)
 - decrement_current, [250](#)
 - decrement_top, [250](#)
 - delete_current_event, [245](#)
 - draw_event, [248](#)
 - draw_events, [249](#)
 - enqueue_draw, [248](#)
 - event_count, [242](#)
 - eventedit, [254](#)
 - eventslots, [242](#)
 - get_length, [242](#)
 - increment_bottom, [251](#)
 - increment_current, [251](#)
 - increment_top, [250](#)
 - insert_event, [244](#), [245](#)
 - line_count, [243](#)
 - line_increment, [243](#)
 - line_maximum, [243](#)
 - load_events, [243](#)
 - m_bottom_iterator, [256](#)
 - m_char_w, [255](#)
 - m_current_index, [256](#)
 - m_current_iterator, [257](#)
 - m_event_container, [254](#)
 - m_event_count, [255](#)
 - m_last_max_timestamp, [255](#)
 - m_line_count, [256](#)
 - m_line_maximum, [256](#)
 - m_line_overlap, [256](#)
 - m_measures, [255](#)
 - m_pager_index, [257](#)
 - m_parent, [254](#)
 - m_seq, [254](#)
 - m_setbox_w, [255](#)
 - m_slots_chars, [254](#)
 - m_slots_x, [255](#)
 - m_slots_y, [255](#)
 - m_top_index, [256](#)
 - m_top_iterator, [256](#)
 - modify_current_event, [246](#)
 - on_button_press_event, [252](#)
 - on_button_release_event, [252](#)
 - on_expose_event, [252](#)
 - on_focus_in_event, [252](#)
 - on_focus_out_event, [252](#)
 - on_frame_down, [253](#)
 - on_frame_end, [254](#)
 - on_frame_home, [253](#)
 - on_frame_up, [253](#)
 - on_move_down, [253](#)
 - on_move_up, [253](#)
 - on_realize, [252](#)
 - on_scroll_event, [253](#)
 - on_size_allocate, [253](#)
 - page_movement, [249](#)
 - page_topper, [250](#)
 - pager_index, [243](#)
 - save_events, [247](#)
 - select_event, [247](#)
 - seq, [243](#)
 - set_current_event, [244](#)
 - set_text, [248](#)
 - top_index, [243](#)
- seq64::font, [257](#)
 - char_height, [260](#)
 - char_width, [260](#)
 - Color, [258](#)
 - font, [259](#)
 - init, [259](#)
 - m_b_on_c_pixmap, [262](#)
 - m_b_on_y_pixmap, [262](#)
 - m_black_pixmap, [261](#)
 - m_c_on_b_pixmap, [262](#)
 - m_cell_h, [260](#)
 - m_cell_w, [260](#)
 - m_clip_mask, [262](#)
 - m_font_h, [261](#)
 - m_font_w, [261](#)
 - m_offset, [261](#)
 - m_padded_h, [261](#)
 - m_pixmap, [261](#)
 - m_use_new_font, [260](#)
 - m_white_pixmap, [261](#)
 - m_y_on_b_pixmap, [262](#)
 - padded_height, [260](#)
 - render_string_on_drawable, [259](#)
- seq64::gui_assistant, [276](#)
 - ~gui_assistant, [277](#)
 - gui_assistant, [277](#)
 - jack_idle_connect, [277](#)
 - keys, [278](#)

- lash_timeout_connect, 278
- m_keys_perform, 278
- quit, 277
- seq64::gui_assistant_gtk2, 279
 - ~gui_assistant_gtk2, 280
 - gui_assistant_gtk2, 280
 - jack_idle_connect, 280
 - lash_timeout_connect, 280
 - quit, 280
 - sm_internal_keys, 281
- seq64::gui_drawingarea_gtk2, 281
 - ~gui_drawingarea_gtk2, 285
 - clear_window, 287
 - current_x, 286
 - current_y, 286
 - draw_drawable, 296
 - draw_line, 288–290
 - draw_line_on_pixmap, 288, 289
 - draw_normal_rectangle_on_pixmap, 295
 - draw_rectangle, 291–294
 - draw_rectangle_on_pixmap, 294, 295
 - drop_x, 286
 - drop_y, 286
 - force_draw, 287
 - get_color, 286
 - get_sequence_color, 286
 - gtk_drawarea_init, 297
 - gui_drawingarea_gtk2, 284, 285
 - height, 286
 - m_background, 298
 - m_current_x, 299
 - m_current_y, 299
 - m_drop_x, 299
 - m_drop_y, 300
 - m_foreground, 299
 - m_gc, 298
 - m_hadjust, 298
 - m_mainperf, 299
 - m_pixmap, 298
 - m_vadjust, 298
 - m_window, 298
 - m_window_x, 299
 - m_window_y, 299
 - on_realize, 298
 - operator=, 285
 - perf, 287
 - render_string, 291
 - render_string_on_pixmap, 291
 - scroll_hadjust, 296
 - scroll_hset, 297
 - scroll_vadjust, 296
 - scroll_vset, 297
 - set_current_drop_x, 297
 - set_current_drop_y, 297
 - set_line, 287
 - width, 285
 - window_x, 285
 - window_y, 286
- seq64::gui_palette_gtk2, 300
 - ~gui_palette_gtk2, 305
 - bg_color, 310
 - black, 307
 - black_key_paint, 310
 - black_paint, 309
 - blue, 309
 - Color, 304
 - dark_blue, 308
 - dark_cyan, 308
 - dark_green, 308
 - dark_grey_paint, 308
 - dark_magenta, 308
 - dark_orange, 308
 - dark_red, 307
 - fg_color, 310, 311
 - get_color, 305
 - get_color_ex, 305
 - green, 309
 - grey_paint, 308
 - gui_palette_gtk2, 305
 - initialize, 305
 - is_inverse, 307
 - light_grey_paint, 309
 - line_color, 307
 - load_inverse_palette, 307
 - m_bg_color, 315
 - m_black, 311
 - m_blk_key_paint, 314
 - m_blk_paint, 314
 - m_blue, 312
 - m_cyan, 312
 - m_dk_black, 312
 - m_dk_blue, 313
 - m_dk_cyan, 313
 - m_dk_green, 312
 - m_dk_grey, 314
 - m_dk_grey_paint, 314
 - m_dk_magenta, 313
 - m_dk_orange, 313
 - m_dk_pink, 314
 - m_dk_red, 312
 - m_dk_white, 313
 - m_dk_yellow, 312
 - m_fg_color, 315
 - m_green, 311
 - m_grey, 313
 - m_grey_paint, 314
 - m_is_inverse, 311
 - m_line_color, 315
 - m_lt_grey_paint, 314
 - m_magenta, 312
 - m_orange, 313
 - m_palette, 311
 - m_pink, 313
 - m_progress_color, 315
 - m_red, 311
 - m_sel_paint, 315

- m_tempo_paint, 315
- m_white, 312
- m_wht_key_paint, 315
- m_wht_paint, 314
- m_yellow, 311
- magenta, 309
- orange, 309
- progress_color, 307
- red, 309
- sel_paint, 310
- tempo_paint, 310
- white, 308
- white_key_paint, 310
- white_paint, 310
- yellow, 309
- seq64::gui_window_gtk2, 316
 - ~gui_window_gtk2, 318
 - gui_window_gtk2, 317
 - is_realized, 318
 - m_is_realized, 320
 - m_mainperf, 320
 - m_redraw_period_ms, 320
 - m_window_x, 320
 - m_window_y, 320
 - on_realize, 320
 - perf, 318
 - quit, 318
 - redraw_period_ms, 318
 - scroll_hadjust, 318
 - scroll_hset, 319
 - scroll_vadjust, 319
 - scroll_vset, 319
- seq64::jack_assistant, 320
 - ~jack_assistant, 325
 - activate, 330
 - client, 334
 - client_name, 335
 - client_open, 335
 - client_uuid, 335
 - deinit, 329
 - error_message, 337
 - get_beat_width, 327
 - get_beats_per_measure, 327
 - get_beats_per_minute, 328
 - get_current_jack_position, 340
 - get_follow_transport, 334
 - get_jack_client_info, 336
 - get_jack_mode, 333
 - get_jack_pos, 333
 - get_jack_stop_tick, 333
 - get_jack_tick, 333
 - get_ppqn, 327
 - get_state_name, 326
 - info_message, 337
 - init, 328
 - is_master, 327
 - is_running, 327
 - jack_assistant, 324
 - jack_frame_rate, 333
 - jack_session_callback, 340
 - jack_shutdown_callback, 338
 - jack_timebase_callback, 338
 - jack_transport_callback, 337
 - m_beat_width, 343
 - m_beats_per_measure, 343
 - m_beats_per_minute, 343
 - m_follow_transport, 343
 - m_jack_client, 341
 - m_jack_client_name, 341
 - m_jack_client_uuid, 341
 - m_jack_frame_current, 341
 - m_jack_frame_last, 341
 - m_jack_frame_rate, 342
 - m_jack_master, 342
 - m_jack_parent, 341
 - m_jack_pos, 341
 - m_jack_running, 342
 - m_jack_stop_tick, 343
 - m_jack_tick, 342
 - m_jack_transport_state, 341
 - m_jack_transport_state_last, 342
 - m_session_ev, 342
 - m_ppqn, 343
 - m_toggle_jack, 342
 - output, 332
 - parent, 326
 - position, 330
 - session_event, 329
 - set_beat_width, 327
 - set_beats_per_measure, 327
 - set_beats_per_minute, 328
 - set_follow_transport, 334
 - set_jack_mode, 333
 - set_jack_running, 335
 - set_jack_stop_tick, 333
 - set_ppqn, 332
 - set_start_from_perfedit, 334
 - show_position, 325
 - sm_status_pairs, 340
 - song_start_mode, 334
 - start, 330
 - stop, 330
 - sync, 336
 - tick_multiplier, 335
 - toggle_follow_transport, 334
 - toggle_jack_mode, 333
 - toggle_song_start_mode, 334
 - transport_not_starting, 328
 - transport_state, 328
- seq64::jack_scratchpad, 343
 - js_clock_tick, 344
 - js_current_tick, 344
 - js_delta_tick_frac, 346
 - js_dumping, 345
 - js_init_clock, 345
 - js_jack_stopped, 345

- js_looping, 345
- js_playback_mode, 345
- js_ticks_converted, 345
- js_ticks_converted_last, 345
- js_ticks_delta, 345
- js_total_tick, 344
- seq64::jack_status_pair_t, 346
 - jf_bit, 346
 - jf_meaning, 346
- seq64::keybindentry, 346
 - keybindentry, 348
 - m_key, 349
 - m_perf, 349
 - m_slot, 349
 - m_type, 349
 - on_key_press_event, 348
 - options, 349
 - set, 348
 - type, 347
- seq64::keys_perform, 350
 - ~keys_perform, 357
 - at_bpm_dn, 372
 - at_bpm_up, 372
 - at_event_edit, 376
 - at_fast_forward, 375
 - at_follow_transport, 374
 - at_group_learn, 374
 - at_group_off, 374
 - at_group_on, 373
 - at_keep_queue, 373
 - at_menu_mode, 374
 - at_oneshot_queue, 375
 - at_pattern_edit, 375
 - at_pattern_shift, 376
 - at_pause, 374
 - at_pointer_position, 375
 - at_queue, 373
 - at_replace, 372
 - at_rewind, 375
 - at_screenset_dn, 373
 - at_screenset_up, 373
 - at_set_playing_screenset, 373
 - at_show_ui_sequence_key, 376
 - at_show_ui_sequence_number, 376
 - at_snapshot_1, 373
 - at_snapshot_2, 373
 - at_song_mode, 374
 - at_song_record, 375
 - at_start, 374
 - at_stop, 376
 - at_tap_bpm, 375
 - at_toggle_jack, 374
 - at_toggle_mutes, 375
 - bpm_dn, 358
 - bpm_up, 358
 - event_edit, 364
 - fast_forward, 366
 - follow_transport, 365, 366
 - get_key_count, 369
 - get_key_events, 369
 - get_key_events_rev, 369
 - get_key_groups, 369
 - get_key_groups_rev, 369
 - get_keys, 358
 - group_learn, 362
 - group_max, 372
 - group_off, 362
 - group_on, 362
 - keep_queue, 359
 - key_name, 371
 - keys_perform, 357
 - lookup_keyevent_key, 369
 - lookup_keyevent_seq, 370
 - lookup_keygroup_group, 370
 - lookup_keygroup_key, 370
 - m_group_max, 378
 - m_key_bpm_dn, 378
 - m_key_bpm_up, 378
 - m_key_event_edit, 381
 - m_key_events, 377
 - m_key_events_rev, 378
 - m_key_fast_forward, 381
 - m_key_follow_transport, 380
 - m_key_group_learn, 380
 - m_key_group_off, 379
 - m_key_group_on, 379
 - m_key_groups, 377
 - m_key_groups_rev, 378
 - m_key_keep_queue, 379
 - m_key_menu_mode, 380
 - m_key_oneshot_queue, 382
 - m_key_pattern_edit, 381
 - m_key_pattern_shift, 381
 - m_key_pause, 380
 - m_key_pointer_position, 381
 - m_key_queue, 378
 - m_key_replace, 378
 - m_key_rewind, 380
 - m_key_screenset_dn, 379
 - m_key_screenset_up, 379
 - m_key_set_playing_screenset, 379
 - m_key_show_ui_sequence_key, 377
 - m_key_show_ui_sequence_number, 377
 - m_key_snapshot_1, 379
 - m_key_snapshot_2, 379
 - m_key_song_mode, 380
 - m_key_song_record, 382
 - m_key_start, 380
 - m_key_stop, 381
 - m_key_tap_bpm, 381
 - m_key_toggle_jack, 380
 - m_key_toggle_mutes, 381
 - menu_mode, 365
 - oneshot_queue, 368
 - options, 377
 - optionsfile, 377

- pattern_edit, 363, 364
- pattern_shift, 364
- pause, 363
- perform, 377
- pointer_position, 366
- queue, 359
- replace, 359
- RevSlotMap, 357
- rewind, 366
- screenset_dn, 361
- screenset_up, 360, 361
- set_all_key_events, 371
- set_all_key_groups, 371
- set_basic_key_events, 376
- set_basic_key_groups, 376
- set_key_event, 371
- set_key_group, 372
- set_keys, 357
- set_playing_screenset, 361
- show_ui_sequence_key, 368
- show_ui_sequence_number, 368, 369
- SlotMap, 357
- snapshot_1, 360
- snapshot_2, 360
- song_mode, 365
- song_record, 367, 368
- start, 363
- stop, 365
- tap_bpm, 367
- toggle_jack, 367
- toggle_mutes, 367
- seq64::keys_perform_gtk2, 382
 - ~keys_perform_gtk2, 384
 - key_name, 384
 - keys_perform_gtk2, 384
 - set_all_key_events, 385
 - set_all_key_groups, 385
- seq64::keys_perform_transfer, 385
 - kpt_bpm_dn, 386
 - kpt_bpm_up, 386
 - kpt_event_edit, 388
 - kpt_fast_forward, 389
 - kpt_follow_transport, 389
 - kpt_group_learn, 387
 - kpt_group_off, 387
 - kpt_group_on, 387
 - kpt_keep_queue, 387
 - kpt_menu_mode, 389
 - kpt_oneshot_queue, 390
 - kpt_pattern_edit, 388
 - kpt_pattern_shift, 388
 - kpt_pause, 389
 - kpt_pointer_position, 389
 - kpt_queue, 387
 - kpt_replace, 387
 - kpt_rewind, 389
 - kpt_screenset_dn, 386
 - kpt_screenset_up, 386
 - kpt_set_playing_screenset, 386
 - kpt_show_ui_sequence_key, 388
 - kpt_show_ui_sequence_number, 388
 - kpt_snapshot_1, 387
 - kpt_snapshot_2, 387
 - kpt_song_mode, 389
 - kpt_song_record, 390
 - kpt_start, 388
 - kpt_stop, 388
 - kpt_tap_bpm, 388
 - kpt_toggle_jack, 389
 - kpt_toggle_mutes, 390
- seq64::keystroke, 390
 - is, 394
 - is_delete, 394
 - is_letter, 392
 - is_press, 392
 - key, 394
 - keystroke, 391, 392
 - m_is_press, 396
 - m_key, 396
 - m_modifier, 396
 - mod_control, 395
 - mod_control_shift, 395
 - mod_super, 395
 - modifier, 395
 - operator=, 392
 - shift_lock, 395
- seq64::lash, 396
 - handle_config, 398
 - handle_event, 398
 - init, 398
 - lash, 397
 - m_client, 399
 - m_is_lash_supported, 399
 - m_lash_args, 399
 - m_perform, 399
 - process_events, 398
 - set_alsa_client_id, 397
 - start, 397
- seq64::lfownd, 400
 - ~lfownd, 402
 - lfownd, 402
 - m_hbox, 403
 - m_phase, 404
 - m_range, 404
 - m_scale_phase, 403
 - m_scale_range, 403
 - m_scale_speed, 403
 - m_scale_value, 403
 - m_scale_wave, 403
 - m_seq, 403
 - m_seqdata, 403
 - m_speed, 404
 - m_value, 404
 - m_wave, 404
 - m_wave_name, 404
 - on_focus_out_event, 402

