

# Sequencer64 (seq64) Developer Reference Manual

## 0.90.2

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Sequencer64</b>	<b>1</b>
1.1	Introduction . . . . .	1
<b>2</b>	<b>MIDI File Parsing in Sequencer64</b>	<b>3</b>
2.1	Introduction . . . . .	3
2.2	SMF 1 Parsing . . . . .	3
2.2.1	MIDI File Header, MThd . . . . .	3
2.2.2	MIDI Track, MTrk . . . . .	4
2.2.2.1	Channel Events . . . . .	4
2.2.2.2	Meta Events . . . . .	5
2.2.3	Meta Events Summary . . . . .	6
2.2.3.1	Sequence Number (0x00) . . . . .	6
2.2.3.2	Track/Sequence Name (0x03) . . . . .	7
2.2.3.3	End of Track (0x2F) . . . . .	7
2.2.3.4	Set Tempo Event (0x51) . . . . .	7
2.2.3.5	Time Signature Event (0x58) . . . . .	8
2.2.3.6	SysEx Event (0xF0) . . . . .	9
2.2.3.7	Sequencer Specific (0x7F) . . . . .	10
2.2.3.8	Non-Specific End of Sequence . . . . .	10
2.3	SMF 0 Parsing . . . . .	10
2.4	Running Status . . . . .	11

<b>3</b>	<b>JACK, Live, and Song Modes in Sequencer64</b>	<b>13</b>
3.1	Introduction . . . . .	13
3.2	JACK Functions . . . . .	13
3.2.1	jack_client_open() . . . . .	14
3.2.2	jack_on_shutdown() . . . . .	14
3.2.3	jack_set_sync_callback() . . . . .	14
3.2.4	jack_set_process_callback() . . . . .	15
3.2.5	jack_set_session_callback() . . . . .	15
3.2.6	jack_activate() . . . . .	15
3.2.7	jack_release_timebase() . . . . .	15
3.2.8	jack_client_close() . . . . .	15
3.2.9	jack_transport_start() . . . . .	15
3.2.10	jack_transport_stop() . . . . .	15
3.2.11	jack_transport_locate() . . . . .	15
3.2.12	jack_transport_reposition() . . . . .	16
3.2.13	jack_transport_query() . . . . .	16
3.3	Modes Operation . . . . .	16
3.3.1	No JACK, Live Mode . . . . .	16
3.3.2	No JACK, Song Mode . . . . .	16
3.3.3	JACK Transport . . . . .	17
3.4	Breakage . . . . .	17
3.5	JACK References . . . . .	18
<b>4</b>	<b>User Testing of Sequencer64 with Yoshimi</b>	<b>19</b>
4.1	Introduction . . . . .	19
4.2	Smoke Test . . . . .	19
4.3	Tests in the Patterns Window . . . . .	20
4.3.1	Button Clicks on a Pattern . . . . .	21
4.3.2	Patterns Window Key Shortcuts . . . . .	21
4.3.3	The Sequencer64 User File . . . . .	21
4.4	Tests Using Valgrind . . . . .	21
4.4.1	Valgrind Suppressions . . . . .	22
4.4.2	Full Valgrind Leak-Checking . . . . .	22
4.4.2.1	Leak-Checking Basic Operation . . . . .	23
4.5	Specific Fault Debugging . . . . .	23
4.6	Snipping of a MIDI file. . . . .	23

<b>5</b>	<b>Speed Issue of Sequencer64</b>	<b>25</b>
5.1	Introduction . . . . .	25
5.2	Initial Change of Containers . . . . .	25
5.3	Back to the Original Container . . . . .	26
5.4	What is Next, the Vector? . . . . .	26
<b>6</b>	<b>Licenses</b>	<b>27</b>
6.1	License Terms for the This Project. . . . .	27
6.2	XPC Application License . . . . .	27
6.3	XPC Library License . . . . .	28
6.4	XPC Documentation License . . . . .	28
6.5	XPC Affero License . . . . .	29
6.6	XPC License Summary . . . . .	29
<b>7</b>	<b>Todo List</b>	<b>31</b>
<b>8</b>	<b>Deprecated List</b>	<b>35</b>
<b>9</b>	<b>Namespace Index</b>	<b>37</b>
9.1	Namespace List . . . . .	37
<b>10</b>	<b>Hierarchical Index</b>	<b>39</b>
10.1	Class Hierarchy . . . . .	39
<b>11</b>	<b>Data Structure Index</b>	<b>43</b>
11.1	Data Structures . . . . .	43

<b>12 Namespace Documentation</b>	<b>49</b>
12.1 Gtk Namespace Reference . . . . .	49
12.2 seq64 Namespace Reference . . . . .	49
12.2.1 Detailed Description . . . . .	64
12.2.2 Typedef Documentation . . . . .	64
12.2.2.1 midibyte . . . . .	64
12.2.2.2 bussbyte . . . . .	64
12.2.2.3 midishort . . . . .	65
12.2.2.4 midilong . . . . .	65
12.2.2.5 midipulse . . . . .	65
12.2.2.6 midibpm . . . . .	65
12.2.2.7 rterror_callback . . . . .	65
12.2.2.8 rtmidi_callback_t . . . . .	65
12.2.3 Enumeration Type Documentation . . . . .	66
12.2.3.1 wave_type_t . . . . .	66
12.2.3.2 seq_modifier_t . . . . .	66
12.2.3.3 seq_event_type_t . . . . .	67
12.2.3.4 seq_scroll_direction_t . . . . .	67
12.2.3.5 clock_e . . . . .	68
12.2.3.6 interaction_method_t . . . . .	68
12.2.3.7 c_music_scales . . . . .	68
12.2.3.8 draw_type_t . . . . .	69
12.2.3.9 mouse_action_e . . . . .	69
12.2.3.10 edit_action_t . . . . .	69
12.2.3.11 rtmidi_api . . . . .	70
12.2.4 Function Documentation . . . . .	70
12.2.4.1 swap() . . . . .	70
12.2.4.2 wave_type_name() . . . . .	71
12.2.4.3 extract_timing_numbers() . . . . .	71
12.2.4.4 pulses_to_string() . . . . .	72

12.2.4.5	<a href="#">pulses_to_measurestring()</a>	72
12.2.4.6	<a href="#">pulses_to_midi_measures()</a>	73
12.2.4.7	<a href="#">pulses_to_timestring()</a> [1/2]	73
12.2.4.8	<a href="#">pulses_to_timestring()</a> [2/2]	74
12.2.4.9	<a href="#">measurestring_to_pulses()</a>	74
12.2.4.10	<a href="#">midi_measures_to_pulses()</a>	75
12.2.4.11	<a href="#">timestring_to_pulses()</a> [1/2]	75
12.2.4.12	<a href="#">string_to_pulses()</a>	75
12.2.4.13	<a href="#">string_to_midibyte()</a>	76
12.2.4.14	<a href="#">shorten_file_spec()</a>	76
12.2.4.15	<a href="#">string_not_void()</a>	76
12.2.4.16	<a href="#">string_is_void()</a>	77
12.2.4.17	<a href="#">strings_match()</a>	77
12.2.4.18	<a href="#">log2_time_sig_value()</a>	78
12.2.4.19	<a href="#">tempo_us_to_bytes()</a>	78
12.2.4.20	<a href="#">zoom_power_of_2()</a>	78
12.2.4.21	<a href="#">bpm_from_tempo_us()</a>	79
12.2.4.22	<a href="#">tempo_us_from_bpm()</a>	79
12.2.4.23	<a href="#">pulse_length_us()</a>	80
12.2.4.24	<a href="#">delta_time_us_to_ticks()</a>	80
12.2.4.25	<a href="#">ticks_to_delta_time_us()</a>	81
12.2.4.26	<a href="#">clock_tick_duration_bogus()</a>	81
12.2.4.27	<a href="#">clock_ticks_from_ppqn()</a>	82
12.2.4.28	<a href="#">double_ticks_from_ppqn()</a>	82
12.2.4.29	<a href="#">pulses_per_measure()</a>	83
12.2.4.30	<a href="#">measures_to_ticks()</a>	83
12.2.4.31	<a href="#">wave_func()</a>	84
12.2.4.32	<a href="#">extract_port_names()</a>	84
12.2.4.33	<a href="#">extract_bus_name()</a>	85
12.2.4.34	<a href="#">extract_port_name()</a>	85

12.2.4.35 help_check()	85
12.2.4.36 parse_options_files()	86
12.2.4.37 parse_command_line_options()	87
12.2.4.38 write_options_files()	87
12.2.4.39 build_details()	87
12.2.4.40 message_concatenate()	88
12.2.4.41 info_message()	88
12.2.4.42 error_message()	88
12.2.4.43 to_string()	89
12.2.4.44 file_access()	89
12.2.4.45 file_exists()	89
12.2.4.46 file_readable()	90
12.2.4.47 file_writable()	90
12.2.4.48 file_accessible()	90
12.2.4.49 file_executable()	91
12.2.4.50 file_is_directory()	91
12.2.4.51 make_directory()	91
12.2.4.52 ppqn_is_valid()	92
12.2.4.53 jack_sync_callback()	92
12.2.4.54 jack_shutdown_callback()	93
12.2.4.55 jack_timebase_callback()	93
12.2.4.56 jack_transport_callback()	94
12.2.4.57 create_jack_client()	94
12.2.4.58 show_jack_statuses()	96
12.2.4.59 get_current_jack_position()	96
12.2.4.60 jack_session_callback()	96
12.2.4.61 invalid_key()	97
12.2.4.62 keyval_name()	97
12.2.4.63 keyval_normalize()	97
12.2.4.64 create_lash_driver()	98



12.2.4.65 lash_driver()	98
12.2.4.66 delete_lash_driver()	98
12.2.4.67 millisleep()	98
12.2.4.68 is_null_midipulse()	99
12.2.4.69 output_thread_func()	99
12.2.4.70 input_thread_func()	99
12.2.4.71 rc()	99
12.2.4.72 usr()	100
12.2.4.73 choose_ppqn()	100
12.2.4.74 timestring_to_pulses() [2/2]	100
12.2.4.75 jack_dummy_callback()	101
12.2.4.76 seq_app_name()	101
12.2.4.77 seq_client_name()	101
12.2.4.78 seq_version()	101
12.2.4.79 make_section_name()	101
12.2.4.80 font_render()	102
12.2.4.81 adjustment_dummy()	102
12.2.4.82 is_ctrl_key() [1/3]	102
12.2.4.83 is_shift_key() [1/3]	103
12.2.4.84 is_no_modifier()	103
12.2.4.85 is_ctrl_key() [2/3]	103
12.2.4.86 is_shift_key() [2/3]	103
12.2.4.87 is_ctrl_key() [3/3]	104
12.2.4.88 is_shift_key() [3/3]	104
12.2.4.89 is_ctrl_shift_key()	104
12.2.4.90 is_super_key()	104
12.2.4.91 test_widget_click()	105
12.2.4.92 update_mainwid_sequences()	105
12.2.4.93 update_perfeddit_sequences()	105
12.2.4.94 FF_RW_timeout()	105

12.2.4.95 clamp() [1/2]	105
12.2.4.96 clamp() [2/2]	106
12.2.4.97 silence_jack_errors()	106
12.2.4.98 silence_jack_info()	106
12.2.4.99 midi_api_name()	106
12.2.4.100midi_probe()	106
12.2.4.101midi_input_test()	107
12.2.4.102min()	107
12.2.4.103jack_process_rtmidi_input()	107
12.2.4.104jack_process_rtmidi_output()	108
12.2.4.105jack_process_io()	109
12.2.4.106jack_message_bit_bucket()	109
12.2.4.107midi_input_callback()	109
12.2.5 Variable Documentation	109
12.2.5.1 c_controller_names	109
12.2.5.2 EVENT_STATUS_BIT	109
12.2.5.3 EVENT_ANY	110
12.2.5.4 EVENT_NOTE_OFF	110
12.2.5.5 EVENT_NOTE_ON	110
12.2.5.6 EVENT_AFTERTOUCH	110
12.2.5.7 EVENT_CONTROL_CHANGE	110
12.2.5.8 EVENT_PROGRAM_CHANGE	110
12.2.5.9 EVENT_CHANNEL_PRESSURE	110
12.2.5.10 EVENT_PITCH_WHEEL	111
12.2.5.11 EVENT_MIDI_SYSEX	111
12.2.5.12 EVENT_MIDI_QUARTER_FRAME	111
12.2.5.13 EVENT_MIDI_SONG_POS	111
12.2.5.14 EVENT_MIDI_SONG_SELECT	111
12.2.5.15 EVENT_MIDI_SONG_F4	111
12.2.5.16 EVENT_MIDI_SONG_F5	112