- scale_lfo_change, 402
- toggle_visible, 402
- seq64::maintime, 405
 - ~maintime, 407
 - idle_progress, 407
 - m_bar_width, 408
 - m_beat_width, 408
 - m_box_height, 409
 - m_box_less_pill, 409
 - m_box_width, 408
 - m_flash_height, 409
 - m_flash_width, 409
 - m_flash_x, 409
 - m_pill_width, 408
 - m_ppqn, 409
 - m_tick, 409
 - maintime, 407
 - mainwnd, 408
 - on_expose_event, 408
 - operator=, 407
- seq64::mainwid, 410
 - ~mainwid, 414
 - calculate_base_sizes, 418
 - draw_marker_on_sequence, 416
 - draw_pixmap_on_window, 415
 - draw_sequence_on_pixmap, 417
 - draw_sequence_pixmap_on_window, 418
 - draw_sequences_on_pixmap, 418
 - fill_background_window, 415
 - log_screenset, 414
 - m_armed_progress_color, 422
 - m_button_down, 422
 - m_last_tick_x, 423
 - m_mainwid_border_x, 424
 - m_mainwid_border_y, 424
 - m_mainwid_spacing, 424
 - m_mainwid_x, 423
 - m_mainwid_y, 424
 - m_mainwnd_cols, 423
 - m_mainwnd_rows, 423
 - m_max_sets, 424
 - m_moving, 422
 - m_moving_seq, 422
 - m_old_seq, 422
 - m_progress_height, 425
 - m_screenset, 422
 - m_screenset_offset, 425
 - m_screenset_slots, 424
 - m_seqarea_seq_x, 423
 - m_seqarea_seq_y, 423
 - m_seqarea_x, 423
 - m_seqarea_y, 423
 - m_text_size_x, 424
 - m_text_size_y, 424
 - mainwid, 413
 - mainwnd, 421
 - nominal_height, 415
 - nominal_width, 415
 - on_button_press_event, 419
 - on_button_release_event, 420
 - on_expose_event, 419
 - on_focus_in_event, 421
 - on_focus_out_event, 421
 - on_motion_notify_event, 420
 - on_realize, 419
 - redraw, 415
 - reset, 414
 - select_fg_bg_colors, 419
 - seq_from_xy, 418
 - seq_set_and_edit, 416
 - seq_set_and_eventedit, 416
 - set_screenset, 414
 - timeout, 418
 - update_mainwid_sequences, 422
 - update_markers, 416
 - update_sequences_on_window, 415
 - valid_sequence, 417
- seq64::mainwnd, 425
 - ~mainwnd, 434
 - about_dialog, 445
 - adj_callback_bpm, 436
 - adj_callback_ss, 436
 - adj_callback_wid, 435
 - apply_song_transpose, 442
 - build_info_dialog, 445
 - choose_file, 447
 - clear_mute_groups, 442
 - debug_text, 435
 - edit_callback_notepad, 437
 - edit_field_has_focus, 447
 - enregister_perfedits, 442
 - file_exit, 446
 - file_import_dialog, 444
 - file_new, 443
 - file_open, 443
 - file_save, 444
 - file_save_as, 445
 - handle_signal, 435
 - independent, 436
 - install_signal_handlers, 447
 - is_save, 447
 - jack_dialog, 445
 - learn_toggle, 441
 - load_recent_file, 443
 - m_adjust_bpm, 456
 - m_adjust_load_offset, 456
 - m_adjust_ss, 453
 - m_button_jack, 455
 - m_button_learn, 454
 - m_button_panic, 454
 - m_button_perfedits, 455
 - m_button_play, 454
 - m_button_queue, 456
 - m_button_song_record, 455
 - m_button_song_snap, 455
 - m_button_stop, 454

m_button_tempo_log, 454
m_button_tempo_record, 454
m_button_time_type, 456
m_call_seq_edit, 457
m_call_seq_eventedit, 457
m_call_seq_shift, 457
m_current_screensset, 453
m_entry_notes, 457
m_image_play, 454
m_is_running, 457
m_is_snap_recording, 455
m_is_song_recording, 455
m_is_tempo_recording, 455
m_main_cursor, 454
m_main_time, 453
m_main_wid, 452
m_mainwid_adjustors, 451
m_mainwid_blocks, 452
m_mainwid_columns, 452
m_mainwid_count, 452
m_mainwid_frames, 451
m_mainwid_grid, 451
m_mainwid_independent, 452
m_mainwid_rows, 452
m_mainwid_spinners, 452
m_menu_edit, 451
m_menu_file, 450
m_menu_help, 451
m_menu_mode, 457
m_menu_recent, 450
m_menu_view, 451
m_menubar, 450
m_options, 453
m_perf_edit, 453
m_perf_edit_2, 453
m_ppqn, 451
m_spinbutton_bpm, 456
m_spinbutton_load_offset, 456
m_spinbutton_ss, 453
m_status_label, 451
m_tick_time, 456
m_tick_time_as_bbt, 456
m_timeout_connect, 457
mainwnd, 433
multi_wid, 435
need_set_spinner, 436
new_file, 446
new_open_error_dialog, 445
on_delete_event, 449
on_grouplearnchange, 450
on_key_press_event, 449
on_key_release_event, 449
on_realize, 449
open_file, 434
open_performance_edit, 441
open_performance_edit_2, 442
options_dialog, 444
panic, 441
pause_playing, 439
populate_menu_edit, 448
populate_menu_file, 448
populate_menu_help, 448
populate_menu_view, 448
ppqn, 434
query_save_changes, 445
queue_it, 440
rc_error_dialog, 434
reload_mute_groups, 442
reset, 437
reset_window, 438
save_file, 447
SaveOption, 433
sequence_key, 442
set_play_image, 438
set_screensset, 439
set_song_mute, 444
set_song_playback, 441
set_song_record, 440
set_sONGLIVE_image, 438
set_status_text, 448
set_wid_label, 437
signal_action, 447
sm_sigpipe, 450
sm_widmax, 450
song_record_snap, 441
spinner_max, 442
start_playing, 438
stop_playing, 439
tempo_log, 440
timer_callback, 439
toLower, 443
toggle_playing, 439
toggle_song_record, 440
toggle_song_snap, 441
toggle_tempo_record, 440
toggle_time_format, 440
update_markers, 437
update_recent_files_menu, 443
update_screensset, 437
update_window_title, 443
wid_box_to_slot, 444
wid_slot_to_box, 444
seq64::mastermidibase, 458
 ~mastermidibase, 463
 activate, 474
 announce_bus_exists, 464
 api_clock, 475
 api_continue_from, 474
 api_flush, 475
 api_get_midi_event, 476
 api_init, 474
 api_init_clock, 474
 api_poll_for_midi, 476
 api_port_start, 475
 api_set_beats_per_minute, 475
 api_set_ppqn, 475

- api_start, [474](#)
- api_stop, [475](#)
- clock, [473](#)
- continue_from, [466](#)
- dump_midi_input, [469](#)
- emit_clock, [467](#)
- filter_by_channel, [464](#)
- flush, [468](#)
- get_beats_per_minute, [464](#)
- get_bpm, [465](#)
- get_clock, [472](#)
- get_input, [471](#)
- get_midi_event, [470](#)
- get_midi_in_bus_name, [469](#)
- get_midi_out_bus_name, [469](#)
- get_num_in_buses, [464](#)
- get_num_out_buses, [464](#)
- get_port_statuses, [473](#)
- get_ppqn, [465](#)
- get_sequence, [465](#)
- init, [463](#)
- init_clock, [467](#)
- input, [474](#)
- is_dumping, [465](#)
- is_input_system_port, [472](#)
- is_more_input, [470](#)
- m_beats_per_minute, [478](#)
- m_bus_announce, [477](#)
- m_dumping_input, [479](#)
- m_filter_by_channel, [479](#)
- m_inbus_array, [478](#)
- m_master_clocks, [478](#)
- m_master_inputs, [478](#)
- m_max_busses, [477](#)
- m_mutex, [479](#)
- m_outbus_array, [478](#)
- m_ppqn, [478](#)
- m_queue, [478](#)
- m_seq, [479](#)
- m_vector_sequence, [479](#)
- mastermidibase, [463](#)
- midi_alsa_info, [477](#)
- panic, [468](#)
- perform, [477](#)
- play, [466](#)
- poll_for_midi, [470](#)
- port_exit, [466](#)
- port_start, [465](#)
- print, [468](#)
- save_clock, [476](#)
- save_input, [477](#)
- set_beats_per_minute, [473](#)
- set_clock, [471](#)
- set_input, [471](#)
- set_port_statuses, [473](#)
- set_ppqn, [472](#)
- set_sequence_input, [468](#)
- start, [465](#)
- stop, [465](#)
- sysex, [467](#)
- seq64::mastermidibus, [480](#)
 - ~mastermidibus, [481](#)
 - activate, [482](#)
 - api_flush, [483](#)
 - api_get_midi_event, [482](#)
 - api_init, [482](#)
 - api_poll_for_midi, [482](#)
 - api_port_start, [483](#)
 - api_set_beats_per_minute, [483](#)
 - api_set_ppqn, [482](#)
 - m_midi_master, [484](#)
 - m_use_jack_polling, [484](#)
 - mastermidibus, [481](#)
 - midi_alsa_info, [483](#)
 - midi_jack_info, [483](#)
 - port_list, [483](#)
- seq64::midi_alsa, [485](#)
 - ~midi_alsa, [487](#)
 - api_clock, [489](#)
 - api_continue_from, [489](#)
 - api_deinit_in, [488](#)
 - api_flush, [488](#)
 - api_get_midi_event, [488](#)
 - api_init_in, [487](#)
 - api_init_in_sub, [488](#)
 - api_init_out, [487](#)
 - api_init_out_sub, [487](#)
 - api_play, [488](#)
 - api_set_beats_per_minute, [489](#)
 - api_set_ppqn, [489](#)
 - api_start, [489](#)
 - api_stop, [489](#)
 - api_sysex, [488](#)
 - get_client, [487](#)
 - get_port, [487](#)
 - m_dest_addr_client, [490](#)
 - m_dest_addr_port, [490](#)
 - m_input_port_name, [491](#)
 - m_local_addr_client, [490](#)
 - m_local_addr_port, [490](#)
 - m_seq, [490](#)
 - midi_alsa, [486](#)
 - set_virtual_name, [490](#)
- seq64::midi_alsa_info, [491](#)
 - ~midi_alsa_info, [493](#)
 - api_flush, [495](#)
 - api_get_midi_event, [494](#)
 - api_poll_for_midi, [494](#)
 - api_port_start, [494](#)
 - api_set_beats_per_minute, [494](#)
 - api_set_ppqn, [494](#)
 - get_all_port_info, [495](#)
 - m_alsa_seq, [495](#)
 - m_num_poll_descriptors, [495](#)
 - m_poll_descriptors, [496](#)
 - midi_alsa_info, [493](#)

- seq, [494](#)
- sm_input_caps, [495](#)
- sm_output_caps, [495](#)
- seq64::midi_api, [496](#)
 - ~midi_api, [499](#)
 - api_clock, [502](#)
 - api_connect, [500](#)
 - api_continue_from, [502](#)
 - api_deinit_in, [501](#)
 - api_flush, [502](#)
 - api_get_bus_name, [503](#)
 - api_get_midi_event, [501](#)
 - api_get_port_name, [503](#)
 - api_init_in, [500](#)
 - api_init_in_sub, [501](#)
 - api_init_out, [500](#)
 - api_init_out_sub, [500](#)
 - api_play, [501](#)
 - api_poll_for_midi, [500](#)
 - api_set_beats_per_minute, [503](#)
 - api_set_ppqn, [503](#)
 - api_start, [502](#)
 - api_stop, [502](#)
 - api_sysex, [501](#)
 - cancel_callback, [504](#)
 - error, [504](#)
 - input_data, [505](#)
 - is_input_port, [499](#)
 - is_port_open, [503](#)
 - is_system_port, [500](#)
 - is_virtual_port, [499](#)
 - m_connected, [505](#)
 - m_error_callback, [506](#)
 - m_error_callback_user_data, [506](#)
 - m_error_string, [505](#)
 - m_first_error_occurred, [506](#)
 - m_input_data, [505](#)
 - m_master_info, [505](#)
 - m_parent_bus, [505](#)
 - master_info, [503](#), [504](#)
 - master_midi_mode, [504](#)
 - midi_api, [499](#)
 - parent_bus, [504](#)
 - set_port_open, [505](#)
 - user_callback, [504](#)
- seq64::midi_container, [506](#)
 - ~midi_container, [509](#)
 - add_event, [513](#)
 - add_ex_event, [513](#)
 - add_long, [512](#)
 - add_short, [512](#)
 - add_variable, [512](#)
 - clear, [511](#)
 - done, [511](#)
 - fill, [509](#)
 - fill_meta_track_end, [514](#)
 - fill_proprietary, [514](#)
 - fill_seq_name, [514](#)
 - fill_seq_number, [513](#)
 - get, [511](#)
 - m_position_for_get, [516](#)
 - m_sequence, [515](#)
 - midi_container, [509](#)
 - midifile, [515](#)
 - position, [511](#)
 - position_increment, [512](#)
 - position_reset, [511](#)
 - put, [511](#)
 - size, [510](#)
 - song_fill_seq_event, [514](#)
 - song_fill_seq_trigger, [515](#)
- seq64::midi_control, [516](#)
 - action, [517](#)
 - active, [518](#)
 - data, [518](#)
 - in_range, [520](#)
 - inverse_active, [518](#)
 - m_active, [520](#)
 - m_data, [520](#)
 - m_inverse_active, [520](#)
 - m_max_value, [521](#)
 - m_min_value, [520](#)
 - m_status, [520](#)
 - match, [519](#)
 - max_value, [519](#)
 - midi_control, [518](#)
 - min_value, [519](#)
 - set, [519](#)
 - status, [518](#)
- seq64::midi_in_alsa, [521](#)
 - midi_in_alsa, [523](#)
- seq64::midi_in_jack, [523](#)
 - ~midi_in_jack, [525](#)
 - api_get_midi_event, [525](#)
 - api_poll_for_midi, [525](#)
 - m_client_name, [526](#)
 - midi_in_jack, [525](#)
 - open_client, [525](#)
- seq64::midi_info, [526](#)
 - ~midi_info, [529](#)
 - add_bus, [534](#)
 - api_connect, [532](#)
 - api_flush, [532](#)
 - api_get_midi_event, [531](#)
 - api_poll_for_midi, [532](#)
 - api_port_start, [531](#)
 - api_set_beats_per_minute, [531](#)
 - api_set_ppqn, [531](#)
 - app_name, [531](#)
 - bpm, [531](#)
 - bus_container, [535](#)
 - clear, [530](#)
 - connect_name, [533](#)
 - error, [534](#)
 - full_port_count, [530](#)
 - get_all_port_info, [534](#)

- get_bus_id, 532
- get_bus_name, 532
- get_input, 533
- get_port_count, 532
- get_port_id, 533
- get_port_name, 533
- get_system, 533
- get_virtual, 533
- global_queue, 534
- input_ports, 530
- m_app_name, 536
- m_bpm, 536
- m_bus_container, 536
- m_error_string, 536
- m_global_queue, 536
- m_input, 535
- m_midi_handle, 536
- m_midi_mode_input, 535
- m_output, 536
- m_ppqn, 536
- midi_handle, 530, 534
- midi_info, 529
- midi_mode, 530
- nc_midi_port_info, 535
- output_ports, 530
- port_list, 534
- ppqn, 531
- queue_number, 533
- ref_midi_port_info, 535
- rtmidi_info, 535
- seq64::midi_jack, 537
 - ~midi_jack, 539
 - api_clock, 544
 - api_connect, 541
 - api_continue_from, 543
 - api_deinit_in, 542
 - api_flush, 543
 - api_get_midi_event, 542
 - api_get_port_name, 544
 - api_init_in, 542
 - api_init_in_sub, 542
 - api_init_out, 542
 - api_init_out_sub, 542
 - api_play, 543
 - api_set_beats_per_minute, 544
 - api_set_ppqn, 544
 - api_start, 543
 - api_stop, 544
 - api_sysex, 543
 - client_handle, 539, 540
 - close_client, 541
 - close_port, 541
 - connect_port, 541
 - create_ringbuffer, 541
 - jack_data, 540
 - m_jack_data, 545
 - m_jack_info, 545
 - m_remote_port_name, 545
 - midi_jack, 539
 - midi_jack_info, 545
 - open_client, 541
 - open_client_impl, 540
 - port_handle, 540
 - register_port, 541
 - remote_port_name, 540
 - send_byte, 544
 - send_message, 545
 - set_virtual_name, 545
- seq64::midi_jack_data, 546
 - ~midi_jack_data, 547
 - m_jack_buffmessage, 547
 - m_jack_buffsize, 547
 - m_jack_client, 547
 - m_jack_lasttime, 548
 - m_jack_port, 547
 - m_jack_rtmidiin, 548
 - midi_jack_data, 546
 - valid_buffer, 547
- seq64::midi_jack_info, 548
 - ~midi_jack_info, 550
 - add, 553
 - api_connect, 551
 - api_flush, 552
 - api_get_midi_event, 551
 - api_poll_for_midi, 551
 - api_port_start, 552
 - api_set_beats_per_minute, 551
 - api_set_ppqn, 551
 - client_handle, 551, 552
 - connect, 552
 - disconnect, 552
 - extract_names, 553
 - get_all_port_info, 552
 - jack_process_io, 553
 - m_jack_client, 553
 - m_jack_client_2, 554
 - m_jack_ports, 553
 - midi_jack, 553
 - midi_jack_info, 550
- seq64::midi_list, 554
 - ~midi_list, 557
 - CharList, 556
 - clear, 557
 - done, 557
 - get, 557
 - m_char_list, 558
 - midi_list, 556
 - put, 557
 - size, 557
- seq64::midi_measures, 558
 - beats, 560
 - divisions, 560
 - m_beats, 561
 - m_divisions, 561
 - m_measures, 560
 - measures, 559