12.2.5.17 EVENT_MIDI_TUNE_SELECT . . . . .	112
12.2.5.18 EVENT_MIDI_SYSEX_END . . . . .	112
12.2.5.19 EVENT_MIDI_SYSEX_CONTINUE . . . . .	112
12.2.5.20 EVENT_MIDI_CLOCK . . . . .	112
12.2.5.21 EVENT_MIDI_SONG_F9 . . . . .	112
12.2.5.22 EVENT_MIDI_START . . . . .	112
12.2.5.23 EVENT_MIDI_CONTINUE . . . . .	112
12.2.5.24 EVENT_MIDI_STOP . . . . .	113
12.2.5.25 EVENT_MIDI_SONG_FD . . . . .	113
12.2.5.26 EVENT_MIDI_ACTIVE_SENS . . . . .	113
12.2.5.27 EVENT_MIDI_RESET . . . . .	113
12.2.5.28 EVENT_MIDI_META . . . . .	113
12.2.5.29 EVENT_NULL_CHANNEL . . . . .	113
12.2.5.30 EVENT_GET_CHAN_MASK . . . . .	113
12.2.5.31 EVENT_CLEAR_CHAN_MASK . . . . .	114
12.2.5.32 EVENTS_ALL . . . . .	114
12.2.5.33 EVENTS_UNSELECTED . . . . .	114
12.2.5.34 c_midibus_output_size . . . . .	114
12.2.5.35 c_midibus_input_size . . . . .	114
12.2.5.36 c_midibus_sysex_chunk . . . . .	114
12.2.5.37 c_midibus . . . . .	115
12.2.5.38 c_midich . . . . .	116
12.2.5.39 c_midiclocks . . . . .	116
12.2.5.40 c_triggers . . . . .	116
12.2.5.41 c_notes . . . . .	116
12.2.5.42 c_timesig . . . . .	116
12.2.5.43 c_bpmtag . . . . .	116
12.2.5.44 c_triggers_new . . . . .	116
12.2.5.45 c_mutegroups . . . . .	116
12.2.5.46 c_midictrl . . . . .	117

12.2.5.47 c_musickey . . . . .	117
12.2.5.48 c_musicscale . . . . .	117
12.2.5.49 c_backsequence . . . . .	117
12.2.5.50 c_transpose . . . . .	117
12.2.5.51 c_perf_bp_mes . . . . .	117
12.2.5.52 c_perf_bw . . . . .	117
12.2.5.53 c_midi_track_ctrl . . . . .	118
12.2.5.54 c_midi_control_bpm_up . . . . .	118
12.2.5.55 c_midi_control_bpm_dn . . . . .	118
12.2.5.56 c_midi_control_ss_up . . . . .	118
12.2.5.57 c_midi_control_ss_dn . . . . .	118
12.2.5.58 c_midi_control_mod_replace . . . . .	118
12.2.5.59 c_midi_control_mod_snapshot . . . . .	119
12.2.5.60 c_midi_control_mod_queue . . . . .	119
12.2.5.61 c_midi_control_mod_gmute . . . . .	119
12.2.5.62 c_midi_control_mod_glearn . . . . .	119
12.2.5.63 c_midi_control_play_ss . . . . .	119
12.2.5.64 c_midi_controls . . . . .	119
12.2.5.65 c_midi_control_playback . . . . .	119
12.2.5.66 c_midi_control_record . . . . .	119
12.2.5.67 c_midi_control_solo . . . . .	120
12.2.5.68 c_midi_control_thru . . . . .	120
12.2.5.69 c_midi_control_bpm_page_up . . . . .	120
12.2.5.70 c_midi_control_bpm_page_dn . . . . .	120
12.2.5.71 c_midi_control_16 . . . . .	120
12.2.5.72 c_midi_control_17 . . . . .	120
12.2.5.73 c_midi_control_18 . . . . .	120
12.2.5.74 c_midi_control_19 . . . . .	120
12.2.5.75 c_midi_controls_extended . . . . .	121
12.2.5.76 g_midi_control_limit . . . . .	121

12.2.5.77 c_scales_policy . . . . .	121
12.2.5.78 c_scales_transpose_up . . . . .	121
12.2.5.79 c_scales_transpose_dn . . . . .	122
12.2.5.80 c_scales_text . . . . .	123
12.2.5.81 c_key_text . . . . .	123
12.2.5.82 c_interval_text . . . . .	123
12.2.5.83 c_chord_text . . . . .	123
12.2.5.84 c_chord_number . . . . .	123
12.2.5.85 c_chord_table_text . . . . .	123
12.2.5.86 c_chord_size . . . . .	123
12.2.5.87 c_chord_table . . . . .	124
12.2.5.88 c_max_instruments . . . . .	124
12.2.5.89 c_max_busses . . . . .	124
12.2.5.90 versiontext . . . . .	124
12.2.5.91 long_options . . . . .	124
12.2.5.92 s_arg_list . . . . .	124
12.2.5.93 s_help_1a . . . . .	125
12.2.5.94 s_help_1b . . . . .	125
12.2.5.95 s_help_2 . . . . .	125
12.2.5.96 s_help_3 . . . . .	125
12.2.5.97 s_help_4 . . . . .	125
12.2.5.98 s_build_alsamidi_support . . . . .	125
12.2.5.99 s_build_portmidi_support . . . . .	125
12.2.5.100s_build_rtmidi_support . . . . .	125
12.2.5.101s_build_highlight_empty . . . . .	126
12.2.5.102s_build_lash_support . . . . .	126
12.2.5.103s_build_jack_support . . . . .	126
12.2.5.104s_build_jack_session . . . . .	126
12.2.5.105s_event_editor . . . . .	126
12.2.5.106s_build_pause_support . . . . .	126

12.2.5.107s_build_use_event_map . . . . .	126
12.2.5.108s_build_chord_generator . . . . .	126
12.2.5.109s_build_edit_highlight . . . . .	127
12.2.5.110s_build_timesig_tempo . . . . .	127
12.2.5.111s_build_midi_vector . . . . .	127
12.2.5.112s_build_solid_grid . . . . .	127
12.2.5.113s_build_follow_progress . . . . .	127
12.2.5.114s_statistics_support . . . . .	127
12.2.5.115s_strip_empty_mutes . . . . .	127
12.2.5.116s_seq32_jack_support . . . . .	127
12.2.5.117s_seq32_transport . . . . .	128
12.2.5.118s_seq32_transpose . . . . .	128
12.2.5.119s_seq32_menu_buttons . . . . .	128
12.2.5.120s_seq32_lfo_support . . . . .	128
12.2.5.121s_debug_mode . . . . .	128
12.2.5.122s_status_pairs . . . . .	128
12.2.5.123s_character_mapping . . . . .	128
12.2.5.124s_global_lash_driver . . . . .	129
12.2.5.125c_status_replace . . . . .	129
12.2.5.126c_status_snapshot . . . . .	129
12.2.5.127c_status_queue . . . . .	130
12.2.5.128g_rc_settings . . . . .	130
12.2.5.129g_user_settings . . . . .	130
12.2.5.130s_handlesize [1/2] . . . . .	130
12.2.5.131s_jitter_amount . . . . .	130
12.2.5.132gs_mainwid_pointer . . . . .	130
12.2.5.133c_mainwid_x . . . . .	130
12.2.5.134c_mainwid_y . . . . .	131
12.2.5.135gs_perfedit_pointer_0 . . . . .	131
12.2.5.136gs_perfedit_pointer_1 . . . . .	131
12.2.5.137s_handlesize [2/2] . . . . .	131

<b>13 Data Structure Documentation</b>	<b>133</b>
13.1 seq64::AbstractPerfInput Class Reference	133
13.1.1 Constructor & Destructor Documentation	134
13.1.1.1 AbstractPerfInput()	134
13.1.1.2 ~AbstractPerfInput()	134
13.1.2 Member Function Documentation	135
13.1.2.1 on_button_press_event()	135
13.1.2.2 on_button_release_event()	135
13.1.2.3 on_motion_notify_event()	135
13.1.2.4 activate_adding()	135
13.1.2.5 handle_motion_key()	135
13.1.2.6 is_adding()	136
13.1.2.7 set_adding()	136
13.1.2.8 is_adding_pressed()	136
13.1.2.9 set_adding_pressed()	136
13.1.3 Friends And Related Function Documentation	136
13.1.3.1 perfroll	136
13.1.4 Field Documentation	136
13.1.4.1 m_adding	136
13.1.4.2 m_adding_pressed	137
13.2 seq64::automutex Class Reference	137
13.2.1 Detailed Description	137
13.2.2 Constructor & Destructor Documentation	137
13.2.2.1 automutex() [1/3]	137
13.2.2.2 automutex() [2/3]	137
13.2.2.3 automutex() [3/3]	137
13.2.2.4 ~automutex()	138
13.2.3 Member Function Documentation	138
13.2.3.1 operator=()	138
13.2.4 Field Documentation	138

13.2.4.1	<code>m_safety_mutex</code>	138
13.3	<code>seq64::busarray</code> Class Reference	138
13.3.1	Constructor & Destructor Documentation	140
13.3.1.1	<code>busarray()</code>	140
13.3.1.2	<code>~busarray()</code>	140
13.3.2	Member Function Documentation	140
13.3.2.1	<code>add()</code> [1/2]	140
13.3.2.2	<code>add()</code> [2/2]	141
13.3.2.3	<code>initialize()</code>	141
13.3.2.4	<code>count()</code>	141
13.3.2.5	<code>bus()</code>	142
13.3.2.6	<code>start()</code>	142
13.3.2.7	<code>stop()</code>	142
13.3.2.8	<code>continue_from()</code>	142
13.3.2.9	<code>init_clock()</code>	142
13.3.2.10	<code>clock()</code>	142
13.3.2.11	<code>sysex()</code>	143
13.3.2.12	<code>play()</code>	143
13.3.2.13	<code>set_clock()</code>	143
13.3.2.14	<code>set_all_clocks()</code>	144
13.3.2.15	<code>get_clock()</code>	144
13.3.2.16	<code>get_midi_bus_name()</code>	144
13.3.2.17	<code>print()</code>	145
13.3.2.18	<code>port_exit()</code>	145
13.3.2.19	<code>set_input()</code>	145
13.3.2.20	<code>set_all_inputs()</code>	146
13.3.2.21	<code>get_input()</code>	146
13.3.2.22	<code>is_system_port()</code>	146
13.3.2.23	<code>poll_for_midi()</code>	147
13.3.2.24	<code>get_midi_event()</code>	147



13.3.2.25 replacement_port()	147
13.3.3 Field Documentation	148
13.3.3.1 m_container	148
13.4 seq64::businfo Class Reference	148
13.4.1 Detailed Description	149
13.4.2 Constructor & Destructor Documentation	149
13.4.2.1 businfo() [1/3]	149
13.4.2.2 businfo() [2/3]	149
13.4.2.3 businfo() [3/3]	150
13.4.2.4 ~businfo()	150
13.4.3 Member Function Documentation	150
13.4.3.1 remove()	150
13.4.3.2 bus() [1/3]	150
13.4.3.3 bus() [2/3]	150
13.4.3.4 active()	151
13.4.3.5 initialize()	151
13.4.3.6 initialized()	151
13.4.3.7 init_clock() [1/3]	151
13.4.3.8 init_input() [1/2]	152
13.4.3.9 bus() [3/3]	152
13.4.3.10 activate()	152
13.4.3.11 deactivate()	152
13.4.3.12 init_clock() [2/3]	152
13.4.3.13 init_input() [2/2]	152
13.4.3.14 start()	152
13.4.3.15 stop()	153
13.4.3.16 continue_from()	153
13.4.3.17 init_clock() [3/3]	153
13.4.3.18 clock()	153
13.4.3.19 sysex()	153

13.4.4	Friends And Related Function Documentation . . . . .	153
13.4.4.1	busarray . . . . .	153
13.4.5	Field Documentation . . . . .	153
13.4.5.1	m_bus . . . . .	154
13.4.5.2	m_active . . . . .	154
13.4.5.3	m_initialized . . . . .	154
13.4.5.4	m_init_clock . . . . .	154
13.4.5.5	m_init_input . . . . .	154
13.5	seq64::click Class Reference . . . . .	154
13.5.1	Detailed Description . . . . .	155
13.5.2	Constructor & Destructor Documentation . . . . .	156
13.5.2.1	click() [1/3] . . . . .	156
13.5.2.2	click() [2/3] . . . . .	156
13.5.2.3	click() [3/3] . . . . .	156
13.5.3	Member Function Documentation . . . . .	156
13.5.3.1	operator=() . . . . .	157
13.5.3.2	is_press() . . . . .	157
13.5.3.3	is_left() . . . . .	157
13.5.3.4	is_middle() . . . . .	157
13.5.3.5	is_right() . . . . .	157
13.5.3.6	x() . . . . .	157
13.5.3.7	y() . . . . .	158
13.5.3.8	button() . . . . .	158
13.5.3.9	modifier() . . . . .	158
13.5.3.10	mod_control() . . . . .	158
13.5.3.11	mod_control_shift() . . . . .	158
13.5.3.12	mod_super() . . . . .	158
13.5.4	Field Documentation . . . . .	158
13.5.4.1	m_is_press . . . . .	158
13.5.4.2	m_x . . . . .	159