- midi_measures, 559
- seq64::midi_message, 561
 - array, 562
 - container, 562
 - count, 562
 - empty, 563
 - is_sysex, 563
 - m_bytes, 563
 - m_timestamp, 564
 - midi_message, 562
 - operator[], 562
 - push, 563
 - show, 563
 - timestamp, 563
- seq64::midi_out_alsa, 564
 - midi_out_alsa, 566
- seq64::midi_out_jack, 566
 - ~midi_out_jack, 568
 - midi_out_jack, 568
 - open_client, 568
- seq64::midi_port_info, 568
 - add, 569
 - clear, 570
 - connect_name, 571
 - get_bus_id, 570
 - get_bus_name, 570
 - get_input, 570
 - get_port_count, 570
 - get_port_id, 570
 - get_port_name, 570
 - get_queue_number, 571
 - get_system, 571
 - get_virtual, 571
 - m_port_container, 571
 - m_port_count, 571
 - midi_port_info, 569
- seq64::midi_port_info::port_info_t, 832
 - m_client_name, 833
 - m_client_number, 833
 - m_is_input, 834
 - m_is_system, 834
 - m_is_virtual, 834
 - m_port_name, 833
 - m_port_number, 833
 - m_queue_number, 834
- seq64::midi_queue, 572
 - ~midi_queue, 572
 - add, 573
 - allocate, 574
 - count, 573
 - deallocate, 574
 - empty, 573
 - front, 574
 - full, 573
 - m_back, 574
 - m_front, 574
 - m_ring, 575
 - m_ring_size, 574
 - m_size, 574
 - midi_queue, 572
 - pop, 573
 - pop_front, 573
- seq64::midi_splitter, 575
 - ~midi_splitter, 576
 - count, 578
 - increment, 577
 - initialize, 577
 - log_main_sequence, 577
 - m_ppqn, 579
 - m_smf0_channels, 579
 - m_smf0_channels_count, 579
 - m_smf0_main_sequence, 579
 - m_smf0_seq_number, 579
 - m_use_default_ppqn, 579
 - midi_splitter, 576
 - ppqn, 578
 - split, 577
 - split_channel, 578
- seq64::midi_timing, 580
 - beat_width, 582
 - beats_per_measure, 582
 - beats_per_minute, 581
 - m_beat_width, 583
 - m_beats_per_measure, 583
 - m_beats_per_minute, 583
 - m_ppqn, 583
 - midi_timing, 581
 - ppqn, 582
- seq64::midi_vector, 584
 - ~midi_vector, 586
 - CharVector, 585
 - clear, 587
 - done, 586
 - get, 587
 - m_char_vector, 587
 - midi_vector, 585
 - put, 586
 - size, 586
- seq64::midibase, 588
 - ~midibase, 593
 - api_clock, 606
 - api_continue_from, 606
 - api_deinit_in, 605
 - api_flush, 605
 - api_get_midi_event, 604
 - api_init_in, 606
 - api_init_in_sub, 605
 - api_init_out, 606
 - api_init_out_sub, 605
 - api_play, 605
 - api_poll_for_midi, 604
 - api_start, 606
 - api_stop, 606
 - api_sysex, 605
 - bpm, 595
 - bus_name, 594, 604

- clock, 602
- clock_enabled, 597
- connect_name, 594
- continue_from, 603
- deinit_in, 601
- display_name, 594, 604
- flush, 602
- get_bus_id, 595
- get_bus_index, 595
- get_clock, 597
- get_clock_mod, 600
- get_input, 597
- get_midi_event, 600
- get_port_id, 595
- init_clock, 603
- init_in, 600
- init_in_sub, 601
- init_out, 600
- init_out_sub, 601
- is_input_port, 596
- is_output_port, 596
- is_system_port, 596
- is_virtual_port, 595, 596
- m_bpm, 608
- m_bus_id, 607
- m_bus_index, 607
- m_bus_name, 608
- m_clock_mod, 607
- m_clock_type, 607
- m_display_name, 608
- m_inputting, 608
- m_is_input_port, 609
- m_is_system_port, 609
- m_is_virtual_port, 609
- m_lasttick, 609
- m_mutex, 609
- m_port_id, 607
- m_port_name, 608
- m_ppqn, 608
- m_queue, 608
- mastermidibus, 607
- match, 595
- midibase, 592
- play, 601
- poll_for_midi, 600
- port_disabled, 597
- port_name, 594, 604
- ppqn, 595
- print, 603
- queue_number, 597
- set_alt_name, 598
- set_bus_id, 597
- set_clock, 596
- set_clock_mod, 599
- set_input, 603
- set_input_status, 597
- set_multi_name, 599
- set_name, 598
- set_port_id, 604
- set_system_port_flag, 596
- show_bus_values, 594
- show_clock, 594
- start, 602
- stop, 602
- sysex, 602
- seq64::midibus, 610
 - ~midibus, 611
 - api_clock, 613
 - api_connect, 612
 - api_continue_from, 613
 - api_deinit_in, 612
 - api_get_midi_event, 613
 - api_init_in, 612
 - api_init_in_sub, 612
 - api_init_out, 612
 - api_init_out_sub, 612
 - api_play, 614
 - api_poll_for_midi, 613
 - api_start, 613
 - api_stop, 613
 - m_master_info, 614
 - m_rt_midi, 614
 - mastermidibus, 614
 - midibus, 611
- seq64::midifile, 614
 - ~midifile, 618
 - add_trigger, 626
 - at_end, 621
 - checklen, 625
 - clear_errors, 621
 - error_is_fatal, 620
 - error_message, 620
 - finalize_sequence, 621
 - get_file_pos, 621
 - grab_input_stream, 621
 - initialize_sequence, 621
 - is_sysex_special_id, 637
 - m_char_list, 638
 - m_data, 638
 - m_disable_reported, 638
 - m_error_is_fatal, 638
 - m_error_message, 637
 - m_file_size, 637
 - m_global_bgsequence, 639
 - m_mutex, 637
 - m_name, 638
 - m_new_format, 638
 - m_pos, 638
 - m_ppqn, 639
 - m_smf0_splitter, 639
 - m_use_default_ppqn, 639
 - midifile, 618
 - parse, 618
 - parse_prop_header, 624
 - parse_proprietary_track, 625
 - parse_smf_0, 622

- parse_smf_1, 622
- ppqn, 620, 621
- prop_item_size, 635
- read_byte, 627
- read_byte_array, 627, 628
- read_gap, 628
- read_long, 627
- read_seek, 626
- read_seq_number, 631
- read_short, 627
- read_track_name, 631
- read_varinum, 627
- seq_number_size, 637
- set_error, 635
- set_error_dump, 636
- track_end_size, 637
- track_name_size, 635
- varinum_size, 633
- write, 620
- write_byte, 629
- write_header, 632
- write_long, 628
- write_prop_header, 632
- write_proprietary_track, 633
- write_seq_number, 631
- write_short, 629
- write_song, 620
- write_track, 636
- write_track_end, 631
- write_track_name, 630
- write_triple, 629
- write_varinum, 630
- seq64::mutex, 640
 - lock, 641
 - m_mutex_lock, 641
 - mutex, 641
 - sm_recursive_mutex, 641
 - unlock, 641
- seq64::options, 642
 - add_extended_keys_page, 647
 - add_jack_sync_page, 648
 - add_keyboard_page, 647
 - add_midi_clock_page, 647
 - add_midi_input_page, 647
 - add_mouse_page, 647
 - button, 643
 - clock_callback_disable, 645
 - clock_callback_mod, 645
 - clock_callback_off, 644
 - clock_callback_on, 645
 - clock_mod_callback, 645
 - edit_tempo_track_number, 645
 - filter_callback, 646
 - input_callback, 646
 - lash_support_callback, 647
 - log_tempo_track_number, 645
 - m_button_jack_connect, 649
 - m_button_jack_disconnect, 649
 - m_button_jack_master, 648
 - m_button_jack_master_cond, 648
 - m_button_jack_transport, 648
 - m_button_ok, 648
 - m_mainperf, 648
 - m_notebook, 649
 - m_tooltips, 648
 - mouse_click_edit_callback, 647
 - mouse_fruity_callback, 646
 - mouse_mod4_callback, 646
 - mouse_seq24_callback, 646
 - mouse_snap_split_callback, 646
 - options, 644
 - perf, 644
 - transport_callback, 646
- seq64::optionsfile, 649
 - ~optionsfile, 651
 - error_message, 654
 - optionsfile, 651
 - parse, 651
 - parse_mute_group_section, 653
 - write, 653
- seq64::perfedit, 654
 - ~perfedit, 660
 - collapse, 664
 - copy, 664
 - draw_sequences, 665
 - enqueue_draw, 660
 - enregister_peer, 661
 - expand, 664
 - fast_forward, 662
 - get_toggle_jack, 661
 - grow, 664
 - init_before_show, 660
 - m_bpm, 672
 - m_button_bpm, 671
 - m_button_bw, 671
 - m_button_collapse, 670
 - m_button_copy, 670
 - m_button_expand, 670
 - m_button_follow, 671
 - m_button_grow, 670
 - m_button_jack, 670
 - m_button_loop, 670
 - m_button_play, 669
 - m_button_redo, 670
 - m_button_snap, 669
 - m_button_stop, 669
 - m_button_undo, 670
 - m_button_xpose, 669
 - m_bw, 672
 - m_entry_bpm, 671
 - m_entry_bw, 671
 - m_entry_snap, 669
 - m_entry_xpose, 669
 - m_hadjust, 668
 - m_hbox, 671
 - m_hlbox, 671

- m_hscroll, 668
- m_image_play, 669
- m_is_running, 672
- m_menu_bpm, 671
- m_menu_bw, 672
- m_menu_snap, 668
- m_menu_xpose, 669
- m_peer_perfedit, 667
- m_perfnames, 668
- m_perfroll, 668
- m_perftime, 668
- m_ppqn, 672
- m_snap, 672
- m_standard_bpm, 672
- m_table, 667
- m_vadjust, 668
- m_vscroll, 668
- on_delete_event, 667
- on_key_press_event, 666
- on_key_release_event, 667
- on_realize, 666
- pause_playing, 666
- perfedit, 659
- popup_menu, 665
- redo, 665
- rewind, 661
- set_beat_width, 663
- set_beats_per_bar, 663
- set_follow_transport, 662
- set_guides, 664
- set_image, 665
- set_jack_mode, 662
- set_looped, 664
- set_snap, 663
- set_transpose, 662
- set_zoom, 661
- start_playing, 666
- stop_playing, 666
- timeout, 665
- toggle_follow_transport, 662
- toggle_jack, 661
- toggle_playing, 666
- transpose_button_callback, 663
- undo, 665
- update_perfedit_sequences, 667
- zoom_check, 660
- seq64::perfnames, 673
- ~perfnames, 676
- change_vert, 677
- convert_y, 676
- draw_sequence, 676
- draw_sequences, 676
- enqueue_draw, 676
- m_char_w, 680
- m_namebox_w, 680
- m_names_chars, 680
- m_names_x, 680
- m_names_y, 680
- m_parent, 679
- m_seqs_in_set, 681
- m_sequence_active, 681
- m_sequence_max, 681
- m_sequence_offset, 681
- m_setbox_w, 680
- m_xy_offset, 680
- on_button_press_event, 678
- on_button_release_event, 678
- on_expose_event, 677
- on_realize, 677
- on_scroll_event, 679
- on_size_allocate, 679
- perfedit, 679
- perfnames, 675
- redraw, 677
- redraw_dirty_sequences, 676
- seq64::perform, 681
- ~perform, 700
- action_t, 699
- activate, 769
- add_clock, 780
- add_input, 780
- add_or_delete_trigger, 750
- add_sequence, 715
- add_trigger, 749
- all_notes_off, 770
- any_group_unmutes, 765
- apply_song_transpose, 720
- are_any_armed, 754
- armed_saved, 725
- bpm, 701
- clamp_group, 778
- clamp_track, 777
- clear_all, 714
- clear_current_screenset, 732
- clear_mute_groups, 721
- clear_sequence_triggers, 747
- clocks_per_metronome, 704
- collapse, 761
- combine_bytes, 708
- copy, 761
- copy_triggers, 748
- create_master_bus, 779
- current_screenset_notepad, 758
- decrement_beats_per_minute, 738
- decrement_screenset, 739
- deinit_jack_transport, 773
- delete_sequence, 716
- delete_trigger, 749
- enregister, 707
- expand, 761
- FF_RW_timeout, 708
- FF_rewind, 708
- fast_forward, 710
- ff_rw_button_t, 700
- ff_rw_type, 710
- filter_by_channel, 706

finish, 715
follow_progress, 709
get_32nds_per_quarter, 704
get_bank_name, 756
get_beat_width, 703
get_beats_per_bar, 703
get_beats_per_minute, 721
get_clock, 780
get_current_jack_position, 785
get_follow_transport, 709
get_group_mute_state, 728
get_input, 781
get_jack_tick, 717
get_key_count, 732
get_key_events, 732
get_key_events_rev, 732
get_key_groups, 732
get_key_groups_rev, 733
get_left_tick, 718
get_max_trigger, 751
get_playing_screensets, 753
get_right_tick, 720
get_screensets_notepad, 765
get_sequence, 740
get_sequence_color, 756
get_start_tick, 718
get_tempo_track_number, 704
get_tick, 717
get_toggle_jack, 708
get_transpose, 721
get_trigger_state, 749
group_max, 702
gui, 705
handle_midi_control, 763
handle_midi_control_event, 765
handle_midi_control_ex, 764
have_redo, 757
have_undo, 756
highlight, 739
increment_beats_per_minute, 738
increment_screensets, 739
init_jack_transport, 773
inner_start, 777
inner_stop, 777
input_func, 726
input_thread_func, 782
install_sequence, 776
intersect_triggers, 751
is_active, 720
is_control_status, 702
is_dirty_edit, 752
is_dirty_main, 751
is_dirty_names, 752
is_dirty_perf, 752
is_edit_sequence, 703
is_exportable, 753
is_group_learning, 760
is_input_system_port, 781
is_jack_master, 707
is_jack_running, 707
is_keep_queue, 724
is_modified, 701, 773
is_mseq_valid, 776
is_pattern_playing, 706
is_running, 706, 775
is_screensets_valid, 774
is_seq_valid, 775
is_sequence_in_edit, 716
is_smf_0, 740
jack_assistant, 781
jack_shutdown, 784
jack_sync_callback, 783
jack_timebase_callback, 784
jack_transport_callback, 783
key_name, 732
keybindentry, 781
keyboard_control_press, 745
keyboard_group_action, 746
keyboard_group_c_status_press, 745
keyboard_group_c_status_release, 745
keyboard_group_press, 746
keyboard_group_release, 746
keys, 705
launch, 714
launch_input_thread, 772
launch_output_thread, 772
learn_toggle, 738
left_right_size, 720
load_mute_group, 768
log_current_tempo, 779
lookup_keyevent_key, 733
lookup_keyevent_seq, 734
lookup_keygroup_group, 735
lookup_keygroup_key, 735
lookup_slot_key, 734
m_32nds_per_quarter, 793
m_FF_RW_button_type, 787
m_armed_saved, 787
m_armed_statuses, 787
m_beat_width, 792
m_beats_per_bar, 792
m_bpm, 792
m_clocks_per_metronome, 793
m_condition_var, 797
m_control_status, 796
m_current_tick, 792
m_dont_reset_ticks, 795
m_edit_sequence, 797
m_excell_FF_RW, 786
m_filter_by_channel, 793
m_gui_support, 798
m_have_redo, 798
m_have_undo, 798
m_in_thread, 790
m_in_thread_launched, 790
m_inputting, 791

`m_is_modified`, 797
`m_is_pattern_playing`, 791
`m_is_running`, 791
`m_jack_asst`, 798
`m_jack_tick`, 794
`m_left_tick`, 794
`m_looping`, 791
`m_master_bus`, 793
`m_master_clocks`, 793
`m_master_inputs`, 794
`m_max_groups`, 787
`m_max_sets`, 796
`m_midi_cc_off`, 796
`m_midi_cc_on`, 795
`m_midi_cc_toggle`, 795
`m_midi_mute_group_present`, 788
`m_midiclockpos`, 795
`m_midiclockrunning`, 795
`m_midiclocktick`, 795
`m_mode_group`, 788
`m_mode_group_learn`, 788
`m_mute_group`, 787
`m_mute_group_rc`, 787
`m_mute_group_selected`, 788
`m_notify`, 798
`m_one_measure`, 794
`m_out_thread`, 790
`m_out_thread_launched`, 790
`m_outputing`, 791
`m_playback_mode`, 792
`m_playscreen`, 796
`m_playscreen_offset`, 796
`m_ppqn`, 792
`m_queued_replace_slot`, 790
`m_redo_vect`, 798
`m_reposition`, 786
`m_resume_note_ons`, 792
`m_right_tick`, 794
`m_screenset`, 796
`m_screenset_notepad`, 795
`m_screenset_offset`, 796
`m_screenset_state`, 790
`m_seqs`, 788
`m_seqs_active`, 789
`m_seqs_in_set`, 787
`m_sequence_count`, 797
`m_sequence_high`, 797
`m_sequence_max`, 797
`m_sequence_state`, 789
`m_song_record_snap`, 791
`m_song_recording`, 791
`m_song_start_mode`, 786
`m_start_from_perfedit`, 786
`m_starting_tick`, 794
`m_tempo_track_number`, 793
`m_tick`, 794
`m_tracks_mute_state`, 788
`m_transpose`, 790
`m_undo_vect`, 798
`m_us_per_quarter_note`, 793
`m_usemidiclock`, 795
`m_was_active_edit`, 789
`m_was_active_main`, 789
`m_was_active_names`, 789
`m_was_active_perf`, 789
`main_window_title`, 742
`mainwnd`, 781
`mainwnd_key_event`, 744
`master_bus`, 705
`max_active_set`, 772
`max_sets`, 755, 774
`midi_control_event`, 762
`midi_control_off`, 762
`midi_control_on`, 762
`midi_control_record`, 763
`midi_control_toggle`, 761
`midi_mute_group_present`, 702, 781
`midifile`, 782
`modify`, 701
`move_triggers`, 747
`mute_all_tracks`, 724
`mute_group_offset`, 728
`mute_group_tracks`, 766
`mute_op_t`, 699
`mute_screenset`, 725
`new_sequence`, 715
`off_sequences`, 770
`options`, 782
`optionsfile`, 782
`output_func`, 725
`output_thread_func`, 783
`page_decrement_beats_per_minute`, 738
`page_increment_beats_per_minute`, 738
`panic`, 761
`paste_or_split_trigger`, 750
`paste_trigger`, 750
`pause_key`, 737
`pause_playing`, 736
`perfedit`, 782
`perform`, 700
`perfroll`, 782
`perfroll_key_event`, 746
`play`, 771
`playback_key_event`, 747
`playback_mode`, 760
`pop_trigger_redo`, 748
`pop_trigger_undo`, 748
`position_jack`, 769
`ppqn`, 701, 761
`preallocate_clocks`, 779
`print_busses`, 717
`print_group_unmutes`, 765
`print_triggers`, 747
`push_trigger_undo`, 748
`record_tempo_op_t`, 699
`reload_mute_groups`, 721

reposition, 711
reset_sequences, 771
restore_playing_state, 730
resume_note_ons, 755
rewind, 710
save_current_screensets, 730
save_mute_group, 768
save_playing_state, 730
screensets, 753
screensets_offset, 730, 775
select_and_mute_group, 766
select_group_mute, 767
select_trigger, 755
selected_trigger, 713
seq_edit_mode, 757
seq_in_playing_screen, 773
seqs_in_set, 755
sequence, 782
sequence_count, 701
sequence_high, 701
sequence_key, 741
sequence_label, 741, 742
sequence_max, 701
sequence_playing_change, 723
sequence_playing_off, 724
sequence_playing_on, 724
sequence_playing_toggle, 722
sequence_title, 742
sequence_window_title, 743
set_32nds_per_quarter, 704
set_active, 770
set_and_copy_mute_group, 769
set_beat_width, 703
set_beats_per_bar, 703
set_beats_per_minute, 760
set_clock, 780
set_clock_bus, 743
set_edit_sequence, 702
set_follow_transport, 709
set_group_mute_state, 727
set_have_redo, 757
set_have_undo, 757
set_input, 780
set_input_bus, 743
set_jack_mode, 707
set_jack_stop_tick, 708
set_jack_tick, 717
set_keep_queue, 723
set_key_event, 778
set_key_group, 779
set_left_tick, 717
set_looping, 756
set_mode_group_learn, 768
set_mode_group_mute, 767
set_orig_ticks, 772
set_overwrite_recording, 712
set_playing_screensets, 767
set_quantized_recording, 712
set_recording, 711
set_reposition, 710
set_right_tick, 718
set_screensets, 753
set_screensets_notepad, 758
set_sequence_color, 756
set_sequence_control_status, 722
set_sequence_input, 711
set_song_mute, 766
set_start_tick, 718
set_tempo_track_number, 704
set_thru, 713
set_tick, 717
set_transpose, 720
set_was_active, 771
show_ui_sequence_key, 733
show_ui_sequence_number, 733
slot_number, 730
sm_mc_dummy, 786
song_record_snap, 755, 760
song_recording, 755, 759
song_recording_stop, 759
song_start_mode, 706
split_trigger, 750
start, 758
start_from_perfect, 709
start_jack, 759
start_key, 737
start_playing, 735
stop, 759
stop_jack, 759
stop_key, 737
stop_playing, 737
toggle_all_tracks, 725
toggle_follow_transport, 709
toggle_jack_mode, 707
toggle_other_names, 754
toggle_other_seqs, 754
toggle_playing_tracks, 725
toggle_song_start_mode, 706
unqueue_sequences, 770
unselect_all_triggers, 756
unset_edit_sequence, 702
unset_mode_group_learn, 768
unset_mode_group_mute, 767
unset_queued_replace, 722
unset_sequence_control_status, 722
us_per_quarter_note, 705
valid_midi_control_seq, 774
wrkfile, 782
seq64::performcallback, 799
 on_grouplearnchange, 800
seq64::perforoll, 800
 ~perforoll, 806
 activate_adding, 813
 change_horz, 811
 change_vert, 811
 convert_drop_xy, 811

convert_x, 809
 convert_xy, 808
 draw_all, 807
 draw_background_on, 810
 draw_drawable_row, 810
 draw_progress, 808
 draw_sequence, 810
 draw_sequence_on, 810
 drop_action, 814
 enqueue_draw, 811
 fill_background_pixmap, 807
 follow_progress, 807
 growing, 814
 handle_motion_key, 813
 horizontal_adjust, 812
 horizontal_set, 812
 increment_size, 807
 init_before_show, 807
 is_adding, 813
 is_adding_pressed, 813
 m_4bar_offset, 820
 m_adding, 818
 m_adding_pressed, 818
 m_background_x, 819
 m_beat_length, 820
 m_divs_per_beat, 819
 m_drop_sequence, 821
 m_drop_tick, 821
 m_drop_tick_offset, 821
 m_grow_direction, 822
 m_growing, 822
 m_h_page_increment, 818
 m_have_button_press, 820
 m_measure_length, 820
 m_moving, 822
 m_names_y, 819
 m_old_progress_ticks, 820
 m_page_factor, 819
 m_parent, 817
 m_perf_scale_x, 819
 m_ppqn, 818
 m_roll_length_ticks, 821
 m_scroll_page, 820
 m_sequence_active, 821
 m_sequence_max, 821
 m_sequence_offset, 821
 m_size_box_w, 820
 m_snap_x, 818
 m_snap_y, 818
 m_ticks_per_bar, 819
 m_v_page_increment, 818
 m_w_scale_x, 819
 m_zoom, 819
 moving, 814
 offset_sequence, 810
 on_button_press_event, 815
 on_button_release_event, 815
 on_expose_event, 814
 on_focus_in_event, 816
 on_focus_out_event, 816
 on_key_press_event, 816
 on_motion_notify_event, 815
 on_realize, 814
 on_scroll_event, 815
 on_size_allocate, 816
 on_size_request, 816
 perfedit, 817
 perfroll, 805
 redraw_dirty_sequences, 808
 redraw_progress, 808
 set_adding, 813
 set_adding_pressed, 813
 set_guides, 806
 set_ppqn, 808
 set_zoom, 811
 sm_perfroll_background_x, 817
 sm_perfroll_size_box_click_w, 817
 sm_perfroll_size_box_w, 817
 snap_x, 809
 snap_y, 809
 split_trigger, 811
 update_sizes, 806
 vertical_adjust, 812
 vertical_set, 812
 seq64::perftime, 822
 ~perftime, 826
 change_horz, 827
 draw_background, 827
 draw_pixmap_on_window, 829
 draw_progress_on_window, 827
 enqueue_draw, 827
 idle_progress, 829
 increment_size, 827
 key_press_event, 830
 m_4bar_offset, 831
 m_left_marker_tick, 832
 m_measure_length, 832
 m_parent, 831
 m_perf_scale_x, 832
 m_ppqn, 831
 m_right_marker_tick, 832
 m_snap, 832
 m_tick_offset, 831
 m_timearea_y, 832
 on_button_press_event, 830
 on_button_release_event, 830
 on_expose_event, 829
 on_realize, 829
 on_size_allocate, 830
 perfedit, 831
 perftime, 825
 pixel_to_tick, 828
 reset, 826
 set_guides, 826
 set_ppqn, 827
 set_scale, 826

- set_zoom, 827
- tick_offset, 828
- tick_to_pixel, 828
- update_pixmap, 829
- update_sizes, 829
- seq64::rc_settings, 834
 - add_recent_file, 846
 - allow_click_edit, 843, 850
 - allow_mod4_mode, 842, 849
 - allow_snap_split, 842, 850
 - app_client_name, 848
 - append_comment_line, 842
 - append_recent_file, 847
 - application_name, 848
 - auto_option_save, 842, 849
 - clear_comments, 841
 - comments_block, 841
 - config_directory, 847, 852
 - config_filename, 848, 853
 - config_filename_alt, 848, 853
 - config_filespec, 841
 - device_ignore, 845, 851
 - device_ignore_num, 845, 851
 - filename, 845
 - filter_by_channel, 844, 851
 - help_check, 855
 - home_config_directory, 847
 - interaction_method, 845, 852
 - jack_session_uuid, 846, 852
 - lash_support, 842, 849
 - last_used_dir, 846
 - legacy_format, 842, 849
 - m_allow_click_edit, 856
 - m_allow_mod4_mode, 856
 - m_allow_snap_split, 856
 - m_app_client_name, 859
 - m_application_name, 859
 - m_auto_option_save, 855
 - m_comments_block, 855
 - m_config_directory, 859
 - m_config_filename, 859
 - m_config_filename_alt, 859
 - m_device_ignore, 858
 - m_device_ignore_num, 858
 - m_filename, 858
 - m_filter_by_channel, 857
 - m_interaction_method, 858
 - m_jack_session_uuid, 858
 - m_lash_support, 856
 - m_last_used_dir, 858
 - m_legacy_format, 856
 - m_manual_alsa_ports, 857
 - m_mute_group_saving, 858
 - m_pass_sysex, 857
 - m_print_keys, 858
 - m_priority, 856
 - m_recent_files, 860
 - m_reveal_alsa_ports, 857
 - m_show_midi, 856
 - m_stats, 856
 - m_tempo_track_number, 860
 - m_user_filename, 859
 - m_user_filename_alt, 859
 - m_verbose_option, 855
 - m_with_jack_master, 857
 - m_with_jack_master_cond, 857
 - m_with_jack_midi, 857
 - m_with_jack_transport, 857
 - mainwnd, 854
 - manual_alsa_ports, 844, 851
 - mute_group_saving, 845, 852
 - operator=, 840
 - options, 854
 - optionsfile, 854
 - parse_command_line_options, 854
 - pass_sysex, 843, 850
 - print_keys, 845, 851
 - priority, 843, 850
 - rc_settings, 840
 - recent_file, 848
 - recent_file_count, 849
 - remove_recent_file, 847
 - reveal_alsa_ports, 845, 851
 - rtmidi_info, 854
 - set_config_files, 847
 - set_defaults, 841
 - show_midi, 843, 850
 - stats, 843, 850
 - tempo_track_number, 848, 851
 - user_filename, 848, 853
 - user_filename_alt, 848, 853
 - user_filespec, 841
 - verbose_option, 842, 849
 - with_jack, 844
 - with_jack_master, 843, 844
 - with_jack_master_cond, 843, 844
 - with_jack_midi, 844, 850
 - with_jack_transport, 843, 844
- seq64::rect, 860
 - calculated_height, 866
 - calculated_width, 866
 - clear, 863
 - get, 862
 - height, 865
 - height_incr, 866
 - m_height, 867
 - m_width, 867
 - m_x, 866
 - m_y, 866
 - rect, 862
 - set, 862
 - width, 865
 - width_incr, 865
 - x, 864
 - x_incr, 864
 - xy_incr, 866

- xy_to_rect, 863, 864
- xy_to_rect_get, 863
- y, 864, 865
- y_incr, 865
- seq64::rterror, 867
 - ~rterror, 869
 - get_message, 869
 - getType, 869
 - m_message, 869
 - m_type, 869
 - print_message, 869
 - rterror, 868
 - Type, 868
 - what, 869
- seq64::rtmidi, 870
 - ~rtmidi, 872
 - api_clock, 873
 - api_connect, 872
 - api_continue_from, 872
 - api_deinit_in, 874
 - api_flush, 875
 - api_get_midi_event, 874
 - api_init_in, 874
 - api_init_in_sub, 874
 - api_init_out, 873
 - api_init_out_sub, 874
 - api_play, 872
 - api_poll_for_midi, 874
 - api_set_beats_per_minute, 873
 - api_set_ppqn, 873
 - api_start, 873
 - api_stop, 873
 - api_sysex, 875
 - delete_api, 877
 - full_port_count, 876
 - get_api, 876, 877
 - get_bus_id, 875
 - get_bus_name, 875
 - get_port_count, 876
 - get_port_id, 876
 - get_port_name, 876
 - is_port_open, 875
 - m_midi_api, 877
 - m_midi_info, 877
 - midibus, 877
 - rtmidi, 872
 - set_api, 877
- seq64::rtmidi_in, 878
 - ~rtmidi_in, 880
 - cancel_callback, 881
 - openmidi_api, 881
 - rtmidi_in, 880
 - user_callback, 880
- seq64::rtmidi_in_data, 881
 - api_data, 885
 - continue_sysex, 884
 - do_input, 883, 884
 - first_message, 884
 - ignore_flags, 883
 - m_api_data, 886
 - m_continue_sysex, 887
 - m_do_input, 886
 - m_first_message, 886
 - m_ignore_flags, 886
 - m_message, 886
 - m_queue, 886
 - m_user_callback, 887
 - m_user_data, 887
 - m_using_callback, 887
 - message, 883
 - queue, 882, 883
 - rtmidi_in_data, 882
 - test_ignore_flags, 883
 - user_callback, 885, 886
 - user_data, 885
 - using_callback, 884
- seq64::rtmidi_info, 887
 - ~rtmidi_info, 889
 - add_bus, 891
 - add_input, 890
 - add_output, 890
 - api_connect, 895
 - api_flush, 894
 - api_get_midi_event, 894
 - api_poll_for_midi, 894
 - api_port_start, 893
 - api_set_beats_per_minute, 893
 - api_set_ppqn, 893
 - app_name, 893
 - bpm, 893
 - clear, 890
 - delete_api, 895
 - full_port_count, 891
 - get_all_port_info, 892
 - get_api_info, 894
 - get_bus_id, 891
 - get_bus_name, 891
 - get_compiled_api, 890
 - get_input, 892
 - get_port_count, 891
 - get_port_id, 892
 - get_port_name, 892
 - get_system, 892
 - get_version, 890
 - get_virtual, 892
 - global_queue, 893
 - m_info_api, 896
 - mastermidibus, 895
 - midi_mode, 890
 - midibus, 896
 - openmidi_api, 895
 - port_list, 894
 - ppqn, 893
 - queue_number, 892
 - rtmidi_in, 896
 - rtmidi_info, 889