13.5.4.3	<a href="#">m_y</a>	159
13.5.4.4	<a href="#">m_button</a>	159
13.5.4.5	<a href="#">m_modifier</a>	159
13.6	<a href="#">seq64::condition_var Class Reference</a>	159
13.6.1	<a href="#">Detailed Description</a>	160
13.6.2	<a href="#">Constructor &amp; Destructor Documentation</a>	160
13.6.2.1	<a href="#">condition_var()</a>	160
13.6.3	<a href="#">Member Function Documentation</a>	160
13.6.3.1	<a href="#">wait()</a>	160
13.6.3.2	<a href="#">signal()</a>	161
13.6.4	<a href="#">Field Documentation</a>	161
13.6.4.1	<a href="#">sm_cond</a>	161
13.6.4.2	<a href="#">m_cond</a>	161
13.7	<a href="#">seq64::configfile Class Reference</a>	161
13.7.1	<a href="#">Constructor &amp; Destructor Documentation</a>	163
13.7.1.1	<a href="#">configfile()</a>	163
13.7.1.2	<a href="#">~configfile()</a>	163
13.7.2	<a href="#">Member Function Documentation</a>	163
13.7.2.1	<a href="#">next_data_line()</a>	163
13.7.2.2	<a href="#">line_after()</a>	164
13.7.2.3	<a href="#">parse()</a>	164
13.7.2.4	<a href="#">write()</a>	164
13.7.2.5	<a href="#">get_error_message()</a>	165
13.7.2.6	<a href="#">set_error_message()</a>	165
13.7.3	<a href="#">Field Documentation</a>	165
13.7.3.1	<a href="#">m_error_message</a>	165
13.7.3.2	<a href="#">m_name</a>	165
13.7.3.3	<a href="#">m_d</a>	165
13.7.3.4	<a href="#">m_line</a>	165
13.8	<a href="#">seq64::editable_event Class Reference</a>	166

13.8.1 Detailed Description . . . . .	169
13.8.2 Member Enumeration Documentation . . . . .	169
13.8.2.1 category_t . . . . .	169
13.8.2.2 timestamp_format_t . . . . .	170
13.8.3 Constructor & Destructor Documentation . . . . .	170
13.8.3.1 editable_event() [1/4] . . . . .	170
13.8.3.2 editable_event() [2/4] . . . . .	171
13.8.3.3 editable_event() [3/4] . . . . .	171
13.8.3.4 editable_event() [4/4] . . . . .	171
13.8.3.5 ~editable_event() . . . . .	171
13.8.4 Member Function Documentation . . . . .	172
13.8.4.1 value_to_name() . . . . .	172
13.8.4.2 name_to_value() . . . . .	172
13.8.4.3 operator=() . . . . .	172
13.8.4.4 parent() . . . . .	173
13.8.4.5 category() [1/3] . . . . .	173
13.8.4.6 category() [2/3] . . . . .	173
13.8.4.7 category_string() . . . . .	173
13.8.4.8 category() [3/3] . . . . .	173
13.8.4.9 timestamp_string() . . . . .	173
13.8.4.10 timestamp() [1/3] . . . . .	174
13.8.4.11 timestamp() [2/3] . . . . .	174
13.8.4.12 timestamp() [3/3] . . . . .	174
13.8.4.13 time_as_pulses() . . . . .	174
13.8.4.14 time_as_measures() . . . . .	174
13.8.4.15 time_as_minutes() . . . . .	175
13.8.4.16 set_status_from_string() . . . . .	175
13.8.4.17 format_timestamp() . . . . .	175
13.8.4.18 stock_event_string() . . . . .	175
13.8.4.19 status_string() . . . . .	176

13.8.4.20 meta_string()	176
13.8.4.21 seqspec_string()	176
13.8.4.22 channel_string()	176
13.8.4.23 data_string()	176
13.8.4.24 analyze()	176
13.8.5 Field Documentation	177
13.8.5.1 sm_category_names	177
13.8.5.2 sm_channel_event_names	177
13.8.5.3 sm_system_event_names	177
13.8.5.4 sm_meta_event_names	177
13.8.5.5 sm_prop_event_names	177
13.8.5.6 sm_category_arrays	178
13.8.5.7 m_parent	178
13.8.5.8 m_category	178
13.8.5.9 m_name_category	178
13.8.5.10 m_format_timestamp	178
13.8.5.11 m_name_timestamp	178
13.8.5.12 m_name_status	178
13.8.5.13 m_name_meta	179
13.8.5.14 m_name_seqspec	179
13.8.5.15 m_name_channel	179
13.8.5.16 m_name_data	179
13.9 seq64::editable_events Class Reference	179
13.9.1 Member Typedef Documentation	181
13.9.1.1 Key	181
13.9.1.2 EventsPair	181
13.9.1.3 Events	181
13.9.1.4 iterator	181
13.9.1.5 const_iterator	182
13.9.2 Constructor & Destructor Documentation	182

13.9.2.1	<a href="#">editable_events()</a> [1/3]	182
13.9.2.2	<a href="#">editable_events()</a> [2/3]	182
13.9.2.3	<a href="#">editable_events()</a> [3/3]	182
13.9.2.4	<a href="#">~editable_events()</a>	182
13.9.3	Member Function Documentation	183
13.9.3.1	<a href="#">operator=()</a>	183
13.9.3.2	<a href="#">timing()</a>	183
13.9.3.3	<a href="#">string_to_pulses()</a>	183
13.9.3.4	<a href="#">load_events()</a>	183
13.9.3.5	<a href="#">save_events()</a>	184
13.9.3.6	<a href="#">events()</a>	184
13.9.3.7	<a href="#">begin()</a> [1/2]	184
13.9.3.8	<a href="#">begin()</a> [2/2]	184
13.9.3.9	<a href="#">end()</a> [1/2]	184
13.9.3.10	<a href="#">end()</a> [2/2]	184
13.9.3.11	<a href="#">dref()</a> [1/2]	184
13.9.3.12	<a href="#">dref()</a> [2/2]	185
13.9.3.13	<a href="#">count()</a>	185
13.9.3.14	<a href="#">add()</a> [1/2]	185
13.9.3.15	<a href="#">add()</a> [2/2]	185
13.9.3.16	<a href="#">replace()</a>	186
13.9.3.17	<a href="#">remove()</a>	186
13.9.3.18	<a href="#">clear()</a>	186
13.9.3.19	<a href="#">current_event()</a> [1/2]	186
13.9.3.20	<a href="#">current_event()</a> [2/2]	186
13.9.4	Friends And Related Function Documentation	187
13.9.4.1	<a href="#">eventslots</a>	187
13.9.5	Field Documentation	187
13.9.5.1	<a href="#">m_events</a>	187
13.9.5.2	<a href="#">m_current_event</a>	187

13.9.5.3	<code>m_sequence</code>	187
13.9.5.4	<code>m_midi_parameters</code>	187
13.10	<code>seq64::event</code> Class Reference	188
13.10.1	Detailed Description	192
13.10.2	Member Typedef Documentation	192
13.10.2.1	<code>SysexContainer</code>	192
13.10.3	Constructor & Destructor Documentation	192
13.10.3.1	<code>event()</code> [1/2]	192
13.10.3.2	<code>event()</code> [2/2]	193
13.10.3.3	<code>~event()</code>	193
13.10.4	Member Function Documentation	193
13.10.4.1	<code>operator=()</code>	193
13.10.4.2	<code>operator&lt;()</code>	194
13.10.4.3	<code>set_timestamp()</code>	194
13.10.4.4	<code>get_timestamp()</code>	195
13.10.4.5	<code>get_channel()</code>	195
13.10.4.6	<code>check_channel()</code>	195
13.10.4.7	<code>is_channel_msg()</code>	195
13.10.4.8	<code>is_one_byte_msg()</code>	196
13.10.4.9	<code>is_two_byte_msg()</code>	196
13.10.4.10	<code>is_note_msg()</code>	197
13.10.4.11	<code>is_strict_note_msg()</code>	197
13.10.4.12	<code>is_desired_cc_or_not_cc()</code>	197
13.10.4.13	<code>mod_timestamp()</code>	198
13.10.4.14	<code>set_status()</code> [1/2]	198
13.10.4.15	<code>set_status()</code> [2/2]	199
13.10.4.16	<code>set_status_keep_channel()</code>	199
13.10.4.17	<code>set_channel()</code>	199
13.10.4.18	<code>get_status()</code>	200
13.10.4.19	<code>non_cc_match()</code>	200

13.10.4.20cc_match()	200
13.10.4.21set_data() [1/2]	200
13.10.4.22set_data() [2/2]	201
13.10.4.23get_data()	201
13.10.4.24increment_data1()	201
13.10.4.25decrement_data1()	201
13.10.4.26increment_data2()	201
13.10.4.27decrement_data2()	202
13.10.4.28append_sysex() [1/2]	202
13.10.4.29append_sysex() [2/2]	202
13.10.4.30restart_sysex()	202
13.10.4.31get_sysex() [1/2]	202
13.10.4.32get_sysex() [2/2]	203
13.10.4.33set_sysex_size()	203
13.10.4.34get_sysex_size()	203
13.10.4.35link()	203
13.10.4.36get_linked()	203
13.10.4.37is_linked()	203
13.10.4.38clear_link()	203
13.10.4.39paint()	204
13.10.4.40unpaint()	204
13.10.4.41is_painted()	204
13.10.4.42mark()	204
13.10.4.43unmark()	204
13.10.4.44is_marked()	204
13.10.4.45select()	204
13.10.4.46unselect()	204
13.10.4.47is_selected()	205
13.10.4.48make_clock()	205
13.10.4.49data()	205



13.10.4.50	<code>get_note()</code>	205
13.10.4.51	<code>set_note()</code>	205
13.10.4.52	<code>transpose_note()</code>	205
13.10.4.53	<code>get_note_velocity()</code>	206
13.10.4.54	<code>set_note_velocity()</code>	206
13.10.4.55	<code>s_note_on()</code>	206
13.10.4.56	<code>s_note_off()</code>	206
13.10.4.57	<code>s_note()</code>	207
13.10.4.58	<code>s_note_off_recorded()</code>	207
13.10.4.59	<code>print()</code>	207
13.10.4.60	<code>get_rank()</code>	207
13.10.5	Field Documentation	207
13.10.5.1	<code>m_timestamp</code>	208
13.10.5.2	<code>m_status</code>	208
13.10.5.3	<code>m_channel</code>	208
13.10.5.4	<code>m_data</code>	208
13.10.5.5	<code>m_sysex</code>	208
13.10.5.6	<code>m_sysex_size</code>	208
13.10.5.7	<code>m_linked</code>	208
13.10.5.8	<code>m_has_link</code>	209
13.10.5.9	<code>m_selected</code>	209
13.10.5.10	<code>m_marked</code>	209
13.10.5.11	<code>m_painted</code>	209
13.11	<code>seq64::event_list::event_key</code> Class Reference	209
13.11.1	Detailed Description	209
13.11.2	Constructor & Destructor Documentation	210
13.11.2.1	<code>event_key()</code> [1/2]	210
13.11.2.2	<code>event_key()</code> [2/2]	210
13.11.3	Member Function Documentation	210
13.11.3.1	<code>operator&lt;()</code>	210

13.11.4 Field Documentation . . . . .	211
13.11.4.1 m_timestamp . . . . .	211
13.11.4.2 m_rank . . . . .	211
13.12seq64::event_list Class Reference . . . . .	211
13.12.1 Detailed Description . . . . .	213
13.12.2 Member Typedef Documentation . . . . .	213
13.12.2.1 Events . . . . .	213
13.12.2.2 EventsPair . . . . .	214
13.12.2.3 iterator . . . . .	214
13.12.2.4 const_iterator . . . . .	214
13.12.3 Constructor & Destructor Documentation . . . . .	214
13.12.3.1 event_list() [1/2] . . . . .	214
13.12.3.2 event_list() [2/2] . . . . .	214
13.12.3.3 ~event_list() . . . . .	214
13.12.4 Member Function Documentation . . . . .	214
13.12.4.1 operator=() . . . . .	215
13.12.4.2 begin() [1/2] . . . . .	215
13.12.4.3 begin() [2/2] . . . . .	215
13.12.4.4 end() [1/2] . . . . .	215
13.12.4.5 end() [2/2] . . . . .	215
13.12.4.6 count() . . . . .	215
13.12.4.7 empty() . . . . .	215
13.12.4.8 add() . . . . .	215
13.12.4.9 append() . . . . .	216
13.12.4.10push_back() . . . . .	216
13.12.4.11is_modified() . . . . .	217
13.12.4.12unmodify() . . . . .	217
13.12.4.13remove() . . . . .	217
13.12.4.14clear() . . . . .	217
13.12.4.15merge() . . . . .	217

13.12.4.16	sort()	218
13.12.4.17	dref() [1/2]	218
13.12.4.18	dref() [2/2]	218
13.12.4.19	link_new()	219
13.12.4.20	clear_links()	219
13.12.4.21	verify_and_link()	219
13.12.4.22	mark_selected()	219
13.12.4.23	mark_out_of_range()	219
13.12.4.24	mark_all()	220
13.12.4.25	unmark_all()	220
13.12.4.26	remove_marked()	220
13.12.4.27	unpaint_all()	220
13.12.4.28	count_selected_notes()	220
13.12.4.29	any_selected_notes()	221
13.12.4.30	count_selected_events()	221
13.12.4.31	select_all()	221
13.12.4.32	inselect_all()	221
13.12.4.33	print()	221
13.12.4.34	events()	222
13.12.5	Friends And Related Function Documentation	222
13.12.5.1	editable_events	222
13.12.5.2	midifile	222
13.12.5.3	midi_container	222
13.12.5.4	midi_splitter	222
13.12.5.5	sequence	222
13.12.6	Field Documentation	222
13.12.6.1	m_events	222
13.12.6.2	m_is_modified	223
13.13	seq64::eventedit Class Reference	223
13.13.1	Constructor & Destructor Documentation	227