- rtmidi_out, 896
- selected_api, 894, 895
- set_api_info, 895
- sm_selected_api, 896
- seq64::rtmidi_out, 897
 - ~rtmidi_out, 898
 - openmidi_api, 898
 - rtmidi_out, 898
- seq64::seqdata, 907
 - ~seqdata, 911
 - change_horz, 914
 - convert_x, 914
 - draw_events_on, 914
 - draw_events_on_pixmap, 915
 - draw_line_on_window, 912
 - draw_pixmap_on_window, 915
 - idle_redraw, 912
 - lfownd, 918
 - m_cc, 919
 - m_drag_handle, 920
 - m_dragging, 920
 - m_number_h, 919
 - m_number_offset_y, 919
 - m_number_w, 919
 - m_old, 920
 - m_scroll_offset_ticks, 919
 - m_scroll_offset_x, 919
 - m_seq, 918
 - m_status, 919
 - m_zoom, 919
 - on_button_press_event, 916
 - on_button_release_event, 916
 - on_expose_event, 916
 - on_leave_notify_event, 917
 - on_motion_notify_event, 917
 - on_realize, 916
 - on_scroll_event, 917
 - on_size_allocate, 918
 - redraw, 911
 - render_digits, 915
 - render_number, 915
 - reset, 911
 - seqdata, 910
 - sequevent, 918
 - seqroll, 918
 - set_data_type, 912
 - set_zoom, 911
 - update_pixmap, 912
 - update_sizes, 912
 - xy_to_rect, 912
- seq64::seqlist, 920
 - ~seqlist, 929
 - apply_length, 934
 - change_focus, 942
 - create_menu_image, 941
 - create_menus, 938
 - do_action, 942
 - fill_top_bar, 938
 - follow_change_callback, 936
 - get_measures, 933
 - handle_close, 943
 - horizontal_adjust, 932
 - horizontal_set, 933
 - m_bgsequence, 947
 - m_button_bpm, 955
 - m_button_bus, 953
 - m_button_bw, 956
 - m_button_channel, 953
 - m_button_chord, 955
 - m_button_data, 955
 - m_button_key, 954
 - m_button_length, 954
 - m_button_lfo, 951
 - m_button_minidata, 955
 - m_button_note_length, 953
 - m_button_quantize, 952
 - m_button_rec_type, 956
 - m_button_rec_vol, 956
 - m_button_redo, 952
 - m_button_scale, 954
 - m_button_sequence, 952
 - m_button_snap, 953
 - m_button_tools, 952
 - m_button_undo, 952
 - m_button_zoom, 954
 - m_chord, 946
 - m_editing_cc, 957
 - m_editing_status, 957
 - m_entry_bpm, 956
 - m_entry_bus, 953
 - m_entry_bw, 956
 - m_entry_channel, 953
 - m_entry_chord, 955
 - m_entry_data, 955
 - m_entry_key, 954
 - m_entry_length, 954
 - m_entry_name, 957
 - m_entry_note_length, 954
 - m_entry_scale, 955
 - m_entry_seqnumber, 957
 - m_entry_sequence, 953
 - m_entry_snap, 953
 - m_entry_zoom, 954
 - m_first_event, 957
 - m_first_event_name, 958
 - m_hadjust, 950
 - m_have_focus, 958
 - m_hbox, 952
 - m_hbox2, 952
 - m_hscroll_new, 950
 - m_image_mousemode, 957
 - m_image_transpose, 948
 - m_initial_chord, 946
 - m_initial_note_length, 945
 - m_initial_snap, 945
 - m_initial_zoom, 946

- m_key, 946
- m_lfo_wnd, 951
- m_measures, 947
- m_menu_bpm, 950
- m_menu_bw, 950
- m_menu_chords, 949
- m_menu_data, 949
- m_menu_key, 949
- m_menu_length, 948
- m_menu_midibus, 949
- m_menu_midich, 949
- m_menu_minidata, 949
- m_menu_note_length, 948
- m_menu_rec_type, 950
- m_menu_rec_vol, 950
- m_menu_scale, 949
- m_menu_sequences, 949
- m_menu_snap, 948
- m_menu_tools, 948
- m_menu_zoom, 948
- m_menubar, 947
- m_note_length, 946
- m_pp_eighth, 947
- m_pp_sixteenth, 947
- m_pp_whole, 947
- m_ppqn, 947
- m_scale, 946
- m_seq, 947
- m_seqdata_wid, 951
- m_sequevent_wid, 951
- m_seqkeys_wid, 951
- m_seqroll_wid, 951
- m_seqtime_wid, 951
- m_snap, 946
- m_table, 951
- m_toggle_follow, 956
- m_toggle_play, 956
- m_toggle_q_rec, 957
- m_toggle_record, 956
- m_toggle_thru, 957
- m_toggle_transpose, 948
- m_tooltips, 955
- m_vadjust, 950
- m_vbox, 952
- m_vscroll_new, 950
- m_zoom, 946
- mouse_action, 942
- name_change_callback, 936
- on_delete_event, 943
- on_focus_in_event, 943
- on_focus_out_event, 943
- on_key_press_event, 944
- on_realize, 943
- on_scroll_event, 944
- on_set_focus, 943
- play_change_callback, 936
- popup_event_menu, 939
- popup_menu, 939
- popup_midibus_menu, 940
- popup_midich_menu, 941
- popup_mini_event_menu, 940
- popup_record_menu, 940
- popup_sequence_menu, 940
- popup_tool_menu, 941
- q_rec_change_callback, 937
- record_change_callback, 936
- redo_callback, 937
- repopulate_event_menu, 939
- repopulate_midich_menu, 941
- repopulate_mini_event_menu, 940
- seqedit, 928
- seqmenu, 945
- set_background_sequence, 936
- set_beat_width, 931
- set_beats_per_bar, 930
- set_beats_per_bar_manual, 931
- set_chord, 935
- set_data_type, 938
- set_event_entry, 939
- set_key, 935
- set_measures, 933
- set_measures_manual, 934
- set_midi_bus, 935
- set_midi_channel, 934
- set_mousemode_image, 931
- set_note_length, 930
- set_rec_type, 932
- set_rec_vol, 932
- set_scale, 935
- set_snap, 930
- set_transpose_image, 931
- set_zoom, 929
- start_playing, 942
- stop_playing, 942
- thru_change_callback, 937
- timeout, 941
- transpose_change_callback, 936
- undo_callback, 937
- update_all_windows, 937
- vertical_adjust, 932
- vertical_set, 933
- seq64::sequevent, 958
- ~sequevent, 962
- change_horz, 966
- convert_t, 967
- convert_x, 966
- draw_background, 964
- draw_events_on, 966
- draw_events_on_pixmap, 964
- draw_pixmap_on_window, 964
- draw_selection_on_window, 964
- drop_event, 966
- force_draw, 965
- idle_redraw, 965
- m_adding, 972
- m_cc, 973

- m_growing, 972
- m_move_snap_offset_x, 973
- m_moving, 972
- m_moving_init, 972
- m_old, 971
- m_painting, 972
- m_paste, 973
- m_scroll_offset_ticks, 971
- m_scroll_offset_x, 972
- m_selected, 971
- m_selecting, 972
- m_seq, 971
- m_seqdata_wid, 972
- m_snap, 971
- m_status, 973
- m_zoom, 971
- on_button_press_event, 968
- on_button_release_event, 969
- on_expose_event, 968
- on_focus_in_event, 970
- on_focus_out_event, 970
- on_key_press_event, 970
- on_motion_notify_event, 969
- on_realize, 968
- on_size_allocate, 970
- redraw, 963
- reset, 962
- seqevent, 962
- set_adding, 963
- set_data_type, 963
- set_snap, 963
- set_zoom, 963
- snap_x, 967
- snap_y, 967
- start_paste, 966
- update_pixmap, 965
- update_sizes, 964
- x_to_w, 965
- seq64::seqkeys, 973
 - ~seqkeys, 977
 - change_vert, 980
 - convert_y, 979
 - draw_area, 979
 - draw_key, 979
 - force_draw, 978
 - FruitySeqRollInput, 983
 - is_black_key, 980
 - m_hint_key, 984
 - m_hint_state, 984
 - m_key, 984
 - m_keying, 984
 - m_keying_note, 984
 - m_scale, 984
 - m_scroll_offset_key, 983
 - m_scroll_offset_y, 984
 - m_seq, 983
 - m_show_octave_letters, 984
 - on_button_press_event, 981
 - on_button_release_event, 981
 - on_enter_notify_event, 982
 - on_expose_event, 981
 - on_leave_notify_event, 982
 - on_motion_notify_event, 982
 - on_realize, 980
 - on_scroll_event, 982
 - on_size_allocate, 983
 - reset, 980
 - seqkeys, 976
 - seqroll, 983
 - set_hint_key, 977
 - set_hint_state, 978
 - set_key, 977
 - set_listen_button_press, 978
 - set_listen_button_release, 978
 - set_listen_motion_notify, 978
 - set_scale, 977
 - update_pixmap, 979
 - update_sizes, 980
- seq64::seqmenu, 985
 - ~seqmenu, 990
 - const_iterator, 989
 - create_seqedit, 993
 - current_seq, 990
 - delete_current_sequence, 992
 - get_current_sequence, 991
 - get_sequence, 991
 - is_current_seq_active, 991
 - is_current_seq_in_edit, 991
 - is_edit_sequence, 991
 - is_modified, 990, 991
 - iterator, 989
 - m_current_seq, 998
 - m_eventedit, 998
 - m_mainperf, 997
 - m_menu, 997
 - m_modified, 998
 - m_seqedit, 997
 - mainwnd, 996
 - mute_all_tracks, 996
 - new_current_sequence, 991
 - new_sequence, 992
 - on_realize, 996
 - popup_menu, 992
 - redraw, 994
 - remove_seqedit, 993
 - seq_clear_perf, 995
 - seq_copy, 994
 - seq_cut, 994
 - seq_edit, 992
 - seq_event_edit, 992
 - seq_new, 994
 - seq_paste, 995
 - seq_set_and_edit, 993
 - seq_set_and_eventedit, 994
 - seqedit, 996
 - SeqeditMap, 989

- SeqeditPair, 989
- seqmenu, 989
- set_bus_and_midi_channel, 995
- set_edit_sequence, 990
- set_transposable, 995
- sm_clipboard, 997
- sm_clipboard_empty, 997
- sm_seqedit_list, 997
- toggle_all_tracks, 996
- toggle_current_sequence, 992
- toggle_playing_tracks, 996
- unmute_all_tracks, 996
- unset_edit_sequence, 990
- seq64::seqroll, 998
 - ~seqroll, 1005
 - add_note, 1006
 - adding, 1022
 - align_selection, 1019
 - button_press, 1019
 - button_press_initial, 1019
 - button_release, 1020
 - change_horz, 1017
 - change_vert, 1017
 - clear_flags, 1020
 - complete_paste, 1011
 - convert_drop_xy, 1013
 - convert_sel_box_to_rect, 1015
 - convert_tn, 1014
 - convert_tn_box_to_rect, 1014, 1015
 - convert_xy, 1013
 - draw_background_on_pixmap, 1009
 - draw_events_on, 1016
 - draw_events_on_pixmap, 1009
 - draw_progress_on_window, 1010
 - draw_selection_on_window, 1009
 - drop_action, 1022
 - follow_progress, 1011
 - force_draw, 1012
 - get_expanded_record, 1011
 - get_progress_follow, 1012
 - get_selected_box, 1016
 - grow_selected_notes, 1018
 - growing, 1022
 - horizontal_adjust, 1012
 - idle_progress, 1017
 - idle_redraw, 1016
 - m_adding, 1028
 - m_background_sequence, 1031
 - m_cc, 1032
 - m_chord, 1028
 - m_drawing_background_seq, 1031
 - m_expanded_recording, 1031
 - m_growing, 1029
 - m_horizontal_adjust, 1027
 - m_is_drag_pasting, 1029
 - m_is_drag_pasting_start, 1029
 - m_justselected_one, 1030
 - m_key, 1028
 - m_move_delta_x, 1030
 - m_move_delta_y, 1030
 - m_move_snap_offset_x, 1030
 - m_moving, 1029
 - m_moving_init, 1029
 - m_note_length, 1028
 - m_old, 1027
 - m_painting, 1029
 - m_paste, 1029
 - m_pos, 1028
 - m_progress_follow, 1031
 - m_progress_x, 1030
 - m_scale, 1028
 - m_scroll_offset_key, 1030
 - m_scroll_offset_ticks, 1030
 - m_scroll_offset_x, 1030
 - m_scroll_offset_y, 1031
 - m_scroll_page, 1031
 - m_selected, 1027
 - m_selecting, 1029
 - m_seq, 1027
 - m_seqkeys_wid, 1027
 - m_snap, 1028
 - m_status, 1032
 - m_trans_button_press, 1031
 - m_transport_follow, 1031
 - m_vertical_adjust, 1027
 - m_zoom, 1028
 - motion_notify, 1020
 - move_selected_notes, 1017
 - move_selection_box, 1017
 - moving, 1023
 - normal_action, 1022
 - note_off_length, 1006
 - on_button_press_event, 1023
 - on_button_release_event, 1024
 - on_enter_notify_event, 1026
 - on_expose_event, 1023
 - on_focus_in_event, 1024
 - on_focus_out_event, 1025
 - on_key_press_event, 1025
 - on_leave_notify_event, 1026
 - on_motion_notify_event, 1024
 - on_realize, 1023
 - on_scroll_event, 1026
 - on_size_allocate, 1026
 - redraw, 1010
 - redraw_events, 1011
 - reset, 1010
 - scroll_offset_x, 1021
 - scroll_offset_y, 1021
 - select_action, 1022
 - selecting, 1022
 - seqroll, 1005
 - set_adding, 1018
 - set_background_sequence, 1007
 - set_chord, 1007
 - set_current_offset_x_y, 1021

- set_data_type, 1007
- set_expanded_recording, 1011
- set_key, 1007
- set_note_length, 1006
- set_progress_follow, 1012
- set_scale, 1007
- set_scroll_x, 1021
- set_scroll_y, 1021
- set_snap, 1005
- set_zoom, 1006
- snap_x, 1013
- snap_y, 1013
- start_paste, 1011
- update_and_draw, 1010
- update_background, 1009
- update_mouse_pointer, 1019
- update_pixmap, 1009
- update_sizes, 1009
- vertical_adjust, 1012
- xy_to_rect, 1014
- seq64::seqtime, 1032
 - ~seqtime, 1035
 - change_horz, 1036
 - draw_pixmap_on_window, 1035
 - draw_progress_on_window, 1035
 - idle_progress, 1037
 - m_scroll_offset_ticks, 1038
 - m_scroll_offset_x, 1038
 - m_seq, 1038
 - m_zoom, 1038
 - on_button_press_event, 1037
 - on_button_release_event, 1037
 - on_expose_event, 1037
 - on_realize, 1037
 - on_size_allocate, 1037
 - redraw, 1035
 - reset, 1035
 - seqtime, 1034
 - set_zoom, 1035
 - update_pixmap, 1036
 - update_sizes, 1036
- seq64::sequence, 1038
 - ~sequence, 1053
 - add_chord, 1076
 - add_event, 1076, 1077
 - add_note, 1075
 - add_trigger, 1078
 - adjust_offset, 1112
 - adjust_timestamp, 1094
 - adjust_trigger_offsets_to_length, 1112
 - any_selected_notes, 1054
 - append_event, 1077
 - apply_length, 1064, 1065
 - apply_song_transpose, 1061
 - background_sequence, 1108
 - calculate_measures, 1058
 - change_event_data_lfo, 1098
 - change_event_data_range, 1097
 - channel_match, 1109
 - channels_match, 1113
 - check_queued_tick, 1067
 - clear_triggers, 1085
 - clip_timestamp, 1095
 - clocks_per_metronome, 1060
 - color, 1055
 - copy_events, 1108
 - copy_selected, 1091
 - copy_selected_trigger, 1083
 - copy_triggers, 1085
 - cut_selected, 1091
 - cut_selected_trigger, 1083
 - decrement_selected, 1099
 - default_name, 1062
 - delete_selected_triggers, 1083
 - delete_trigger, 1079
 - edit_mode, 1056
 - event_count, 1056
 - event_in_range, 1110
 - EventStack, 1052
 - events, 1053
 - exact_split_trigger, 1079
 - extend, 1065
 - get_32nds_per_quarter, 1060
 - get_beat_width, 1060
 - get_beats_per_bar, 1059
 - get_clipboard_box, 1093
 - get_editing, 1063
 - get_hold_undo, 1057
 - get_last_tick, 1065
 - get_length, 1065
 - get_loop_reset, 1110
 - get_max_trigger, 1084
 - get_midi_bus, 1086
 - get_midi_channel, 1073
 - get_minmax_note_events, 1104
 - get_next_event, 1104
 - get_next_event_ex, 1105
 - get_next_note_event, 1103
 - get_next_trigger, 1105
 - get_num_measures, 1064
 - get_num_selected_events, 1090
 - get_num_selected_notes, 1090
 - get_onsets_selected_box, 1093
 - get_overwrite_rec, 1109
 - get_playing, 1066
 - get_ppqn, 1059
 - get_quantized_rec, 1069
 - get_queued, 1067
 - get_queued_tick, 1067
 - get_raise, 1063
 - get_recording, 1068
 - get_selected_box, 1092
 - get_snap_tick, 1068
 - get_song_mute, 1061
 - get_thru, 1069
 - get_transposable, 1062