13.13.1.1 eventedit()	227
13.13.1.2 ~eventedit()	228
13.13.2 Member Function Documentation	229
13.13.2.1 enqueue_draw()	229
13.13.2.2 set_seq_title()	229
13.13.2.3 set_seq_time_sig()	229
13.13.2.4 set_seq_ppqn()	229
13.13.2.5 set_seq_count()	229
13.13.2.6 set_event_category()	230
13.13.2.7 set_event_timestamp()	230
13.13.2.8 set_event_name()	230
13.13.2.9 set_event_data_0()	230
13.13.2.10 set_event_data_1()	231
13.13.2.11 perf_modify()	231
13.13.2.12 set_dirty()	231
13.13.2.13 v_adjustment() [1/2]	231
13.13.2.14 v_adjustment() [2/2]	231
13.13.2.15 change_focus()	232
13.13.2.16 close_out()	232
13.13.2.17 handle_close()	232
13.13.2.18 handle_delete()	232
13.13.2.19 handle_insert()	233
13.13.2.20 handle_modify()	233
13.13.2.21 handle_save()	233
13.13.2.22 handle_cancel()	233
13.13.2.23 on_realize()	233
13.13.2.24 on_set_focus()	233
13.13.2.25 on_focus_in_event()	234
13.13.2.26 on_focus_out_event()	234
13.13.2.27 on_key_press_event()	234

13.13.2.28on_delete_event()	234
13.13.3 Friends And Related Function Documentation	235
13.13.3.1 eventslots	235
13.13.4 Field Documentation	235
13.13.4.1 m_table	235
13.13.4.2 m_vadjust	235
13.13.4.3 m_vscroll	235
13.13.4.4 m_eventslots	235
13.13.4.5 m_htopbox	236
13.13.4.6 m_showbox	236
13.13.4.7 m_editbox	236
13.13.4.8 m_optsbox	236
13.13.4.9 m_bottbox	236
13.13.4.10m_rightbox	236
13.13.4.11m_button_del	236
13.13.4.12m_button_ins	236
13.13.4.13m_button_modify	237
13.13.4.14m_button_save	237
13.13.4.15m_button_cancel	237
13.13.4.16m_label_seq_name	237
13.13.4.17m_label_time_sig	237
13.13.4.18m_label_ppqn	237
13.13.4.19m_label_channel	237
13.13.4.20m_label_ev_count	237
13.13.4.21m_label_spacer	238
13.13.4.22m_label_modified	238
13.13.4.23m_label_category	238
13.13.4.24m_entry_ev_timestamp	238
13.13.4.25m_entry_ev_name	238
13.13.4.26m_entry_ev_data_0	238

13.13.4.27	m_entry_ev_data_1	238
13.13.4.28	m_label_time_fmt	238
13.13.4.29	m_label_right	239
13.13.4.30	m_seq	239
13.13.4.31	m_have_focus	239
13.14	seq64::eventslots Class Reference	239
13.14.1	Constructor & Destructor Documentation	243
13.14.1.1	eventslots()	244
13.14.1.2	~eventslots()	244
13.14.2	Member Function Documentation	244
13.14.2.1	event_count()	244
13.14.2.2	line_count()	244
13.14.2.3	line_maximum()	244
13.14.2.4	line_increment()	244
13.14.2.5	top_index()	244
13.14.2.6	current_index()	245
13.14.2.7	pager_index()	245
13.14.2.8	load_events()	245
13.14.2.9	set_current_event()	245
13.14.2.10	insert_event() [1/2]	245
13.14.2.11	insert_event() [2/2]	246
13.14.2.12	delete_current_event()	247
13.14.2.13	modify_current_event()	248
13.14.2.14	save_events()	248
13.14.2.15	select_event()	248
13.14.2.16	set_text()	249
13.14.2.17	enqueue_draw()	249
13.14.2.18	convert_y()	249
13.14.2.19	draw_event()	250
13.14.2.20	draw_events()	250

13.14.2.21	<a href="#">change_vert()</a>	251
13.14.2.22	<a href="#">page_movement()</a>	251
13.14.2.23	<a href="#">page_topper()</a>	251
13.14.2.24	<a href="#">decrement_top()</a>	252
13.14.2.25	<a href="#">increment_top()</a>	252
13.14.2.26	<a href="#">decrement_current()</a>	252
13.14.2.27	<a href="#">increment_current()</a>	252
13.14.2.28	<a href="#">decrement_bottom()</a>	253
13.14.2.29	<a href="#">increment_bottom()</a>	253
13.14.2.30	<a href="#">on_realize()</a>	253
13.14.2.31	<a href="#">on_expose_event()</a>	253
13.14.2.32	<a href="#">on_button_press_event()</a>	253
13.14.2.33	<a href="#">on_button_release_event()</a>	253
13.14.2.34	<a href="#">on_focus_in_event()</a>	254
13.14.2.35	<a href="#">on_focus_out_event()</a>	254
13.14.2.36	<a href="#">on_scroll_event()</a>	254
13.14.2.37	<a href="#">on_size_allocate()</a>	254
13.14.2.38	<a href="#">on_move_up()</a>	254
13.14.2.39	<a href="#">on_move_down()</a>	254
13.14.2.40	<a href="#">on_frame_up()</a>	254
13.14.2.41	<a href="#">on_frame_down()</a>	255
13.14.2.42	<a href="#">on_frame_home()</a>	255
13.14.2.43	<a href="#">on_frame_end()</a>	255
13.14.3	<a href="#">Friends And Related Function Documentation</a>	255
13.14.3.1	<a href="#">eventedit</a>	255
13.14.4	<a href="#">Field Documentation</a>	255
13.14.4.1	<a href="#">m_parent</a>	255
13.14.4.2	<a href="#">m_seq</a>	255
13.14.4.3	<a href="#">m_event_container</a>	255
13.14.4.4	<a href="#">m_slots_chars</a>	256

13.14.4.5 m_char_w . . . . .	256
13.14.4.6 m_setbox_w . . . . .	256
13.14.4.7 m_slots_x . . . . .	256
13.14.4.8 m_slots_y . . . . .	256
13.14.4.9 m_event_count . . . . .	256
13.14.4.10m_line_count . . . . .	256
13.14.4.11m_line_maximum . . . . .	257
13.14.4.12m_line_overlap . . . . .	257
13.14.4.13m_top_index . . . . .	257
13.14.4.14m_current_index . . . . .	257
13.14.4.15m_top_iterator . . . . .	257
13.14.4.16m_bottom_iterator . . . . .	257
13.14.4.17m_current_iterator . . . . .	257
13.14.4.18m_pager_index . . . . .	258
13.15seq64::font Class Reference . . . . .	258
13.15.1 Member Enumeration Documentation . . . . .	259
13.15.1.1 Color . . . . .	259
13.15.2 Constructor & Destructor Documentation . . . . .	259
13.15.2.1 font() . . . . .	260
13.15.3 Member Function Documentation . . . . .	260
13.15.3.1 init() . . . . .	260
13.15.3.2 render_string_on_drawable() . . . . .	260
13.15.3.3 char_width() . . . . .	261
13.15.3.4 char_height() . . . . .	261
13.15.3.5 padded_height() . . . . .	261
13.15.4 Field Documentation . . . . .	261
13.15.4.1 m_use_new_font . . . . .	261
13.15.4.2 m_cell_w . . . . .	261
13.15.4.3 m_cell_h . . . . .	261
13.15.4.4 m_font_w . . . . .	261



13.15.4.5 m_font_h . . . . .	262
13.15.4.6 m_offset . . . . .	262
13.15.4.7 m_padded_h . . . . .	262
13.15.4.8 m_pixmap . . . . .	262
13.15.4.9 m_black_pixmap . . . . .	262
13.15.4.10m_white_pixmap . . . . .	262
13.15.4.11m_b_on_y_pixmap . . . . .	262
13.15.4.12m_y_on_b_pixmap . . . . .	263
13.15.4.13m_b_on_c_pixmap . . . . .	263
13.15.4.14m_c_on_b_pixmap . . . . .	263
13.15.4.15m_clip_mask . . . . .	263
13.16seq64::FruityPerfInput Class Reference . . . . .	263
13.16.1 Constructor & Destructor Documentation . . . . .	265
13.16.1.1 FruityPerfInput() . . . . .	265
13.16.2 Member Function Documentation . . . . .	265
13.16.2.1 on_button_press_event() . . . . .	265
13.16.2.2 on_button_release_event() . . . . .	266
13.16.2.3 on_motion_notify_event() . . . . .	266
13.16.2.4 update_mouse_pointer() . . . . .	267
13.16.2.5 on_left_button_pressed() . . . . .	267
13.16.2.6 on_right_button_pressed() . . . . .	267
13.16.2.7 activate_adding() . . . . .	268
13.16.2.8 handle_motion_key() . . . . .	268
13.16.3 Friends And Related Function Documentation . . . . .	268
13.16.3.1 perfroll . . . . .	268
13.16.4 Field Documentation . . . . .	268
13.16.4.1 m_current_x . . . . .	268
13.16.4.2 m_current_y . . . . .	269
13.17seq64::FruitySeqEventInput Struct Reference . . . . .	269
13.17.1 Constructor & Destructor Documentation . . . . .	269

13.17.1.1 FruitySeqEventInput()	269
13.17.2 Member Function Documentation	269
13.17.2.1 update_mouse_pointer()	269
13.17.2.2 on_button_press_event()	270
13.17.2.3 on_button_release_event()	271
13.17.2.4 on_motion_notify_event()	271
13.17.3 Field Documentation	271
13.17.3.1 m_justselected_one	271
13.17.3.2 m_is_drag_pasting_start	271
13.17.3.3 m_is_drag_pasting	272
13.18seq64::FruitySeqRollInput Class Reference	272
13.18.1 Constructor & Destructor Documentation	272
13.18.1.1 FruitySeqRollInput()	272
13.18.2 Member Function Documentation	272
13.18.2.1 update_mouse_pointer()	272
13.18.2.2 on_button_press_event()	273
13.18.2.3 on_button_release_event()	273
13.18.2.4 on_motion_notify_event()	273
13.18.3 Field Documentation	274
13.18.3.1 m_erase_painting	274
13.18.3.2 m_drag_paste_start_pos	274
13.19seq64::gui_assistant Class Reference	274
13.19.1 Detailed Description	276
13.19.2 Constructor & Destructor Documentation	276
13.19.2.1 gui_assistant()	276
13.19.2.2 ~gui_assistant()	276
13.19.3 Member Function Documentation	276
13.19.3.1 quit()	276
13.19.3.2 jack_idle_connect()	276
13.19.3.3 lash_timeout_connect()	277

13.19.3.4 keys() [1/2]	277
13.19.3.5 keys() [2/2]	277
13.19.4 Field Documentation	277
13.19.4.1 m_keys_perform	277
13.20seq64::gui_assistant_gtk2 Class Reference	277
13.20.1 Constructor & Destructor Documentation	278
13.20.1.1 gui_assistant_gtk2()	279
13.20.1.2 ~gui_assistant_gtk2()	279
13.20.2 Member Function Documentation	279
13.20.2.1 quit()	279
13.20.2.2 lash_timeout_connect()	279
13.20.2.3 jack_idle_connect()	279
13.20.3 Field Documentation	279
13.20.3.1 sm_internal_keys	280
13.21seq64::gui_drawingarea_gtk2 Class Reference	280
13.21.1 Detailed Description	283
13.21.2 Constructor & Destructor Documentation	283
13.21.2.1 gui_drawingarea_gtk2() [1/3]	283
13.21.2.2 gui_drawingarea_gtk2() [2/3]	283
13.21.2.3 gui_drawingarea_gtk2() [3/3]	284
13.21.2.4 ~gui_drawingarea_gtk2()	284
13.21.3 Member Function Documentation	284
13.21.3.1 operator=()	284
13.21.3.2 window_x()	284
13.21.3.3 window_y()	284
13.21.3.4 current_x()	284
13.21.3.5 current_y()	285
13.21.3.6 drop_x()	285
13.21.3.7 drop_y()	285
13.21.3.8 force_draw()	285

13.21.3.9 <code>perf()</code> . . . . .	285
13.21.3.10 <code>clear_window()</code> . . . . .	285
13.21.3.11 <code>set_line()</code> . . . . .	285
13.21.3.12 <code>draw_line()</code> [1/6] . . . . .	286
13.21.3.13 <code>draw_line()</code> [2/6] . . . . .	286
13.21.3.14 <code>draw_line_on_pixmap()</code> [1/2] . . . . .	286
13.21.3.15 <code>draw_line_on_pixmap()</code> [2/2] . . . . .	287
13.21.3.16 <code>draw_line()</code> [3/6] . . . . .	287
13.21.3.17 <code>draw_line()</code> [4/6] . . . . .	288
13.21.3.18 <code>draw_line()</code> [5/6] . . . . .	288
13.21.3.19 <code>draw_line()</code> [6/6] . . . . .	288
13.21.3.20 <code>render_string()</code> . . . . .	289
13.21.3.21 <code>render_string_on_pixmap()</code> . . . . .	289
13.21.3.22 <code>draw_rectangle()</code> [1/6] . . . . .	290
13.21.3.23 <code>draw_rectangle()</code> [2/6] . . . . .	290
13.21.3.24 <code>draw_rectangle()</code> [3/6] . . . . .	290
13.21.3.25 <code>draw_rectangle()</code> [4/6] . . . . .	291
13.21.3.26 <code>draw_rectangle()</code> [5/6] . . . . .	291
13.21.3.27 <code>draw_rectangle()</code> [6/6] . . . . .	292
13.21.3.28 <code>draw_rectangle_on_pixmap()</code> [1/2] . . . . .	292
13.21.3.29 <code>draw_rectangle_on_pixmap()</code> [2/2] . . . . .	293
13.21.3.30 <code>draw_normal_rectangle_on_pixmap()</code> . . . . .	293
13.21.3.31 <code>draw_drawable()</code> . . . . .	294
13.21.3.32 <code>scroll_hadjust()</code> . . . . .	294
13.21.3.33 <code>scroll_vadjust()</code> . . . . .	294
13.21.3.34 <code>scroll_hset()</code> . . . . .	295
13.21.3.35 <code>scroll_vset()</code> . . . . .	295
13.21.3.36 <code>set_current_drop_x()</code> . . . . .	295
13.21.3.37 <code>set_current_drop_y()</code> . . . . .	295
13.21.3.38 <code>gtk_drawarea_init()</code> . . . . .	296

13.21.3.39on_realize()	296
13.21.4 Field Documentation	296
13.21.4.1 m_gc	296
13.21.4.2 m_window	296
13.21.4.3 m_vadjust	296
13.21.4.4 m_hadjust	296
13.21.4.5 m_pixmap	296
13.21.4.6 m_background	297
13.21.4.7 m_foreground	297
13.21.4.8 m_mainperf	297
13.21.4.9 m_window_x	297
13.21.4.10m_window_y	297
13.21.4.11m_current_x	297
13.21.4.12m_current_y	297
13.21.4.13m_drop_x	298
13.21.4.14m_drop_y	298
13.22seq64::gui_palette_gtk2 Class Reference	298
13.22.1 Detailed Description	301
13.22.2 Member Typedef Documentation	301
13.22.2.1 Color	301
13.22.3 Constructor & Destructor Documentation	302
13.22.3.1 gui_palette_gtk2()	302
13.22.3.2 ~gui_palette_gtk2()	302
13.22.4 Member Function Documentation	302
13.22.4.1 load_inverse_palette()	302
13.22.4.2 is_inverse()	303
13.22.4.3 line_color()	303
13.22.4.4 progress_color()	303
13.22.4.5 black()	303
13.22.4.6 dark_red()	303

13.22.4.7 dark_green()	303
13.22.4.8 dark_orange()	303
13.22.4.9 dark_blue()	304
13.22.4.10dark_magenta()	304
13.22.4.11dark_cyan()	304
13.22.4.12white()	304
13.22.4.13grey()	304
13.22.4.14dark_grey()	304
13.22.4.15light_grey()	304
13.22.4.16red()	304
13.22.4.17orange()	305
13.22.4.18yellow()	305
13.22.4.19green()	305
13.22.4.20blue()	305
13.22.4.21black_paint()	305
13.22.4.22white_paint()	305
13.22.4.23black_key()	305
13.22.4.24white_key()	305
13.22.4.25bg_color() [1/2]	306
13.22.4.26bg_color() [2/2]	306
13.22.4.27fg_color() [1/2]	306
13.22.4.28fg_color() [2/2]	306
13.22.5 Field Documentation	306
13.22.5.1 m_is_inverse	306
13.22.5.2 m_black	306
13.22.5.3 m_dk_red	306
13.22.5.4 m_dk_green	307
13.22.5.5 m_dk_orange	307
13.22.5.6 m_dk_blue	307
13.22.5.7 m_dk_magenta	307

13.22.5.8 m_dk_cyan . . . . .	307
13.22.5.9 m_red . . . . .	307
13.22.5.10m_white . . . . .	307
13.22.5.11m_orange . . . . .	307
13.22.5.12m_yellow . . . . .	308
13.22.5.13m_green . . . . .	308
13.22.5.14m_blue . . . . .	308
13.22.5.15m_grey . . . . .	308
13.22.5.16m_dk_grey . . . . .	308
13.22.5.17m_lt_grey . . . . .	308
13.22.5.18m_blk_paint . . . . .	308
13.22.5.19m_wht_paint . . . . .	308
13.22.5.20m_blk_key . . . . .	309
13.22.5.21m_wht_key . . . . .	309
13.22.5.22m_line_color . . . . .	309
13.22.5.23m_progress_color . . . . .	309
13.22.5.24m_bg_color . . . . .	309
13.22.5.25m_fg_color . . . . .	309
13.23seq64::gui_window_gtk2 Class Reference . . . . .	310
13.23.1 Constructor & Destructor Documentation . . . . .	311
13.23.1.1 gui_window_gtk2() . . . . .	311
13.23.1.2 ~gui_window_gtk2() . . . . .	312
13.23.2 Member Function Documentation . . . . .	312
13.23.2.1 perf() . . . . .	312
13.23.2.2 quit() . . . . .	312
13.23.2.3 redraw_period_ms() . . . . .	312
13.23.2.4 is_realized() . . . . .	312
13.23.2.5 scroll_hadjust() . . . . .	312
13.23.2.6 scroll_vadjust() . . . . .	313
13.23.2.7 scroll_hset() . . . . .	313

13.23.2.8 scroll_vset()	313
13.23.2.9 on_realize()	313
13.23.3 Field Documentation	314
13.23.3.1 m_mainperf	314
13.23.3.2 m_window_x	314
13.23.3.3 m_window_y	314
13.23.3.4 m_redraw_period_ms	314
13.23.3.5 m_is_realized	314
13.24seq64::jack_assistant Class Reference	314
13.24.1 Constructor & Destructor Documentation	318
13.24.1.1 jack_assistant()	318
13.24.1.2 ~jack_assistant()	318
13.24.2 Member Function Documentation	319
13.24.2.1 parent() [1/2]	319
13.24.2.2 parent() [2/2]	319
13.24.2.3 is_running()	319
13.24.2.4 is_master()	319
13.24.2.5 get_ppqn()	319
13.24.2.6 get_beat_width()	319
13.24.2.7 set_beat_width()	319
13.24.2.8 get_beats_per_measure()	320
13.24.2.9 set_beats_per_measure()	320
13.24.2.10get_beats_per_minute()	320
13.24.2.11set_beats_per_minute()	320
13.24.2.12transport_state()	320
13.24.2.13transport_not_starting()	321
13.24.2.14nit()	321
13.24.2.15deinit()	322
13.24.2.16session_event()	322
13.24.2.17activate()	322



13.24.2.18start()	322
13.24.2.19stop()	323
13.24.2.20position()	323
13.24.2.21output()	324
13.24.2.22set_ppqn()	324
13.24.2.23get_jack_tick()	325
13.24.2.24get_jack_pos()	325
13.24.2.25toggle_jack_mode()	325
13.24.2.26set_jack_mode()	325
13.24.2.27get_jack_mode()	325
13.24.2.28get_jack_stop_tick()	325
13.24.2.29set_jack_stop_tick()	326
13.24.2.30jack_frame_rate()	326
13.24.2.31get_follow_transport()	326
13.24.2.32set_follow_transport()	326
13.24.2.33toggle_follow_transport()	326
13.24.2.34toggle_song_start_mode()	326
13.24.2.35song_start_mode()	326
13.24.2.36set_start_from_perfedit()	327
13.24.2.37client()	327
13.24.2.38client_name()	327
13.24.2.39client_uuid()	327
13.24.2.40set_jack_running()	327
13.24.2.41tick_multiplier()	327
13.24.2.42client_open()	328
13.24.2.43get_jack_client_info()	328
13.24.2.44show_position()	328
13.24.2.45sync()	329
13.24.2.46set_position()	330
13.24.2.47info_message()	330

13.24.2.48error_message()	330
13.24.3 Friends And Related Function Documentation	332
13.24.3.1 jack_transport_callback	332
13.24.3.2 jack_shutdown_callback	332
13.24.3.3 jack_sync_callback	332
13.24.3.4 jack_timebase_callback	333
13.24.3.5 get_current_jack_position	334
13.24.3.6 jack_session_callback	334
13.24.4 Field Documentation	335
13.24.4.1 sm_status_pairs	335
13.24.4.2 m_jack_parent	335
13.24.4.3 m_jack_client	335
13.24.4.4 m_jack_client_name	335
13.24.4.5 m_jack_client_uuid	335
13.24.4.6 m_jack_frame_current	335
13.24.4.7 m_jack_frame_last	336
13.24.4.8 m_jack_pos	336
13.24.4.9 m_jack_transport_state	336
13.24.4.10m_jack_transport_state_last	336
13.24.4.11m_jack_tick	336
13.24.4.12m_jsession_ev	336
13.24.4.13m_jack_running	336
13.24.4.14m_jack_master	337
13.24.4.15m_jack_frame_rate	337
13.24.4.16m_toggle_jack	337
13.24.4.17m_jack_stop_tick	337
13.24.4.18m_follow_transport	337
13.24.4.19m_ppqn	337
13.24.4.20m_beats_per_measure	337
13.24.4.21m_beat_width	338

13.24.4.22m_beats_per_minute . . . . .	338
13.25seq64::jack_scratchpad Class Reference . . . . .	338
13.25.1 Detailed Description . . . . .	338
13.25.2 Field Documentation . . . . .	339
13.25.2.1 js_current_tick . . . . .	339
13.25.2.2 js_total_tick . . . . .	339
13.25.2.3 js_clock_tick . . . . .	339
13.25.2.4 js_jack_stopped . . . . .	339
13.25.2.5 js_dumping . . . . .	339
13.25.2.6 js_init_clock . . . . .	339
13.25.2.7 js_looping . . . . .	339
13.25.2.8 js_playback_mode . . . . .	340
13.25.2.9 js_ticks_converted . . . . .	340
13.25.2.10js_ticks_delta . . . . .	340
13.25.2.11js_ticks_converted_last . . . . .	340
13.25.2.12js_delta_tick_frac . . . . .	340
13.26seq64::jack_status_pair_t Struct Reference . . . . .	340
13.26.1 Field Documentation . . . . .	340
13.26.1.1 jf_bit . . . . .	341
13.26.1.2 jf_meaning . . . . .	341
13.27seq64::keybindentry Class Reference . . . . .	341
13.27.1 Member Enumeration Documentation . . . . .	342
13.27.1.1 type . . . . .	342
13.27.2 Constructor & Destructor Documentation . . . . .	342
13.27.2.1 keybindentry() . . . . .	342
13.27.3 Member Function Documentation . . . . .	343
13.27.3.1 set() . . . . .	343
13.27.3.2 on_key_press_event() . . . . .	343
13.27.4 Friends And Related Function Documentation . . . . .	343
13.27.4.1 options . . . . .	343

13.27.5 Field Documentation . . . . .	343
13.27.5.1 m_key . . . . .	343
13.27.5.2 m_type . . . . .	344
13.27.5.3 m_perf . . . . .	344
13.27.5.4 m_slot . . . . .	344
13.28seq64::keys_perform Class Reference . . . . .	344
13.28.1 Detailed Description . . . . .	351
13.28.2 Member Typedef Documentation . . . . .	351
13.28.2.1 SlotMap . . . . .	351
13.28.2.2 RevSlotMap . . . . .	351
13.28.3 Constructor & Destructor Documentation . . . . .	351
13.28.3.1 keys_perform() . . . . .	352
13.28.3.2 ~keys_perform() . . . . .	352
13.28.4 Member Function Documentation . . . . .	352
13.28.4.1 set_keys() . . . . .	352
13.28.4.2 get_keys() . . . . .	352
13.28.4.3 bpm_up() [1/2] . . . . .	352
13.28.4.4 bpm_up() [2/2] . . . . .	353
13.28.4.5 bpm_dn() [1/2] . . . . .	353
13.28.4.6 bpm_dn() [2/2] . . . . .	353
13.28.4.7 replace() [1/2] . . . . .	353
13.28.4.8 replace() [2/2] . . . . .	353
13.28.4.9 queue() [1/2] . . . . .	354
13.28.4.10queue() [2/2] . . . . .	354
13.28.4.11keep_queue() [1/2] . . . . .	354
13.28.4.12keep_queue() [2/2] . . . . .	354
13.28.4.13snapshot_1() [1/2] . . . . .	354
13.28.4.14snapshot_1() [2/2] . . . . .	354
13.28.4.15snapshot_2() [1/2] . . . . .	355
13.28.4.16snapshot_2() [2/2] . . . . .	355