get_trigger_offset, 1086
get_trigger_paste_tick, 1055
get_trigger_state, 1080
get_triggers, 1080
get_unit_measure, 1109
grow_selected, 1099
grow_trigger, 1079
half_split_trigger, 1078
handle_size, 1110
have_redo, 1057
have_undo, 1057
inc_draw_marker, 1102
increment_selected, 1098
intersect_events, 1082
intersect_notes, 1082
intersect_triggers, 1081
is_default_name, 1062
is_dirty_edit, 1072
is_dirty_main, 1071
is_dirty_names, 1072
is_dirty_perf, 1072
is_smf_0, 1073
link_new, 1101
link_tempos, 1101
m_32nds_per_quarter, 1122
m_background_sequence, 1123
m_bus, 1116
m_channel_match, 1116
m_clocks_per_metronome, 1122
m_dirty_edit, 1119
m_dirty_main, 1119
m_dirty_names, 1120
m_dirty_perf, 1119
m_editing, 1120
m_events, 1115
m_events_clipboard, 1114
m_events_redo, 1116
m_events_undo, 1115
m_events_undo_hold, 1115
m_have_redo, 1115
m_have_undo, 1115
m_iterator_draw, 1116
m_last_tick, 1120
m_length, 1121
m_loop_reset, 1119
m_master_bus, 1117
m_maxbeats, 1121
m_midi_channel, 1116
m_musical_key, 1123
m_musical_scale, 1123
m_mutex, 1123
m_name, 1120
m_note_off_margin, 1123
m_note_off_velocity, 1123
m_note_on_velocity, 1122
m_notes_on, 1117
m_off_from_snap, 1118
m_one_shot, 1118
m_one_shot_tick, 1118
m_overwrite_recording, 1119
m_parent, 1115
m_playing, 1117
m_playing_notes, 1117
m_ppqn, 1121
m_quantized_rec, 1117
m_queued, 1118
m_queued_tick, 1120
m_raise, 1120
m_rec_vol, 1122
m_recording, 1117
m_seq_color, 1121
m_seq_edit_mode, 1121
m_seq_number, 1121
m_snap_tick, 1121
m_song_mute, 1116
m_song_playback_block, 1118
m_song_record_tick, 1119
m_song_recording, 1118
m_song_recording_snap, 1119
m_thru, 1118
m_time_beat_width, 1122
m_time_beats_per_measure, 1122
m_transposable, 1116
m_trigger_offset, 1120
m_triggers, 1115
m_unit_measure, 1119
m_us_per_quarter_note, 1122
m_was_playing, 1117
mark_selected, 1100
measures_to_ticks, 1060
mod_last_tick, 1066
modify, 1056
move_selected_notes, 1095
move_triggers, 1083, 1085
multiply_pattern, 1107
musical_key, 1107
musical_scale, 1108
name, 1062
note_off_margin, 1109
number, 1055
off_from_snap, 1071, 1113
off_one_shot, 1069
off_playing_notes, 1102
one_shot, 1070, 1113
one_shot_tick, 1070
operator=, 1053
partial_assign, 1053
paste_selected, 1092
paste_trigger, 1083
pause, 1102
perform, 1114
play, 1074
play_note_off, 1102
play_note_on, 1101
play_queue, 1074
pop_redo, 1058

pop_trigger_redo, 1058
pop_trigger_undo, 1058
pop_undo, 1057
print, 1073
print_triggers, 1074
push_quantize, 1106
push_trigger_undo, 1058
push_undo, 1057
put_event_on_bus, 1111
quantize_events, 1106
recording_next_measure, 1068
remove, 1112
remove_all, 1113
remove_marked, 1100
remove_selected, 1100
reset_draw_marker, 1103
reset_draw_trigger_marker, 1103
reset_ex_iterator, 1103
reset_loop, 1111
resume_note_ons, 1071
select_action_t, 1052
select_all, 1091
select_all_notes, 1090
select_even_or_odd_notes, 1089
select_event_handle, 1088
select_events, 1087, 1088
select_linked, 1089
select_note_events, 1086
select_trigger, 1080
selected_trigger_count, 1054
selected_trigger_end, 1084
selected_trigger_start, 1084
seq_number, 1055
sequence, 1052
set_32nds_per_quarter, 1060
set_beat_width, 1059
set_beats_per_bar, 1059
set_dirty, 1073
set_dirty_mp, 1072
set_editing, 1063
set_have_redo, 1057
set_have_undo, 1057
set_hold_undo, 1056
set_input_recording, 1068
set_input_thru, 1069
set_last_tick, 1066
set_length, 1063
set_loop_reset, 1109
set_master_midi_bus, 1086
set_midi_bus, 1086
set_midi_channel, 1073
set_name, 1058
set_num_measures, 1064
set_overwrite_rec, 1109
set_parent, 1111
set_playing, 1066
set_quantized_recording, 1067
set_raise, 1063
set_rec_vol, 1061
set_recording, 1067
set_snap_tick, 1068
set_song_mute, 1061
set_thru, 1069
set_transposable, 1062
set_trigger_offset, 1111
set_trigger_paste_tick, 1054
set_unit_measure, 1109
shift_notes, 1107
show_events, 1108
sm_default_name, 1120
song_playback_block, 1071, 1113
song_record_tick, 1071, 1114
song_recording, 1070, 1114
song_recording_snap, 1071, 1114
song_recording_start, 1069
song_recording_stop, 1070
sort_events, 1078
split_trigger, 1078
stop, 1102
stream_event, 1096
stretch_selected, 1100
title, 1062
toggle_one_shot, 1071
toggle_playing, 1066
toggle_queued, 1067
toggle_song_mute, 1061
transpose_notes, 1107
trigger_count, 1054
triggerlist, 1054
triggers, 1114
trim_timestamp, 1094
unpaint_all, 1101
unselect, 1101
unselect_trigger, 1080
unselect_triggers, 1081
us_per_quarter_note, 1060, 1061
verify_and_link, 1101
zero_markers, 1101
seq64::trigger, 1123
 at_trigger_transition, 1126
 decrement_offset, 1127
 decrement_tick_end, 1127
 decrement_tick_start, 1126
 increment_offset, 1127
 increment_tick_end, 1126
 increment_tick_start, 1126
 length, 1125
 m_offset, 1128
 m_selected, 1128
 m_tick_end, 1128
 m_tick_start, 1128
 offset, 1127
 operator<, 1125
 selected, 1127
 tick_end, 1126
 tick_start, 1125

- trigger, 1125
- seq64::triggers, 1128
 - ~triggers, 1132
 - add, 1136
 - adjust_offset, 1144
 - adjust_offsets_to_length, 1136
 - clear, 1143
 - copy, 1142
 - copy_selected, 1140
 - count, 1133
 - exact_split, 1137
 - FruityPerfInput, 1146
 - get_maximum, 1141
 - get_selected_end, 1141
 - get_selected_start, 1141
 - get_state, 1138
 - get_trigger_paste_tick, 1144
 - grow, 1137
 - grow_edit_t, 1132
 - half_split, 1137
 - intersect, 1139, 1140
 - List, 1131
 - m_clipboard, 1147
 - m_iterator_draw_trigger, 1147
 - m_iterator_play_trigger, 1147
 - m_length, 1148
 - m_number_selected, 1147
 - m_parent, 1146
 - m_paste_tick, 1148
 - m_ppqn, 1148
 - m_redo_stack, 1147
 - m_trigger_copied, 1147
 - m_triggers, 1147
 - m_undo_stack, 1147
 - midi_container, 1146
 - midifile, 1146
 - move, 1142
 - move_selected, 1140
 - next, 1143
 - next_trigger, 1144
 - number_selected, 1133
 - operator=, 1132
 - paste, 1140
 - play, 1134
 - pop_redo, 1134
 - pop_undo, 1134
 - print, 1134
 - push_undo, 1133
 - remove, 1138
 - remove_selected, 1140
 - reset_draw_trigger_marker, 1144
 - select, 1138, 1145
 - Seq24PerfInput, 1146
 - sequence, 1146
 - set_length, 1133
 - set_ppqn, 1133
 - set_trigger_paste_tick, 1144
 - split, 1137, 1145
 - Stack, 1131
 - triggerlist, 1133
 - triggers, 1132
 - unselect, 1139, 1145
- seq64::user_instrument, 1148
 - controller_active, 1151
 - controller_count, 1150
 - controller_max, 1150
 - controller_name, 1150
 - copy_definitions, 1152
 - is_valid, 1150
 - m_controller_count, 1152
 - m_instrument_def, 1152
 - m_is_valid, 1152
 - name, 1150
 - operator=, 1150
 - set_controller, 1151
 - set_defaults, 1150
 - set_name, 1151
 - user_instrument, 1149
- seq64::user_instrument_t, 1152
 - controllers, 1153
 - controllers_active, 1153
 - instrument, 1153
- seq64::user_midi_bus, 1153
 - channel_count, 1156
 - channel_max, 1156
 - copy_definitions, 1157
 - instrument, 1156
 - is_valid, 1155
 - m_channel_count, 1157
 - m_is_valid, 1157
 - m_midi_bus_def, 1158
 - name, 1155
 - operator=, 1155
 - set_defaults, 1155
 - set_instrument, 1157
 - set_name, 1157
 - user_midi_bus, 1154, 1155
- seq64::user_midi_bus_t, 1158
 - alias, 1158
 - instrument, 1158
- seq64::user_settings, 1158
 - add_bus, 1171
 - add_instrument, 1171
 - allow_two_perfedits, 1180, 1187
 - append_comment_line, 1174
 - baseline_ppqn, 1186
 - block_columns, 1181, 1188
 - block_independent, 1181, 1188
 - block_rows, 1181, 1188
 - bpm_page_increment, 1185, 1190
 - bpm_precision, 1185, 1190
 - bpm_step_increment, 1185, 1190
 - bus, 1171
 - bus_count, 1171
 - bus_instrument, 1172
 - bus_name, 1172

BussConstIterator, 1169
BussIterator, 1169
Busses, 1169
clear_comments, 1173
comments_block, 1173
control_height, 1178, 1184
controller_active, 1173
controller_name, 1173
dump_summary, 1184
global_seq_feature, 1178
gmute_tracks, 1175
grid_brackets, 1175, 1181
grid_is_black, 1175
grid_is_normal, 1174
grid_is_white, 1174
grid_style, 1174, 1182
instrument, 1171
instrument_controller_active, 1173
instrument_controller_name, 1173
instrument_count, 1172
instrument_name, 1172
InstrumentConstIterator, 1169
InstrumentIterator, 1169
Instruments, 1169
inverse_colors, 1180, 1188
is_default_mainwid_size, 1175
is_variset, 1175
key_height, 1187, 1189
m_allow_two_perfedits, 1196
m_bpm_page_increment, 1200
m_bpm_precision, 1200
m_bpm_step_increment, 1200
m_comments_block, 1193
m_control_height, 1194
m_current_zoom, 1195
m_global_seq_feature_save, 1195
m_gmute_tracks, 1201
m_grid_brackets, 1193
m_grid_style, 1193
m_h_perf_page_increment, 1196
m_instruments, 1193
m_inverse_colors, 1196
m_mainwid_block_cols, 1197
m_mainwid_block_independent, 1197
m_mainwid_block_rows, 1197
m_mainwid_border, 1194
m_mainwid_spacing, 1194
m_mainwid_x, 1202
m_mainwid_y, 1202
m_mainwnd_cols, 1194
m_mainwnd_rows, 1194
m_mainwnd_x, 1202
m_mainwnd_y, 1202
m_max_sequence, 1201
m_max_sets, 1194
m_midi_beat_width, 1199
m_midi_beats_per_measure, 1198
m_midi_beats_per_minute, 1199
m_midi_bpm_maximum, 1199
m_midi_bpm_minimum, 1199
m_midi_buses, 1193
m_midi_buss_override, 1199
m_midi_ppqn, 1198
m_progress_bar_colored, 1196
m_progress_bar_thick, 1196
m_save_user_config, 1202
m_seqarea_seq_x, 1201
m_seqarea_seq_y, 1201
m_seqarea_x, 1201
m_seqarea_y, 1201
m_seqchars_x, 1198
m_seqchars_y, 1198
m_seqedit_bgsequence, 1195
m_seqedit_key, 1195
m_seqedit_scale, 1195
m_seqs_in_set, 1200
m_text_x, 1197
m_text_y, 1198
m_total_seqs, 1200
m_use_more_icons, 1197
m_use_new_font, 1196
m_user_option_daemonize, 1203
m_user_option_logfile, 1203
m_user_ui_key_height, 1204
m_user_use_logfile, 1203
m_v_perf_page_increment, 1196
m_velocity_override, 1199
m_window_redraw_rate_ms, 1197
m_window_scale, 1194
m_work_around_play_image, 1203
m_work_around_transpose_image, 1203
mainwid_border, 1177, 1183
mainwid_border_x, 1178
mainwid_border_y, 1178
mainwid_grid_style_t, 1169
mainwid_height, 1190
mainwid_height_fudge, 1191
mainwid_spacing, 1177, 1184
mainwid_width, 1190
mainwid_width_fudge, 1191
mainwid_x, 1177
mainwid_y, 1177
mainwnd_cols, 1175, 1182
mainwnd_rows, 1175, 1182
mainwnd_x, 1177
mainwnd_y, 1177
max_sequence, 1176
max_sets, 1176, 1182
max_zoom, 1186
mc_baseline_ppqn, 1203
mc_max_zoom, 1203
mc_min_zoom, 1202
midi_beat_width, 1185, 1192
midi_beats_per_bar, 1184, 1191
midi_beats_per_minute, 1184, 1191
midi_bpm_maximum, 1185, 1191

- midi_bpm_minimum, 1184, 1191
 - midi_buss_override, 1185, 1189
 - midi_ppqn, 1184, 1189
 - midifile, 1192
 - min_zoom, 1185
 - normalize, 1170
 - operator=, 1170
 - option_daemonize, 1186, 1188
 - option_logfile, 1186, 1189
 - option_use_logfile, 1186, 1189
 - parse_o_options, 1192
 - perf_h_page_increment, 1180, 1187
 - perf_v_page_increment, 1180, 1187
 - private_bus, 1192
 - private_instrument, 1192
 - progress_bar_colored, 1180, 1187
 - progress_bar_thick, 1180, 1187
 - save_user_config, 1181
 - scale_size, 1174
 - seqarea_seq_x, 1177, 1183
 - seqarea_seq_y, 1177, 1183
 - seqarea_x, 1176, 1183
 - seqarea_y, 1176, 1183
 - seqchars_x, 1176, 1183
 - seqchars_y, 1176, 1183
 - seqedit_bgsequence, 1179
 - seqedit_key, 1179
 - seqedit_scale, 1179
 - seqs_in_set, 1175
 - set_bus_instrument, 1171
 - set_defaults, 1170
 - set_instrument_controllers, 1172
 - text_x, 1176, 1182
 - text_y, 1176, 1182
 - use_more_icons, 1180, 1188
 - use_new_font, 1179, 1187
 - user_settings, 1170
 - userfile, 1192
 - velocity_override, 1185, 1190
 - window_redraw_rate, 1180, 1188
 - window_scale, 1174, 1181
 - window_scaled_down, 1174
 - work_around_play_image, 1186, 1189
 - work_around_transpose_image, 1186, 1189
 - zoom, 1178
- seq64::userfile, 1204
 - ~userfile, 1205
 - dump_setting_summary, 1206
 - parse, 1205
 - userfile, 1205
 - write, 1206
- seq_app_name
 - seq64, 92
- seq_clear_perf
 - seq64::seqmenu, 995
- seq_client_name
 - seq64, 92
- seq_copy
 - seq64::seqmenu, 994
- seq_cut
 - seq64::seqmenu, 994
- seq_edit
 - seq64::seqmenu, 992
- seq_edit_mode
 - seq64::perform, 757
- seq_event_edit
 - seq64::seqmenu, 992
- seq_event_type_t
 - seq64, 46
- seq_from_xy
 - seq64::mainwid, 418
- seq_in_playing_screen
 - seq64::perform, 773
- seq_modifier_t
 - seq64, 46
- seq_new
 - seq64::seqmenu, 994
- seq_number
 - seq64::sequence, 1055
- seq_number_size
 - seq64::midifile, 637
- seq_paste
 - seq64::seqmenu, 995
- seq_scroll_direction_t
 - seq64, 47
- seq_set_and_edit
 - seq64::mainwid, 416
 - seq64::seqmenu, 993
- seq_set_and_eventedit
 - seq64::mainwid, 416
 - seq64::seqmenu, 994
- seq_version
 - seq64, 92
- seqarea_seq_x
 - seq64::user_settings, 1177, 1183
- seqarea_seq_y
 - seq64::user_settings, 1177, 1183
- seqarea_x
 - seq64::user_settings, 1176, 1183
- seqarea_y
 - seq64::user_settings, 1176, 1183
- seqchars_x
 - seq64::user_settings, 1176, 1183
- seqchars_y
 - seq64::user_settings, 1176, 1183
- seqdata
 - seq64::seqdata, 910
- seqedit
 - seq64::seqedit, 928
 - seq64::seqmenu, 996
- seqedit_bgsequence
 - seq64::user_settings, 1179
- seqedit_key
 - seq64::user_settings, 1179
- seqedit_scale
 - seq64::user_settings, 1179