13.28.4.17	screenset_up() [1/2]	355
13.28.4.18	screenset_up() [2/2]	355
13.28.4.19	screenset_dn() [1/2]	355
13.28.4.20	screenset_dn() [2/2]	356
13.28.4.21	set_playing_screenset() [1/2]	356
13.28.4.22	set_playing_screenset() [2/2]	356
13.28.4.23	group_on() [1/2]	356
13.28.4.24	group_on() [2/2]	356
13.28.4.25	group_off() [1/2]	357
13.28.4.26	group_off() [2/2]	357
13.28.4.27	group_learn() [1/2]	357
13.28.4.28	group_learn() [2/2]	357
13.28.4.29	start() [1/2]	357
13.28.4.30	start() [2/2]	357
13.28.4.31	pause() [1/2]	358
13.28.4.32	pause() [2/2]	358
13.28.4.33	pattern_edit() [1/2]	358
13.28.4.34	pattern_edit() [2/2]	358
13.28.4.35	event_edit() [1/2]	358
13.28.4.36	event_edit() [2/2]	359
13.28.4.37	stop() [1/2]	359
13.28.4.38	stop() [2/2]	359
13.28.4.39	song_mode() [1/2]	359
13.28.4.40	song_mode() [2/2]	359
13.28.4.41	menu_mode() [1/2]	359
13.28.4.42	menu_mode() [2/2]	360
13.28.4.43	follow_transport() [1/2]	360
13.28.4.44	follow_transport() [2/2]	360
13.28.4.45	fast_forward() [1/2]	360
13.28.4.46	fast_forward() [2/2]	360

13.28.4.47	rewind() [1/2]	360
13.28.4.48	rewind() [2/2]	360
13.28.4.49	pointer_position() [1/2]	361
13.28.4.50	pointer_position() [2/2]	361
13.28.4.51	toggle_mutes() [1/2]	361
13.28.4.52	toggle_mutes() [2/2]	361
13.28.4.53	toggle_jack() [1/2]	361
13.28.4.54	toggle_jack() [2/2]	361
13.28.4.55	tap_bpm() [1/2]	361
13.28.4.56	tap_bpm() [2/2]	362
13.28.4.57	show_ui_sequence_key() [1/2]	362
13.28.4.58	show_ui_sequence_key() [2/2]	362
13.28.4.59	show_ui_sequence_number() [1/2]	362
13.28.4.60	show_ui_sequence_number() [2/2]	362
13.28.4.61	get_key_events()	362
13.28.4.62	get_key_groups()	363
13.28.4.63	get_key_events_rev()	363
13.28.4.64	get_key_groups_rev()	363
13.28.4.65	lookup_keyevent_key()	363
13.28.4.66	lookup_keyevent_seq()	363
13.28.4.67	lookup_keygroup_key()	363
13.28.4.68	lookup_keygroup_group()	364
13.28.4.69	key_name()	364
13.28.4.70	set_all_key_events()	364
13.28.4.71	set_all_key_groups()	365
13.28.4.72	set_key_event()	365
13.28.4.73	set_key_group()	365
13.28.4.74	at_bpm_up()	365
13.28.4.75	at_bpm_dn()	366
13.28.4.76	at_replace()	366

13.28.4.77	at_queue()	366
13.28.4.78	at_keep_queue()	366
13.28.4.79	at_snapshot_1()	366
13.28.4.80	at_snapshot_2()	366
13.28.4.81	at_screenset_up()	366
13.28.4.82	at_screenset_dn()	366
13.28.4.83	at_set_playing_screenset()	367
13.28.4.84	at_group_on()	367
13.28.4.85	at_group_off()	367
13.28.4.86	at_group_learn()	367
13.28.4.87	at_start()	367
13.28.4.88	at_pause()	367
13.28.4.89	at_song_mode()	367
13.28.4.90	at_toggle_jack()	367
13.28.4.91	at_menu_mode()	368
13.28.4.92	at_follow_transport()	368
13.28.4.93	at_fast_forward()	368
13.28.4.94	at_rewind()	368
13.28.4.95	at_pointer_position()	368
13.28.4.96	at_toggle_mutes()	368
13.28.4.97	at_tap_bpm()	368
13.28.4.98	at_pattern_edit()	368
13.28.4.99	at_event_edit()	369
13.28.4.100	at_stop()	369
13.28.4.101	at_show_ui_sequence_key()	369
13.28.4.102	at_show_ui_sequence_number()	369
13.28.5	Friends And Related Function Documentation	369
13.28.5.1	options	369
13.28.5.2	perform	369
13.28.5.3	optionsfile	369

13.28.6 Field Documentation . . . . .	370
13.28.6.1 m_key_show_ui_sequence_key . . . . .	370
13.28.6.2 m_key_show_ui_sequence_number . . . . .	370
13.28.6.3 m_key_events . . . . .	370
13.28.6.4 m_key_groups . . . . .	370
13.28.6.5 m_key_events_rev . . . . .	370
13.28.6.6 m_key_groups_rev . . . . .	370
13.28.6.7 m_key_bpm_up . . . . .	370
13.28.6.8 m_key_bpm_dn . . . . .	371
13.28.6.9 m_key_replace . . . . .	371
13.28.6.10 m_key_queue . . . . .	371
13.28.6.11 m_key_keep_queue . . . . .	371
13.28.6.12 m_key_snapshot_1 . . . . .	371
13.28.6.13 m_key_snapshot_2 . . . . .	371
13.28.6.14 m_key_screenset_up . . . . .	371
13.28.6.15 m_key_screenset_dn . . . . .	371
13.28.6.16 m_key_set_playing_screenset . . . . .	372
13.28.6.17 m_key_group_on . . . . .	372
13.28.6.18 m_key_group_off . . . . .	372
13.28.6.19 m_key_group_learn . . . . .	372
13.28.6.20 m_key_start . . . . .	372
13.28.6.21 m_key_pause . . . . .	372
13.28.6.22 m_key_song_mode . . . . .	372
13.28.6.23 m_key_toggle_jack . . . . .	372
13.28.6.24 m_key_menu_mode . . . . .	373
13.28.6.25 m_key_follow_transport . . . . .	373
13.28.6.26 m_key_rewind . . . . .	373
13.28.6.27 m_key_fast_forward . . . . .	373
13.28.6.28 m_key_pointer_position . . . . .	373
13.28.6.29 m_key_toggle_mutes . . . . .	373



13.28.6.30m_key_tap_bpm . . . . .	373
13.28.6.31m_key_pattern_edit . . . . .	373
13.28.6.32m_key_event_edit . . . . .	374
13.28.6.33m_key_stop . . . . .	374
13.29seq64::keys_perform_gtk2 Class Reference . . . . .	374
13.29.1 Detailed Description . . . . .	376
13.29.2 Constructor & Destructor Documentation . . . . .	376
13.29.2.1 keys_perform_gtk2() . . . . .	376
13.29.2.2 ~keys_perform_gtk2() . . . . .	376
13.29.3 Member Function Documentation . . . . .	376
13.29.3.1 key_name() . . . . .	376
13.29.3.2 set_all_key_events() . . . . .	377
13.29.3.3 set_all_key_groups() . . . . .	377
13.30seq64::keys_perform_transfer Struct Reference . . . . .	377
13.30.1 Field Documentation . . . . .	378
13.30.1.1 kpt_bpm_up . . . . .	378
13.30.1.2 kpt_bpm_dn . . . . .	378
13.30.1.3 kpt_screenset_up . . . . .	378
13.30.1.4 kpt_screenset_dn . . . . .	378
13.30.1.5 kpt_set_playing_screenset . . . . .	378
13.30.1.6 kpt_group_on . . . . .	378
13.30.1.7 kpt_group_off . . . . .	378
13.30.1.8 kpt_group_learn . . . . .	379
13.30.1.9 kpt_replace . . . . .	379
13.30.1.10kpt_queue . . . . .	379
13.30.1.11kpt_keep_queue . . . . .	379
13.30.1.12kpt_snapshot_1 . . . . .	379
13.30.1.13kpt_snapshot_2 . . . . .	379
13.30.1.14kpt_start . . . . .	379
13.30.1.15kpt_stop . . . . .	379

13.30.1.16	kpt_show_ui_sequence_key . . . . .	380
13.30.1.17	kpt_show_ui_sequence_number . . . . .	380
13.30.1.18	kpt_pattern_edit . . . . .	380
13.30.1.19	kpt_event_edit . . . . .	380
13.30.1.20	kpt_tap_bpm . . . . .	380
13.30.1.21	kpt_pause . . . . .	380
13.30.1.22	kpt_song_mode . . . . .	380
13.30.1.23	kpt_toggle_jack . . . . .	380
13.30.1.24	kpt_menu_mode . . . . .	381
13.30.1.25	kpt_follow_transport . . . . .	381
13.30.1.26	kpt_fast_forward . . . . .	381
13.30.1.27	kpt_rewind . . . . .	381
13.30.1.28	kpt_pointer_position . . . . .	381
13.30.1.29	kpt_toggle_mutes . . . . .	381
13.31	seq64::keystroke Class Reference . . . . .	381
13.31.1	Detailed Description . . . . .	382
13.31.2	Constructor & Destructor Documentation . . . . .	382
13.31.2.1	keystroke() [1/3] . . . . .	383
13.31.2.2	keystroke() [2/3] . . . . .	383
13.31.2.3	keystroke() [3/3] . . . . .	383
13.31.3	Member Function Documentation . . . . .	383
13.31.3.1	operator=() . . . . .	383
13.31.3.2	is_press() . . . . .	384
13.31.3.3	is_letter() . . . . .	384
13.31.3.4	is() . . . . .	384
13.31.3.5	is_delete() . . . . .	384
13.31.3.6	key() . . . . .	385
13.31.3.7	shift_lock() . . . . .	385
13.31.3.8	modifier() . . . . .	385
13.31.3.9	mod_control() . . . . .	385

13.31.3.10	<code>mod_control_shift()</code>	385
13.31.3.11	<code>mod_super()</code>	385
13.31.4	Field Documentation	386
13.31.4.1	<code>m_is_press</code>	386
13.31.4.2	<code>m_key</code>	386
13.31.4.3	<code>m_modifier</code>	386
13.32	<code>seq64::lash</code> Class Reference	386
13.32.1	Detailed Description	387
13.32.2	Constructor & Destructor Documentation	387
13.32.2.1	<code>lash()</code>	387
13.32.3	Member Function Documentation	387
13.32.3.1	<code>set_alsa_client_id()</code>	387
13.32.3.2	<code>start()</code>	388
13.32.3.3	<code>process_events()</code>	388
13.32.3.4	<code>init()</code>	388
13.32.3.5	<code>handle_event()</code>	388
13.32.3.6	<code>handle_config()</code>	388
13.32.4	Field Documentation	389
13.32.4.1	<code>m_perform</code>	389
13.32.4.2	<code>m_client</code>	389
13.32.4.3	<code>m_lash_args</code>	389
13.32.4.4	<code>m_is_lash_supported</code>	389
13.33	<code>seq64::lfownd</code> Class Reference	390
13.33.1	Detailed Description	391
13.33.2	Constructor & Destructor Documentation	392
13.33.2.1	<code>lfownd()</code>	392
13.33.2.2	<code>~lfownd()</code>	392
13.33.3	Member Function Documentation	392
13.33.3.1	<code>toggle_visible()</code>	392
13.33.3.2	<code>scale_lfo_change()</code>	392

13.33.3.3 on_focus_out_event()	392
13.33.4 Field Documentation	393
13.33.4.1 m_seq	393
13.33.4.2 m_seqdata	393
13.33.4.3 m_hbox	393
13.33.4.4 m_scale_value	393
13.33.4.5 m_scale_range	393
13.33.4.6 m_scale_speed	393
13.33.4.7 m_scale_phase	393
13.33.4.8 m_scale_wave	394
13.33.4.9 m_wave_name	394
13.33.4.10m_value	394
13.33.4.11m_range	394
13.33.4.12m_speed	394
13.33.4.13m_phase	394
13.33.4.14m_wave	394
13.34seq64::maintime Class Reference	395
13.34.1 Detailed Description	397
13.34.2 Constructor & Destructor Documentation	397
13.34.2.1 maintime() [1/2]	397
13.34.2.2 maintime() [2/2]	397
13.34.2.3 ~maintime()	397
13.34.3 Member Function Documentation	397
13.34.3.1 operator=()	397
13.34.3.2 idle_progress()	397
13.34.3.3 on_realize()	398
13.34.3.4 on_expose_event()	398
13.34.4 Friends And Related Function Documentation	398
13.34.4.1 mainwnd	398
13.34.5 Field Documentation	398

13.34.5.1 m_beat_width . . . . .	398
13.34.5.2 m_bar_width . . . . .	399
13.34.5.3 m_pill_width . . . . .	399
13.34.5.4 m_box_width . . . . .	399
13.34.5.5 m_box_height . . . . .	399
13.34.5.6 m_flash_width . . . . .	399
13.34.5.7 m_flash_height . . . . .	399
13.34.5.8 m_flash_x . . . . .	399
13.34.5.9 m_box_less_pill . . . . .	400
13.34.5.10m_tick . . . . .	400
13.34.5.11m_ppqn . . . . .	400
13.35seq64::mainwid Class Reference . . . . .	400
13.35.1 Detailed Description . . . . .	404
13.35.2 Constructor & Destructor Documentation . . . . .	404
13.35.2.1 mainwid() . . . . .	404
13.35.2.2 ~mainwid() . . . . .	404
13.35.3 Member Function Documentation . . . . .	405
13.35.3.1 set_screenset() . . . . .	405
13.35.3.2 reset() . . . . .	405
13.35.3.3 update_sequences_on_window() . . . . .	405
13.35.3.4 draw_pixmap_on_window() . . . . .	405
13.35.3.5 fill_background_window() . . . . .	406
13.35.3.6 redraw() . . . . .	406
13.35.3.7 seq_set_and_edit() . . . . .	406
13.35.3.8 seq_set_and_eventedit() . . . . .	406
13.35.3.9 draw_marker_on_sequence() . . . . .	406
13.35.3.10update_markers() . . . . .	407
13.35.3.11valid_sequence() . . . . .	407
13.35.3.12draw_sequence_on_pixmap() . . . . .	407
13.35.3.13draw_sequences_on_pixmap() . . . . .	408

13.35.3.14	<code>draw_sequence_pixmap_on_window()</code>	408
13.35.3.15	<code>seq_from_xy()</code>	408
13.35.3.16	<code>timeout()</code>	409
13.35.3.17	<code>calculate_base_sizes()</code>	409
13.35.3.18	<code>select_fg_bg_colors()</code>	409
13.35.3.19	<code>on_realize()</code>	409
13.35.3.20	<code>on_expose_event()</code>	409
13.35.3.21	<code>on_button_press_event()</code>	410
13.35.3.22	<code>on_button_release_event()</code>	410
13.35.3.23	<code>on_motion_notify_event()</code>	411
13.35.3.24	<code>on_focus_in_event()</code>	411
13.35.3.25	<code>on_focus_out_event()</code>	411
13.35.4	Friends And Related Function Documentation	412
13.35.4.1	<code>mainwnd</code>	412
13.35.4.2	<code>update_mainwid_sequences</code>	412
13.35.5	Field Documentation	412
13.35.5.1	<code>m_armed_progress_color</code>	412
13.35.5.2	<code>m_moving_seq</code>	412
13.35.5.3	<code>m_button_down</code>	412
13.35.5.4	<code>m_moving</code>	412
13.35.5.5	<code>m_old_seq</code>	413
13.35.5.6	<code>m_screenset</code>	413
13.35.5.7	<code>m_last_tick_x</code>	413
13.35.5.8	<code>m_last_playing</code>	413
13.35.5.9	<code>m_mainwnd_rows</code>	413
13.35.5.10	<code>m_mainwnd_cols</code>	413
13.35.5.11	<code>m_seqarea_x</code>	413
13.35.5.12	<code>m_seqarea_y</code>	413
13.35.5.13	<code>m_seqarea_seq_x</code>	414
13.35.5.14	<code>m_seqarea_seq_y</code>	414

13.35.5.15m_mainwid_x . . . . .	414
13.35.5.16m_mainwid_y . . . . .	414
13.35.5.17m_mainwid_border . . . . .	414
13.35.5.18m_mainwid_spacing . . . . .	414
13.35.5.19m_text_size_x . . . . .	414
13.35.5.20m_text_size_y . . . . .	414
13.35.5.21m_max_sets . . . . .	415
13.35.5.22m_screenset_slots . . . . .	415
13.35.5.23m_screenset_offset . . . . .	415
13.35.5.24m_progress_height . . . . .	415
13.36seq64::mainwnd Class Reference . . . . .	415
13.36.1 Constructor & Destructor Documentation . . . . .	420
13.36.1.1 mainwnd() . . . . .	420
13.36.1.2 ~mainwnd() . . . . .	421
13.36.2 Member Function Documentation . . . . .	421
13.36.2.1 open_file() . . . . .	421
13.36.2.2 rc_error_dialog() . . . . .	421
13.36.2.3 ppqn() [1/2] . . . . .	422
13.36.2.4 ppqn() [2/2] . . . . .	422
13.36.2.5 handle_signal() . . . . .	422
13.36.2.6 adj_callback_ss() . . . . .	422
13.36.2.7 adj_callback_bpm() . . . . .	422
13.36.2.8 edit_callback_notepad() . . . . .	422
13.36.2.9 timer_callback() . . . . .	423
13.36.2.10set_play_image() . . . . .	423
13.36.2.11set_songlive_image() . . . . .	423
13.36.2.12start_playing() . . . . .	423
13.36.2.13pause_playing() . . . . .	424
13.36.2.14stop_playing() . . . . .	424
13.36.2.15toggle_playing() . . . . .	424

13.36.2.16	learn_toggle()	424
13.36.2.17	open_performance_edit()	424
13.36.2.18	open_performance_edit_2()	424
13.36.2.19	enregister_perfedits()	425
13.36.2.20	sequence_key()	425
13.36.2.21	apply_song_transpose()	425
13.36.2.22	update_window_title()	425
13.36.2.23	toLower()	425
13.36.2.24	file_new()	425
13.36.2.25	file_open()	425
13.36.2.26	file_save()	426
13.36.2.27	set_song_mute()	426
13.36.2.28	file_import_dialog()	426
13.36.2.29	options_dialog()	426
13.36.2.30	jack_dialog()	426
13.36.2.31	about_dialog()	426
13.36.2.32	build_info_dialog()	427
13.36.2.33	query_save_changes()	427
13.36.2.34	new_open_error_dialog()	427
13.36.2.35	file_save_as()	427
13.36.2.36	file_exit()	428
13.36.2.37	new_file()	428
13.36.2.38	save_file()	428
13.36.2.39	choose_file()	428
13.36.2.40	s_save()	428
13.36.2.41	install_signal_handlers()	428
13.36.2.42	signal_action()	429
13.36.2.43	edit_field_has_focus()	429
13.36.2.44	on_delete_event()	429
13.36.2.45	on_key_press_event()	429



13.36.2.46	on_key_release_event()	429
13.36.2.47	on_realize()	430
13.36.2.48	on_grouplearnchange()	430
13.36.3	Field Documentation	430
13.36.3.1	m_sigpipe	430
13.36.3.2	m_tooltips	430
13.36.3.3	m_menubar	430
13.36.3.4	m_menu_file	430
13.36.3.5	m_menu_edit	431
13.36.3.6	m_menu_view	431
13.36.3.7	m_menu_help	431
13.36.3.8	m_ppqn	431
13.36.3.9	m_main_wid	431
13.36.3.10	m_main_time	431
13.36.3.11	m_perf_edit	431
13.36.3.12	m_perf_edit_2	432
13.36.3.13	m_options	432
13.36.3.14	m_main_cursor	432
13.36.3.15	m_image_play	432
13.36.3.16	m_button_learn	432
13.36.3.17	m_button_stop	432
13.36.3.18	m_button_play	432
13.36.3.19	m_button_perfectedit	433
13.36.3.20	m_button_jack	433
13.36.3.21	m_adjust_bpm	433
13.36.3.22	m_spinbutton_bpm	433
13.36.3.23	m_adjust_ss	433
13.36.3.24	m_spinbutton_ss	433
13.36.3.25	m_adjust_load_offset	433
13.36.3.26	m_spinbutton_load_offset	434

13.36.3.27m_entry_notes . . . . .	434
13.36.3.28m_is_running . . . . .	434
13.36.3.29m_timeout_connect . . . . .	434
13.36.3.30m_menu_mode . . . . .	434
13.36.3.31m_call_seq_edit . . . . .	434
13.36.3.32m_call_seq_eventedit . . . . .	434
13.37seq64::mastermidibase Class Reference . . . . .	435
13.37.1 Constructor & Destructor Documentation . . . . .	439
13.37.1.1 mastermidibase() . . . . .	439
13.37.1.2 ~mastermidibase() . . . . .	439
13.37.2 Member Function Documentation . . . . .	439
13.37.2.1 init() . . . . .	439
13.37.2.2 get_num_out_buses() . . . . .	440
13.37.2.3 get_num_in_buses() . . . . .	440
13.37.2.4 filter_by_channel() [1/2] . . . . .	440
13.37.2.5 filter_by_channel() [2/2] . . . . .	440
13.37.2.6 get_beats_per_minute() . . . . .	440
13.37.2.7 get_bpm() . . . . .	440
13.37.2.8 get_ppqn() . . . . .	440
13.37.2.9 is_dumping() . . . . .	441
13.37.2.10get_sequence() . . . . .	441
13.37.2.11start() . . . . .	441
13.37.2.12stop() . . . . .	441
13.37.2.13port_start() . . . . .	441
13.37.2.14port_exit() . . . . .	442
13.37.2.15play() . . . . .	442
13.37.2.16continue_from() . . . . .	442
13.37.2.17nit_clock() . . . . .	443
13.37.2.18set_clock() [1/2] . . . . .	443
13.37.2.19sysex() . . . . .	443

13.37.2.20	print()	444
13.37.2.21	flush()	444
13.37.2.22	set_sequence_input()	444
13.37.2.23	dump_midi_input()	444
13.37.2.24	initialize_buses()	445
13.37.2.25	get_midi_out_bus_name()	445
13.37.2.26	get_midi_in_bus_name()	445
13.37.2.27	poll_for_midi()	446
13.37.2.28	s_more_input()	446
13.37.2.29	get_midi_event()	446
13.37.2.30	set_clock() [2/2]	447
13.37.2.31	set_input()	447
13.37.2.32	get_input()	447
13.37.2.33	s_input_system_port()	448
13.37.2.34	get_clock()	448
13.37.2.35	set_ppqn()	448
13.37.2.36	set_beats_per_minute()	450
13.37.2.37	port_settings()	450
13.37.2.38	clock()	450
13.37.2.39	input()	450
13.37.2.40	activate()	450
13.37.2.41	api_init()	451
13.37.2.42	api_start()	451
13.37.2.43	api_continue_from()	451
13.37.2.44	api_init_clock()	451
13.37.2.45	api_stop()	451
13.37.2.46	api_set_ppqn()	451
13.37.2.47	api_set_beats_per_minute()	452
13.37.2.48	api_flush()	452
13.37.2.49	api_clock()	452

13.37.2.50	<code>api_port_start()</code>	452
13.37.2.51	<code>api_is_more_input()</code>	452
13.37.2.52	<code>api_get_midi_event()</code>	452
13.37.2.53	<code>api_poll_for_midi()</code>	453
13.37.2.54	<code>save_clock()</code>	453
13.37.2.55	<code>save_input()</code>	453
13.37.3	Friends And Related Function Documentation	453
13.37.3.1	<code>perform</code>	453
13.37.3.2	<code>midi_alsa_info</code>	454
13.37.4	Field Documentation	454
13.37.4.1	<code>m_max_busses</code>	454
13.37.4.2	<code>m_bus_announce</code>	454
13.37.4.3	<code>m_inbus_array</code>	454
13.37.4.4	<code>m_outbus_array</code>	454
13.37.4.5	<code>m_master_clocks</code>	454
13.37.4.6	<code>m_master_inputs</code>	454
13.37.4.7	<code>m_queue</code>	455
13.37.4.8	<code>m_ppqn</code>	455
13.37.4.9	<code>m_beats_per_minute</code>	455
13.37.4.10	<code>m_dumping_input</code>	455
13.37.4.11	<code>m_vector_sequence</code>	455
13.37.4.12	<code>m_filter_by_channel</code>	455
13.37.4.13	<code>m_seq</code>	455
13.37.4.14	<code>m_mutex</code>	456
13.38	<code>seq64::mastermidibus</code> Class Reference	456
13.38.1	Detailed Description	459
13.38.2	Constructor & Destructor Documentation	459
13.38.2.1	<code>mastermidibus()</code> [1/2]	459
13.38.2.2	<code>~mastermidibus()</code> [1/2]	459
13.38.2.3	<code>mastermidibus()</code> [2/2]	459

13.38.2.4 ~mastermidibus() [2/2]	459
13.38.3 Member Function Documentation	460
13.38.3.1 api_is_more_input() [1/2]	460
13.38.3.2 api_get_midi_event() [1/2]	460
13.38.3.3 api_poll_for_midi() [1/2]	460
13.38.3.4 api_init() [1/2]	461
13.38.3.5 api_set_ppqn() [1/2]	462
13.38.3.6 api_set_beats_per_minute() [1/2]	462
13.38.3.7 api_flush() [1/2]	462
13.38.3.8 api_start()	462
13.38.3.9 api_stop()	463
13.38.3.10 api_continue_from()	463
13.38.3.11 api_port_start() [1/2]	463
13.38.3.12 activate()	463
13.38.3.13 api_is_more_input() [2/2]	463
13.38.3.14 api_get_midi_event() [2/2]	463
13.38.3.15 api_poll_for_midi() [2/2]	464
13.38.3.16 api_init() [2/2]	464
13.38.3.17 api_set_ppqn() [2/2]	464
13.38.3.18 api_set_beats_per_minute() [2/2]	464
13.38.3.19 api_flush() [2/2]	464
13.38.3.20 api_port_start() [2/2]	464
13.38.3.21 port_list()	464
13.38.4 Friends And Related Function Documentation	465
13.38.4.1 midi_alsa_info	465
13.38.4.2 midi_jack_info	465
13.38.5 Field Documentation	465
13.38.5.1 m_alsa_seq	465
13.38.5.2 m_num_poll_descriptors	465
13.38.5.3 m_poll_descriptors	465