- SeqeditMap
 - seq64::seqmenu, 989
- SeqeditPair
 - seq64::seqmenu, 989
- seqevent
 - seq64::seqdata, 918
 - seq64::seqevent, 962
- seqkeys
 - seq64::seqkeys, 976
- seqmenu
 - seq64::seqedit, 945
 - seq64::seqmenu, 989
- seqroll
 - seq64::seqdata, 918
 - seq64::seqkeys, 983
 - seq64::seqroll, 1005
- seqs_in_set
 - seq64::perform, 755
 - seq64::user_settings, 1175
- seqspec_string
 - seq64::editable_event, 165
- seqtime
 - seq64::seqtime, 1034
- sequence
 - seq64::event_list, 220
 - seq64::perform, 782
 - seq64::sequence, 1052
 - seq64::triggers, 1146
- sequence_count
 - seq64::perform, 701
- sequence_high
 - seq64::perform, 701
- sequence_key
 - seq64::mainwnd, 442
 - seq64::perform, 741
- sequence_label
 - seq64::perform, 741, 742
- sequence_max
 - seq64::perform, 701
- sequence_playing_change
 - seq64::perform, 723
- sequence_playing_off
 - seq64::perform, 724
- sequence_playing_on
 - seq64::perform, 724
- sequence_playing_toggle
 - seq64::perform, 722
- sequence_title
 - seq64::perform, 742
- sequence_window_title
 - seq64::perform, 743
- session_event
 - seq64::jack_assistant, 329
- set
 - seq64::keybindentry, 348
 - seq64::midi_control, 519
 - seq64::rect, 862
- set_32nds_per_quarter
 - seq64::perform, 704
 - seq64::sequence, 1060
- set_active
 - seq64::perform, 770
- set_adding
 - seq64::perfroll, 813
 - seq64::seqevent, 963
 - seq64::seqroll, 1018
- set_adding_pressed
 - seq64::perfroll, 813
- set_all_clocks
 - seq64::busarray, 132
- set_all_inputs
 - seq64::busarray, 134
- set_all_key_events
 - seq64::keys_perform, 371
 - seq64::keys_perform_gtk2, 385
- set_all_key_groups
 - seq64::keys_perform, 371
 - seq64::keys_perform_gtk2, 385
- set_alsa_client_id
 - seq64::lash, 397
- set_alt_name
 - seq64::midibase, 598
- set_and_copy_mute_group
 - seq64::perform, 769
- set_api
 - seq64::rtmidi, 877
- set_api_info
 - seq64::rtmidi_info, 895
- set_background_sequence
 - seq64::seqedit, 936
 - seq64::seqroll, 1007
- set_basic_key_events
 - seq64::keys_perform, 376
- set_basic_key_groups
 - seq64::keys_perform, 376
- set_beat_width
 - seq64::jack_assistant, 327
 - seq64::perfedit, 663
 - seq64::perform, 703
 - seq64::seqedit, 931
 - seq64::sequence, 1059
- set_beats_per_bar
 - seq64::perfedit, 663
 - seq64::perform, 703
 - seq64::seqedit, 930
 - seq64::sequence, 1059
- set_beats_per_bar_manual
 - seq64::seqedit, 931
- set_beats_per_measure
 - seq64::jack_assistant, 327
- set_beats_per_minute
 - seq64::jack_assistant, 328
 - seq64::mastermidibase, 473
 - seq64::perform, 760
- set_bus_and_midi_channel
 - seq64::seqmenu, 995

- set_bus_id
 - seq64::midibase, 597
- set_bus_instrument
 - seq64::user_settings, 1171
- set_channel
 - seq64::event, 192
- set_chord
 - seq64::seqedit, 935
 - seq64::seqroll, 1007
- set_clock
 - seq64::busarray, 131
 - seq64::mastermidibase, 471
 - seq64::midibase, 596
 - seq64::perform, 780
- set_clock_bus
 - seq64::perform, 743
- set_clock_mod
 - seq64::midibase, 599
- set_config_files
 - seq64::rc_settings, 847
- set_controller
 - seq64::user_instrument, 1151
- set_current_directory
 - seq64, 90
- set_current_drop_x
 - seq64::gui_drawingarea_gtk2, 297
- set_current_drop_y
 - seq64::gui_drawingarea_gtk2, 297
- set_current_event
 - seq64::eventslots, 244
- set_current_offset_x_y
 - seq64::seqroll, 1021
- set_data
 - seq64::event, 193
- set_data_type
 - seq64::seqdata, 912
 - seq64::seqedit, 938
 - seq64::seqevent, 963
 - seq64::seqroll, 1007
- set_defaults
 - seq64::rc_settings, 841
 - seq64::user_instrument, 1150
 - seq64::user_midi_bus, 1155
 - seq64::user_settings, 1170
- set_dirty
 - seq64::eventedit, 228
 - seq64::sequence, 1073
- set_dirty_mp
 - seq64::sequence, 1072
- set_edit_sequence
 - seq64::perform, 702
 - seq64::seqmenu, 990
- set_editing
 - seq64::sequence, 1063
- set_error
 - seq64::midifile, 635
- set_error_dump
 - seq64::midifile, 636
- set_error_message
 - seq64::configfile, 153
- set_event_category
 - seq64::eventedit, 227
- set_event_data_0
 - seq64::eventedit, 228
- set_event_data_1
 - seq64::eventedit, 228
- set_event_entry
 - seq64::seqedit, 939
- set_event_name
 - seq64::eventedit, 228
- set_event_timestamp
 - seq64::eventedit, 227
- set_expanded_recording
 - seq64::seqroll, 1011
- set_follow_transport
 - seq64::jack_assistant, 334
 - seq64::perfedit, 662
 - seq64::perform, 709
- set_group_mute_state
 - seq64::perform, 727
- set_guides
 - seq64::perfedit, 664
 - seq64::perfroll, 806
 - seq64::perftime, 826
- set_have_redo
 - seq64::perform, 757
 - seq64::sequence, 1057
- set_have_undo
 - seq64::perform, 757
 - seq64::sequence, 1057
- set_hint_key
 - seq64::seqkeys, 977
- set_hint_state
 - seq64::seqkeys, 978
- set_hold_undo
 - seq64::sequence, 1056
- set_image
 - seq64::perfedit, 665
- set_input
 - seq64::busarray, 133
 - seq64::mastermidibase, 471
 - seq64::midibase, 603
 - seq64::perform, 780
- set_input_bus
 - seq64::perform, 743
- set_input_recording
 - seq64::sequence, 1068
- set_input_status
 - seq64::midibase, 597
- set_input_thru
 - seq64::sequence, 1069
- set_instrument
 - seq64::user_midi_bus, 1157
- set_instrument_controllers
 - seq64::user_settings, 1172
- set_jack_mode

- seq64::jack_assistant, [333](#)
- seq64::perfedit, [662](#)
- seq64::perform, [707](#)
- set_jack_running
 - seq64::jack_assistant, [335](#)
- set_jack_stop_tick
 - seq64::jack_assistant, [333](#)
 - seq64::perform, [708](#)
- set_jack_tick
 - seq64::perform, [717](#)
- set_keep_queue
 - seq64::perform, [723](#)
- set_key
 - seq64::seqedit, [935](#)
 - seq64::seqkeys, [977](#)
 - seq64::seqroll, [1007](#)
- set_key_event
 - seq64::keys_perform, [371](#)
 - seq64::perform, [778](#)
- set_key_group
 - seq64::keys_perform, [372](#)
 - seq64::perform, [779](#)
- set_keys
 - seq64::keys_perform, [357](#)
- set_last_tick
 - seq64::sequence, [1066](#)
- set_left_tick
 - seq64::perform, [717](#)
- set_length
 - seq64::sequence, [1063](#)
 - seq64::triggers, [1133](#)
- set_line
 - seq64::gui_drawingarea_gtk2, [287](#)
- set_listen_button_press
 - seq64::seqkeys, [978](#)
- set_listen_button_release
 - seq64::seqkeys, [978](#)
- set_listen_motion_notify
 - seq64::seqkeys, [978](#)
- set_loop_reset
 - seq64::sequence, [1109](#)
- set_looped
 - seq64::perfedit, [664](#)
- set_looping
 - seq64::perform, [756](#)
- set_master_midi_bus
 - seq64::sequence, [1086](#)
- set_measures
 - seq64::seqedit, [933](#)
- set_measures_manual
 - seq64::seqedit, [934](#)
- set_meta_status
 - seq64::event, [192](#)
- set_midi_bus
 - seq64::seqedit, [935](#)
 - seq64::sequence, [1086](#)
- set_midi_channel
 - seq64::seqedit, [934](#)
- seq64::sequence, [1073](#)
- set_mode_group_learn
 - seq64::perform, [768](#)
- set_mode_group_mute
 - seq64::perform, [767](#)
- set_mousemode_image
 - seq64::seqedit, [931](#)
- set_multi_name
 - seq64::midibase, [599](#)
- set_name
 - seq64::midibase, [598](#)
 - seq64::sequence, [1058](#)
 - seq64::user_instrument, [1151](#)
 - seq64::user_midi_bus, [1157](#)
- set_note
 - seq64::event, [199](#)
- set_note_length
 - seq64::seqedit, [930](#)
 - seq64::seqroll, [1006](#)
- set_note_velocity
 - seq64::event, [200](#)
- set_num_measures
 - seq64::sequence, [1064](#)
- set_orig_ticks
 - seq64::perform, [772](#)
- set_overwrite_rec
 - seq64::sequence, [1109](#)
- set_overwrite_recording
 - seq64::perform, [712](#)
- set_parent
 - seq64::sequence, [1111](#)
- set_play_image
 - seq64::mainwnd, [438](#)
- set_playing
 - seq64::sequence, [1066](#)
- set_playing_screenset
 - seq64::keys_perform, [361](#)
 - seq64::perform, [767](#)
- set_port_id
 - seq64::midibase, [604](#)
- set_port_open
 - seq64::midi_api, [505](#)
- set_port_statuses
 - seq64::mastermidibase, [473](#)
- set_ppqn
 - seq64::jack_assistant, [332](#)
 - seq64::mastermidibase, [472](#)
 - seq64::perfroll, [808](#)
 - seq64::perftime, [827](#)
 - seq64::triggers, [1133](#)
- set_progress_follow
 - seq64::seqroll, [1012](#)
- set_quantized_recording
 - seq64::perform, [712](#)
 - seq64::sequence, [1067](#)
- set_raise
 - seq64::sequence, [1063](#)
- set_rec_type

- seq64::seqedit, [932](#)
- set_rec_vol
 - seq64::seqedit, [932](#)
 - seq64::sequence, [1061](#)
- set_recording
 - seq64::perform, [711](#)
 - seq64::sequence, [1067](#)
- set_reposition
 - seq64::perform, [710](#)
- set_right_tick
 - seq64::perform, [718](#)
- set_scale
 - seq64::perftime, [826](#)
 - seq64::seqedit, [935](#)
 - seq64::seqkeys, [977](#)
 - seq64::seqroll, [1007](#)
- set_screenshot
 - seq64::mainwid, [414](#)
 - seq64::mainwnd, [439](#)
 - seq64::perform, [753](#)
- set_screenshot_notepad
 - seq64::perform, [758](#)
- set_scroll_x
 - seq64::seqroll, [1021](#)
- set_scroll_y
 - seq64::seqroll, [1021](#)
- set_seq_count
 - seq64::eventedit, [227](#)
- set_seq_length
 - seq64::eventedit, [227](#)
- set_seq_ppqn
 - seq64::eventedit, [227](#)
- set_seq_time_sig
 - seq64::eventedit, [226](#)
- set_seq_title
 - seq64::eventedit, [226](#)
- set_sequence_color
 - seq64::perform, [756](#)
- set_sequence_control_status
 - seq64::perform, [722](#)
- set_sequence_input
 - seq64::mastermidibase, [468](#)
 - seq64::perform, [711](#)
- set_snap
 - seq64::perfedit, [663](#)
 - seq64::seqedit, [930](#)
 - seq64::seqevent, [963](#)
 - seq64::seqroll, [1005](#)
- set_snap_tick
 - seq64::sequence, [1068](#)
- set_song_mute
 - seq64::mainwnd, [444](#)
 - seq64::perform, [766](#)
 - seq64::sequence, [1061](#)
- set_song_playback
 - seq64::mainwnd, [441](#)
- set_song_record
 - seq64::mainwnd, [440](#)
- set_sonlive_image
 - seq64::mainwnd, [438](#)
- set_start_from_perfedit
 - seq64::jack_assistant, [334](#)
- set_start_tick
 - seq64::perform, [718](#)
- set_status
 - seq64::event, [191](#)
- set_status_from_string
 - seq64::editable_event, [164](#)
- set_status_keep_channel
 - seq64::event, [192](#)
- set_status_text
 - seq64::mainwnd, [448](#)
- set_sysex
 - seq64::event, [196](#)
- set_sysex_size
 - seq64::event, [197](#)
- set_system_port_flag
 - seq64::midibase, [596](#)
- set_tempo
 - seq64::event, [202](#)
- set_tempo_track_number
 - seq64::perform, [704](#)
- set_text
 - seq64::eventslots, [248](#)
- set_thru
 - seq64::perform, [713](#)
 - seq64::sequence, [1069](#)
- set_tick
 - seq64::perform, [717](#)
- set_timestamp
 - seq64::event, [187](#)
- set_transposable
 - seq64::seqmenu, [995](#)
 - seq64::sequence, [1062](#)
- set_transpose
 - seq64::perfedit, [662](#)
 - seq64::perform, [720](#)
- set_transpose_image
 - seq64::seqedit, [931](#)
- set_trigger_offset
 - seq64::sequence, [1111](#)
- set_trigger_paste_tick
 - seq64::sequence, [1054](#)
 - seq64::triggers, [1144](#)
- set_unit_measure
 - seq64::sequence, [1109](#)
- set_virtual_name
 - seq64::midi_alsa, [490](#)
 - seq64::midi_jack, [545](#)
- set_was_active
 - seq64::perform, [771](#)
- set_wid_label
 - seq64::mainwnd, [437](#)
- set_zoom
 - seq64::perfedit, [661](#)
 - seq64::perfroll, [811](#)

- seq64::perftime, [827](#)
- seq64::seqdata, [911](#)
- seq64::seqedit, [929](#)
- seq64::seqevent, [963](#)
- seq64::seqroll, [1006](#)
- seq64::seqtime, [1035](#)
- shift_lock
 - seq64::keystroke, [395](#)
- shift_notes
 - seq64::sequence, [1107](#)
- shorten_file_spec
 - seq64, [58](#)
- show
 - seq64::midi_message, [563](#)
- show_bus_values
 - seq64::midibase, [594](#)
- show_clock
 - seq64::midibase, [594](#)
- show_events
 - seq64::sequence, [1108](#)
- show_jack_statuses
 - seq64, [84](#)
- show_midi
 - seq64::rc_settings, [843](#), [850](#)
- show_position
 - seq64::jack_assistant, [325](#)
- show_ui_sequence_key
 - seq64::keys_perform, [368](#)
 - seq64::perform, [733](#)
- show_ui_sequence_number
 - seq64::keys_perform, [368](#), [369](#)
 - seq64::perform, [733](#)
- signal
 - seq64::condition_var, [149](#)
- signal_action
 - seq64::mainwnd, [447](#)
- silence_jack_errors
 - seq64, [98](#)
- silence_jack_info
 - seq64, [99](#)
- size
 - seq64::midi_container, [510](#)
 - seq64::midi_list, [557](#)
 - seq64::midi_vector, [586](#)
- slot_number
 - seq64::perform, [730](#)
- SlotMap
 - seq64::keys_perform, [357](#)
- sm_category_arrays
 - seq64::editable_event, [167](#)
- sm_category_names
 - seq64::editable_event, [167](#)
- sm_channel_event_names
 - seq64::editable_event, [167](#)
- sm_clipboard
 - seq64::seqmenu, [997](#)
- sm_clipboard_empty
 - seq64::seqmenu, [997](#)
- sm_cond
 - seq64::condition_var, [149](#)
- sm_default_name
 - seq64::sequence, [1120](#)
- sm_input_caps
 - seq64::midi_alsa_info, [495](#)
- sm_internal_keys
 - seq64::gui_assistant_gtk2, [281](#)
- sm_mc_dummy
 - seq64::perform, [786](#)
- sm_meta_event_names
 - seq64::editable_event, [167](#)
- sm_meta_lengths
 - seq64::editable_event, [168](#)
- sm_output_caps
 - seq64::midi_alsa_info, [495](#)
- sm_perfroll_background_x
 - seq64::perfroll, [817](#)
- sm_perfroll_size_box_click_w
 - seq64::perfroll, [817](#)
- sm_perfroll_size_box_w
 - seq64::perfroll, [817](#)
- sm_prop_event_names
 - seq64::editable_event, [167](#)
- sm_recursive_mutex
 - seq64::mutex, [641](#)
- sm_selected_api
 - seq64::rtmidi_info, [896](#)
- sm_seqedit_list
 - seq64::seqmenu, [997](#)
- sm_sigpipe
 - seq64::mainwnd, [450](#)
- sm_status_pairs
 - seq64::jack_assistant, [340](#)
- sm_system_event_names
 - seq64::editable_event, [167](#)
- sm_widmax
 - seq64::mainwnd, [450](#)
- snap_x
 - seq64::perfroll, [809](#)
 - seq64::seqevent, [967](#)
 - seq64::seqroll, [1013](#)
- snap_y
 - seq64::perfroll, [809](#)
 - seq64::seqevent, [967](#)
 - seq64::seqroll, [1013](#)
- snapshot_1
 - seq64::keys_perform, [360](#)
- snapshot_2
 - seq64::keys_perform, [360](#)
- song_fill_seq_event
 - seq64::midi_container, [514](#)
- song_fill_seq_trigger
 - seq64::midi_container, [515](#)
- song_mode
 - seq64::keys_perform, [365](#)
- song_playback_block
 - seq64::sequence, [1071](#), [1113](#)

- song_record
 - seq64::keys_perform, [367](#), [368](#)
- song_record_snap
 - seq64::mainwnd, [441](#)
 - seq64::perform, [755](#), [760](#)
- song_record_tick
 - seq64::sequence, [1071](#), [1114](#)
- song_recording
 - seq64::perform, [755](#), [759](#)
 - seq64::sequence, [1070](#), [1114](#)
- song_recording_snap
 - seq64::sequence, [1071](#), [1114](#)
- song_recording_start
 - seq64::sequence, [1069](#)
- song_recording_stop
 - seq64::perform, [759](#)
 - seq64::sequence, [1070](#)
- song_start_mode
 - seq64::jack_assistant, [334](#)
 - seq64::perform, [706](#)
- sort
 - seq64::editable_events, [178](#)
 - seq64::event_list, [215](#)
- sort_events
 - seq64::sequence, [1078](#)
- spinner_max
 - seq64::mainwnd, [442](#)
- split
 - seq64::midi_splitter, [577](#)
 - seq64::triggers, [1137](#), [1145](#)
- split_channel
 - seq64::midi_splitter, [578](#)
- split_trigger
 - seq64::perform, [750](#)
 - seq64::perroll, [811](#)
 - seq64::sequence, [1078](#)
- Stack
 - seq64::triggers, [1131](#)
- start
 - seq64::busarray, [130](#)
 - seq64::businfo, [141](#)
 - seq64::jack_assistant, [330](#)
 - seq64::keys_perform, [363](#)
 - seq64::lash, [397](#)
 - seq64::mastermidibase, [465](#)
 - seq64::midibase, [602](#)
 - seq64::perform, [758](#)
- start_from_perfedit
 - seq64::perform, [709](#)
- start_jack
 - seq64::perform, [759](#)
- start_key
 - seq64::perform, [737](#)
- start_paste
 - seq64::seqevent, [966](#)
 - seq64::seqroll, [1011](#)
- start_playing
 - seq64::mainwnd, [438](#)
- seq64::perfedit, [666](#)
- seq64::perform, [735](#)
- seq64::seqedit, [942](#)
- stats
 - seq64::rc_settings, [843](#), [850](#)
- status
 - seq64::midi_control, [518](#)
- status_string
 - seq64::editable_event, [165](#)
- stock_event_string
 - seq64::editable_event, [165](#)
- stop
 - seq64::busarray, [130](#)
 - seq64::businfo, [141](#)
 - seq64::jack_assistant, [330](#)
 - seq64::keys_perform, [365](#)
 - seq64::mastermidibase, [465](#)
 - seq64::midibase, [602](#)
 - seq64::perform, [759](#)
 - seq64::sequence, [1102](#)
- stop_jack
 - seq64::perform, [759](#)
- stop_key
 - seq64::perform, [737](#)
- stop_playing
 - seq64::mainwnd, [439](#)
 - seq64::perfedit, [666](#)
 - seq64::perform, [737](#)
 - seq64::seqedit, [942](#)
- strcasecompare
 - seq64, [80](#)
- stream_event
 - seq64::sequence, [1096](#)
- stretch_selected
 - seq64::sequence, [1100](#)
- string_is_void
 - seq64, [59](#)
- string_not_void
 - seq64, [58](#)
- string_to_midibyte
 - seq64, [57](#)
- string_to_pulses
 - seq64, [57](#)
 - seq64::editable_events, [175](#)
- strings_match
 - seq64, [59](#)
- strip_quotes
 - seq64, [80](#)
- swap
 - seq64, [51](#)
- sync
 - seq64::jack_assistant, [336](#)
- sysex
 - seq64::busarray, [131](#)
 - seq64::businfo, [141](#)
 - seq64::mastermidibase, [467](#)
 - seq64::midibase, [602](#)
- SysexContainer

- seq64::event, [185](#)
- tap_bpm
 - seq64::keys_perform, [367](#)
- tempo
 - seq64::event, [202](#)
- tempo_log
 - seq64::mainwnd, [440](#)
- tempo_paint
 - seq64::gui_palette_gtk2, [310](#)
- tempo_to_note_value
 - seq64, [62](#)
- tempo_track_number
 - seq64::rc_settings, [848](#), [851](#)
- tempo_us_from_bpm
 - seq64, [63](#)
- tempo_us_from_bytes
 - seq64, [61](#)
- tempo_us_to_bytes
 - seq64, [61](#)
- test_ignore_flags
 - seq64::rtmidi_in_data, [883](#)
- test_widget_click
 - seq64, [96](#)
- text_x
 - seq64::user_settings, [1176](#), [1182](#)
- text_y
 - seq64::user_settings, [1176](#), [1182](#)
- thru_change_callback
 - seq64::seqedit, [937](#)
- tick_end
 - seq64::trigger, [1126](#)
- tick_multiplier
 - seq64::jack_assistant, [335](#)
- tick_offset
 - seq64::perftime, [828](#)
- tick_start
 - seq64::trigger, [1125](#)
- tick_to_pixel
 - seq64::perftime, [828](#)
- ticks_to_delta_time_us
 - seq64, [65](#)
- ticks_to_measures
 - seq64, [68](#)
- time_as_measures
 - seq64::editable_event, [164](#)
- time_as_minutes
 - seq64::editable_event, [164](#)
- time_as_pulses
 - seq64::editable_event, [164](#)
- timeout
 - seq64::mainwid, [418](#)
 - seq64::perfedit, [665](#)
 - seq64::seqedit, [941](#)
- timer_callback
 - seq64::mainwnd, [439](#)
- timestamp
 - seq64::editable_event, [163](#)
 - seq64::midi_message, [563](#)
- timestamp_format_t
 - seq64::editable_event, [159](#)
- timestamp_string
 - seq64::editable_event, [163](#)
- timestring_to_pulses
 - seq64, [57](#), [89](#)
- timing
 - seq64::editable_events, [174](#)
- title
 - seq64::sequence, [1062](#)
- to_string
 - seq64, [76](#)
- toLowerCase
 - seq64::mainwnd, [443](#)
- toggle_all_tracks
 - seq64::perform, [725](#)
 - seq64::seqmenu, [996](#)
- toggle_current_sequence
 - seq64::seqmenu, [992](#)
- toggle_follow_transport
 - seq64::jack_assistant, [334](#)
 - seq64::perfedit, [662](#)
 - seq64::perform, [709](#)
- toggle_jack
 - seq64::keys_perform, [367](#)
 - seq64::perfedit, [661](#)
- toggle_jack_mode
 - seq64::jack_assistant, [333](#)
 - seq64::perform, [707](#)
- toggle_mutes
 - seq64::keys_perform, [367](#)
- toggle_one_shot
 - seq64::sequence, [1071](#)
- toggle_other_names
 - seq64::perform, [754](#)
- toggle_other_seqs
 - seq64::perform, [754](#)
- toggle_playing
 - seq64::mainwnd, [439](#)
 - seq64::perfedit, [666](#)
 - seq64::sequence, [1066](#)
- toggle_playing_tracks
 - seq64::perform, [725](#)
 - seq64::seqmenu, [996](#)
- toggle_queued
 - seq64::sequence, [1067](#)
- toggle_song_mute
 - seq64::sequence, [1061](#)
- toggle_song_record
 - seq64::mainwnd, [440](#)
- toggle_song_snap
 - seq64::mainwnd, [441](#)
- toggle_song_start_mode
 - seq64::jack_assistant, [334](#)
 - seq64::perform, [706](#)
- toggle_tempo_record
 - seq64::mainwnd, [440](#)
- toggle_time_format

- seq64::mainwnd, 440
- toggle_visible
 - seq64::lfownd, 402
- tokenize_string
 - seq64, 53
- top_index
 - seq64::eventslots, 243
- track_end_size
 - seq64::midifile, 637
- track_name_size
 - seq64::midifile, 635
- transport_callback
 - seq64::options, 646
- transport_not_starting
 - seq64::jack_assistant, 328
- transport_state
 - seq64::jack_assistant, 328
- transpose_button_callback
 - seq64::perfedit, 663
- transpose_change_callback
 - seq64::seqedit, 936
- transpose_note
 - seq64::event, 200
- transpose_notes
 - seq64::sequence, 1107
- trigger
 - seq64::trigger, 1125
- trigger_count
 - seq64::sequence, 1054
- triggerlist
 - seq64::sequence, 1054
 - seq64::triggers, 1133
- triggers
 - seq64::sequence, 1114
 - seq64::triggers, 1132
- trim_timestamp
 - seq64::sequence, 1094
- Type
 - seq64::rterror, 868
- type
 - seq64::keybindentry, 347
- undaemonize
 - seq64, 75
- undo
 - seq64::perfedit, 665
- undo_callback
 - seq64::seqedit, 937
- unlock
 - seq64::mutex, 641
- unmark
 - seq64::event, 199
- unmark_all
 - seq64::event_list, 217
- unmodify
 - seq64::event_list, 214
- unmute_all_tracks
 - seq64::seqmenu, 996
- unpaint
 - seq64::event, 198
- unpaint_all
 - seq64::event_list, 218
 - seq64::sequence, 1101
- unqueue_sequences
 - seq64::perform, 770
- unselect
 - seq64::event, 199
 - seq64::sequence, 1101
 - seq64::triggers, 1139, 1145
- unselect_all
 - seq64::event_list, 219
- unselect_all_triggers
 - seq64::perform, 756
- unselect_trigger
 - seq64::sequence, 1080
- unselect_triggers
 - seq64::sequence, 1081
- unset_edit_sequence
 - seq64::perform, 702
 - seq64::seqmenu, 990
- unset_mode_group_learn
 - seq64::perform, 768
- unset_mode_group_mute
 - seq64::perform, 767
- unset_queued_replace
 - seq64::perform, 722
- unset_sequence_control_status
 - seq64::perform, 722
- update_all_windows
 - seq64::seqedit, 937
- update_and_draw
 - seq64::seqroll, 1010
- update_background
 - seq64::seqroll, 1009
- update_mainwid_sequences
 - seq64, 98
 - seq64::mainwid, 422
- update_markers
 - seq64::mainwid, 416
 - seq64::mainwnd, 437
- update_mouse_pointer
 - seq64::FruityPerfInput, 265
 - seq64::FruitySeqEventInput, 269
 - seq64::FruitySeqRollInput, 273
 - seq64::seqroll, 1019
- update_perfedit_sequences
 - seq64, 98
 - seq64::perfedit, 667
- update_pixmap
 - seq64::perftime, 829
 - seq64::seqdata, 912
 - seq64::seqevent, 965
 - seq64::seqkeys, 979
 - seq64::seqroll, 1009
 - seq64::seqtime, 1036
- update_recent_files_menu
 - seq64::mainwnd, 443

- update_screenset
 - seq64::mainwnd, [437](#)
- update_sequences_on_window
 - seq64::mainwnd, [415](#)
- update_sizes
 - seq64::perfroll, [806](#)
 - seq64::perftime, [829](#)
 - seq64::seqdata, [912](#)
 - seq64::seqevent, [964](#)
 - seq64::seqkeys, [980](#)
 - seq64::seqroll, [1009](#)
 - seq64::seqtime, [1036](#)
- update_window_title
 - seq64::mainwnd, [443](#)
- us_per_quarter_note
 - seq64::perform, [705](#)
 - seq64::sequence, [1060](#), [1061](#)
- use_more_icons
 - seq64::user_settings, [1180](#), [1188](#)
- use_new_font
 - seq64::user_settings, [1179](#), [1187](#)
- user_callback
 - seq64::midi_api, [504](#)
 - seq64::rtmidi_in, [880](#)
 - seq64::rtmidi_in_data, [885](#), [886](#)
- user_data
 - seq64::rtmidi_in_data, [885](#)
- user_filename
 - seq64::rc_settings, [848](#), [853](#)
- user_filename_alt
 - seq64::rc_settings, [848](#), [853](#)
- user_filespec
 - seq64::rc_settings, [841](#)
- user_instrument
 - seq64::user_instrument, [1149](#)
- user_midi_bus
 - seq64::user_midi_bus, [1154](#), [1155](#)
- user_settings
 - seq64::user_settings, [1170](#)
- userfile
 - seq64::user_settings, [1192](#)
 - seq64::userfile, [1205](#)
- using_callback
 - seq64::rtmidi_in_data, [884](#)
- usr
 - seq64, [88](#)
- v_adjustment
 - seq64::eventedit, [229](#)
- valid_buffer
 - seq64::midi_jack_data, [547](#)
- valid_midi_control_seq
 - seq64::perform, [774](#)
- valid_sequence
 - seq64::mainwnd, [417](#)
- value_to_name
 - seq64::editable_event, [161](#)
- varinum_size
 - seq64::midifile, [633](#)
- velocity_override
 - seq64::user_settings, [1185](#), [1190](#)
- verbose_option
 - seq64::rc_settings, [842](#), [849](#)
- verify_and_link
 - seq64::event_list, [216](#)
 - seq64::sequence, [1101](#)
- versiontext
 - seq64, [120](#)
- vertical_adjust
 - seq64::perfroll, [812](#)
 - seq64::seqedit, [932](#)
 - seq64::seqroll, [1012](#)
- vertical_set
 - seq64::perfroll, [812](#)
 - seq64::seqedit, [933](#)
- wait
 - seq64::condition_var, [149](#)
- wave_func
 - seq64, [69](#)
- wave_type_name
 - seq64, [52](#)
- wave_type_t
 - seq64, [45](#)
- what
 - seq64::rterror, [869](#)
- white
 - seq64::gui_palette_gtk2, [308](#)
- white_key_paint
 - seq64::gui_palette_gtk2, [310](#)
- white_paint
 - seq64::gui_palette_gtk2, [310](#)
- wid_box_to_slot
 - seq64::mainwnd, [444](#)
- wid_slot_to_box
 - seq64::mainwnd, [444](#)
- width
 - seq64::gui_drawingarea_gtk2, [285](#)
 - seq64::rect, [865](#)
- width_incr
 - seq64::rect, [865](#)
- window_redraw_rate
 - seq64::user_settings, [1180](#), [1188](#)
- window_scale
 - seq64::user_settings, [1174](#), [1181](#)
- window_scaled_down
 - seq64::user_settings, [1174](#)
- window_x
 - seq64::gui_drawingarea_gtk2, [285](#)
- window_y
 - seq64::gui_drawingarea_gtk2, [286](#)
- with_jack
 - seq64::rc_settings, [844](#)
- with_jack_master
 - seq64::rc_settings, [843](#), [844](#)
- with_jack_master_cond
 - seq64::rc_settings, [843](#), [844](#)
- with_jack_midi

- seq64::rc_settings, [844](#), [850](#)
- with_jack_transport
 - seq64::rc_settings, [843](#), [844](#)
- work_around_play_image
 - seq64::user_settings, [1186](#), [1189](#)
- work_around_transpose_image
 - seq64::user_settings, [1186](#), [1189](#)
- write
 - seq64::configfile, [153](#)
 - seq64::midifile, [620](#)
 - seq64::optionsfile, [653](#)
 - seq64::userfile, [1206](#)
- write_byte
 - seq64::midifile, [629](#)
- write_header
 - seq64::midifile, [632](#)
- write_long
 - seq64::midifile, [628](#)
- write_options_files
 - seq64, [74](#)
- write_prop_header
 - seq64::midifile, [632](#)
- write_proprietary_track
 - seq64::midifile, [633](#)
- write_seq_number
 - seq64::midifile, [631](#)
- write_short
 - seq64::midifile, [629](#)
- write_song
 - seq64::midifile, [620](#)
- write_track
 - seq64::midifile, [636](#)
- write_track_end
 - seq64::midifile, [631](#)
- write_track_name
 - seq64::midifile, [630](#)
- write_triple
 - seq64::midifile, [629](#)
- write_varinum
 - seq64::midifile, [630](#)
- wrkfile
 - seq64::perform, [782](#)
- x
 - seq64::click, [145](#)
 - seq64::rect, [864](#)
- x_incr
 - seq64::rect, [864](#)
- x_to_w
 - seq64::seqevent, [965](#)
- xy_incr
 - seq64::rect, [866](#)
- xy_to_rect
 - seq64::rect, [863](#), [864](#)
 - seq64::seqdata, [912](#)
 - seq64::seqroll, [1014](#)
- xy_to_rect_get
 - seq64::rect, [863](#)
- y
 - seq64::click, [146](#)
 - seq64::rect, [864](#), [865](#)
- y_incr
 - seq64::rect, [865](#)
- yellow
 - seq64::gui_palette_gtk2, [309](#)
- zero_markers
 - seq64::sequence, [1101](#)
- zoom
 - seq64::user_settings, [1178](#)
- zoom_check
 - seq64::perfedit, [660](#)
- zoom_power_of_2
 - seq64, [60](#)