13.38.5.4 m_midi_master . . . . .	466
13.38.5.5 m_use_jack_polling . . . . .	466
13.39seq64::midi_alsa Class Reference . . . . .	466
13.39.1 Constructor & Destructor Documentation . . . . .	469
13.39.1.1 midi_alsa() . . . . .	469
13.39.1.2 ~midi_alsa() . . . . .	469
13.39.2 Member Function Documentation . . . . .	469
13.39.2.1 get_client() . . . . .	470
13.39.2.2 get_port() . . . . .	470
13.39.2.3 api_init_out() . . . . .	470
13.39.2.4 api_init_in() . . . . .	470
13.39.2.5 api_init_out_sub() . . . . .	471
13.39.2.6 api_init_in_sub() . . . . .	471
13.39.2.7 api_deinit_in() . . . . .	471
13.39.2.8 api_get_midi_event() . . . . .	471
13.39.2.9 api_poll_for_midi() . . . . .	472
13.39.2.10api_play() . . . . .	472
13.39.2.11api_sysex() . . . . .	472
13.39.2.12api_flush() . . . . .	473
13.39.2.13api_continue_from() . . . . .	473
13.39.2.14api_start() . . . . .	473
13.39.2.15api_stop() . . . . .	473
13.39.2.16api_clock() . . . . .	473
13.39.2.17api_set_ppqn() . . . . .	474
13.39.2.18api_set_beats_per_minute() . . . . .	474
13.39.2.19set_virtual_name() . . . . .	474
13.39.3 Field Documentation . . . . .	475
13.39.3.1 m_seq . . . . .	475
13.39.3.2 m_dest_addr_client . . . . .	475
13.39.3.3 m_dest_addr_port . . . . .	475

13.39.3.4 m_local_addr_client . . . . .	475
13.39.3.5 m_local_addr_port . . . . .	475
13.39.3.6 m_input_port_name . . . . .	475
13.40seq64::midi_alsa_info Class Reference . . . . .	476
13.40.1 Constructor & Destructor Documentation . . . . .	477
13.40.1.1 midi_alsa_info() . . . . .	477
13.40.1.2 ~midi_alsa_info() . . . . .	478
13.40.2 Member Function Documentation . . . . .	478
13.40.2.1 seq() . . . . .	478
13.40.2.2 api_get_midi_event() . . . . .	478
13.40.2.3 api_poll_for_midi() . . . . .	479
13.40.2.4 api_set_ppqn() . . . . .	479
13.40.2.5 api_set_beats_per_minute() . . . . .	479
13.40.2.6 api_port_start() . . . . .	479
13.40.2.7 api_flush() . . . . .	480
13.40.2.8 get_all_port_info() . . . . .	480
13.40.3 Field Documentation . . . . .	481
13.40.3.1 sm_input_caps . . . . .	481
13.40.3.2 sm_output_caps . . . . .	481
13.40.3.3 m_alsa_seq . . . . .	481
13.40.3.4 m_num_poll_descriptors . . . . .	481
13.40.3.5 m_poll_descriptors . . . . .	481
13.41seq64::midi_api Class Reference . . . . .	482
13.41.1 Detailed Description . . . . .	484
13.41.2 Constructor & Destructor Documentation . . . . .	484
13.41.2.1 midi_api() . . . . .	484
13.41.2.2 ~midi_api() . . . . .	484
13.41.3 Member Function Documentation . . . . .	485
13.41.3.1 is_input_port() . . . . .	485
13.41.3.2 is_virtual_port() . . . . .	485

13.41.3.3 is_system_port()	485
13.41.3.4 api_connect()	485
13.41.3.5 api_init_out()	485
13.41.3.6 api_init_out_sub()	485
13.41.3.7 api_init_in()	486
13.41.3.8 api_init_in_sub()	486
13.41.3.9 api_deinit_in()	486
13.41.3.10 api_get_midi_event()	486
13.41.3.11 api_poll_for_midi()	486
13.41.3.12 api_play()	487
13.41.3.13 api_sysex()	487
13.41.3.14 api_continue_from()	487
13.41.3.15 api_start()	487
13.41.3.16 api_stop()	487
13.41.3.17 api_flush()	488
13.41.3.18 api_clock()	488
13.41.3.19 api_set_ppqn()	488
13.41.3.20 api_set_beats_per_minute()	488
13.41.3.21 api_get_bus_name()	488
13.41.3.22 api_get_port_name()	488
13.41.3.23 s_port_open()	489
13.41.3.24 master_info() [1/2]	489
13.41.3.25 master_info() [2/2]	489
13.41.3.26 parent_bus() [1/2]	489
13.41.3.27 parent_bus() [2/2]	489
13.41.3.28 master_midi_mode()	489
13.41.3.29 error()	489
13.41.3.30 user_callback()	490
13.41.3.31 cancel_callback()	490
13.41.3.32 set_port_open()	490



13.41.3.3	<a href="#">input_data()</a>	490
13.41.4	<a href="#">Field Documentation</a>	490
13.41.4.1	<a href="#">m_master_info</a>	490
13.41.4.2	<a href="#">m_parent_bus</a>	491
13.41.4.3	<a href="#">m_input_data</a>	491
13.41.4.4	<a href="#">m_connected</a>	491
13.41.4.5	<a href="#">m_error_string</a>	491
13.41.4.6	<a href="#">m_error_callback</a>	491
13.41.4.7	<a href="#">m_first_error_occurred</a>	491
13.41.4.8	<a href="#">m_error_callback_user_data</a>	491
13.42	<a href="#">seq64::midi_container Class Reference</a>	492
13.42.1	<a href="#">Detailed Description</a>	494
13.42.2	<a href="#">Constructor &amp; Destructor Documentation</a>	494
13.42.2.1	<a href="#">midi_container()</a>	494
13.42.2.2	<a href="#">~midi_container()</a>	494
13.42.3	<a href="#">Member Function Documentation</a>	494
13.42.3.1	<a href="#">fill()</a>	494
13.42.3.2	<a href="#">size()</a>	495
13.42.3.3	<a href="#">done()</a>	495
13.42.3.4	<a href="#">put()</a>	496
13.42.3.5	<a href="#">get()</a>	496
13.42.3.6	<a href="#">clear()</a>	496
13.42.3.7	<a href="#">position_reset()</a>	496
13.42.3.8	<a href="#">position()</a>	496
13.42.3.9	<a href="#">position_increment()</a>	496
13.42.3.10	<a href="#">add_variable()</a>	496
13.42.3.11	<a href="#">add_long()</a>	497
13.42.3.12	<a href="#">add_short()</a>	497
13.42.3.13	<a href="#">add_event()</a>	497
13.42.3.14	<a href="#">fill_seq_number()</a>	497

13.42.3.15	fill_seq_name()	498
13.42.3.16	fill_meta_track_end()	498
13.42.3.17	fill_proprietary()	498
13.42.3.18	fill_time_sig_and_tempo()	499
13.42.3.19	song_fill_seq_event()	500
13.42.3.20	song_fill_seq_trigger()	500
13.42.4	Friends And Related Function Documentation	500
13.42.4.1	midifile	500
13.42.5	Field Documentation	501
13.42.5.1	m_sequence	501
13.42.5.2	m_position_for_get	501
13.43	seq64::midi_control Class Reference	501
13.43.1	Detailed Description	502
13.43.2	Member Enumeration Documentation	502
13.43.2.1	action	502
13.43.3	Constructor & Destructor Documentation	503
13.43.3.1	midi_control()	503
13.43.4	Member Function Documentation	503
13.43.4.1	active()	503
13.43.4.2	inverse_active()	503
13.43.4.3	status()	503
13.43.4.4	data()	504
13.43.4.5	min_value()	504
13.43.4.6	max_value()	504
13.43.4.7	set() [1/2]	504
13.43.4.8	set() [2/2]	504
13.43.4.9	match()	505
13.43.4.10	n_range()	505
13.43.5	Field Documentation	505
13.43.5.1	m_active	505

13.43.5.2 m_inverse_active . . . . .	505
13.43.5.3 m_status . . . . .	505
13.43.5.4 m_data . . . . .	505
13.43.5.5 m_min_value . . . . .	506
13.43.5.6 m_max_value . . . . .	506
13.44seq64::midi_in_alsa Class Reference . . . . .	506
13.44.1 Constructor & Destructor Documentation . . . . .	508
13.44.1.1 midi_in_alsa() . . . . .	508
13.45seq64::midi_in_jack Class Reference . . . . .	508
13.45.1 Constructor & Destructor Documentation . . . . .	510
13.45.1.1 midi_in_jack() . . . . .	510
13.45.1.2 ~midi_in_jack() . . . . .	510
13.45.2 Member Function Documentation . . . . .	510
13.45.2.1 api_poll_for_midi() . . . . .	511
13.45.2.2 api_get_midi_event() . . . . .	511
13.45.2.3 open_client() . . . . .	511
13.45.3 Field Documentation . . . . .	511
13.45.3.1 m_client_name . . . . .	512
13.46seq64::midi_info Class Reference . . . . .	512
13.46.1 Constructor & Destructor Documentation . . . . .	515
13.46.1.1 midi_info() . . . . .	515
13.46.1.2 ~midi_info() . . . . .	515
13.46.2 Member Function Documentation . . . . .	515
13.46.2.1 midi_mode() [1/2] . . . . .	515
13.46.2.2 midi_mode() [2/2] . . . . .	515
13.46.2.3 midi_handle() [1/2] . . . . .	515
13.46.2.4 input_ports() . . . . .	515
13.46.2.5 output_ports() . . . . .	516
13.46.2.6 full_port_count() . . . . .	516
13.46.2.7 clear() . . . . .	516

13.46.2.8 <code>app_name()</code> . . . . .	516
13.46.2.9 <code>ppqn()</code> . . . . .	516
13.46.2.10 <code>bpm()</code> . . . . .	516
13.46.2.11 <code>api_set_ppqn()</code> . . . . .	516
13.46.2.12 <code>api_set_beats_per_minute()</code> . . . . .	517
13.46.2.13 <code>api_port_start()</code> . . . . .	517
13.46.2.14 <code>api_get_midi_event()</code> . . . . .	517
13.46.2.15 <code>api_poll_for_midi()</code> . . . . .	517
13.46.2.16 <code>api_flush()</code> . . . . .	517
13.46.2.17 <code>api_connect()</code> . . . . .	517
13.46.2.18 <code>get_port_count()</code> . . . . .	518
13.46.2.19 <code>get_bus_id()</code> . . . . .	518
13.46.2.20 <code>get_bus_name()</code> . . . . .	518
13.46.2.21 <code>get_port_id()</code> . . . . .	518
13.46.2.22 <code>get_port_name()</code> . . . . .	518
13.46.2.23 <code>get_input()</code> . . . . .	518
13.46.2.24 <code>get_virtual()</code> . . . . .	518
13.46.2.25 <code>get_system()</code> . . . . .	519
13.46.2.26 <code>queue_number()</code> . . . . .	519
13.46.2.27 <code>connect_name()</code> . . . . .	519
13.46.2.28 <code>port_list()</code> . . . . .	519
13.46.2.29 <code>global_queue()</code> [1/2] . . . . .	519
13.46.2.30 <code>error()</code> . . . . .	519
13.46.2.31 <code>get_all_port_info()</code> . . . . .	520
13.46.2.32 <code>add_bus()</code> . . . . .	520
13.46.2.33 <code>global_queue()</code> [2/2] . . . . .	520
13.46.2.34 <code>midi_handle()</code> [2/2] . . . . .	520
13.46.2.35 <code>bus_container()</code> . . . . .	520
13.46.2.36 <code>nc_midi_port_info()</code> . . . . .	521
13.46.2.37 <code>ref_midi_port_info()</code> . . . . .	521

13.46.3 Friends And Related Function Documentation . . . . .	521
13.46.3.1 rtmidi_info . . . . .	521
13.46.4 Field Documentation . . . . .	521
13.46.4.1 m_midi_mode_input . . . . .	521
13.46.4.2 m_input . . . . .	521
13.46.4.3 m_output . . . . .	521
13.46.4.4 m_bus_container . . . . .	522
13.46.4.5 m_global_queue . . . . .	522
13.46.4.6 m_midi_handle . . . . .	522
13.46.4.7 m_app_name . . . . .	522
13.46.4.8 m_ppqn . . . . .	522
13.46.4.9 m_bpm . . . . .	522
13.46.4.10m_error_string . . . . .	522
13.47seq64::midi_jack Class Reference . . . . .	523
13.47.1 Constructor & Destructor Documentation . . . . .	525
13.47.1.1 midi_jack() . . . . .	526
13.47.1.2 ~midi_jack() . . . . .	527
13.47.2 Member Function Documentation . . . . .	527
13.47.2.1 multi_client() . . . . .	527
13.47.2.2 client_handle() [1/2] . . . . .	527
13.47.2.3 jack_data() . . . . .	527
13.47.2.4 remote_port_name() [1/2] . . . . .	527
13.47.2.5 remote_port_name() [2/2] . . . . .	528
13.47.2.6 port_handle() [1/2] . . . . .	528
13.47.2.7 client_handle() [2/2] . . . . .	528
13.47.2.8 port_handle() [2/2] . . . . .	528
13.47.2.9 open_client_impl() . . . . .	528
13.47.2.10close_client() . . . . .	529
13.47.2.11close_port() . . . . .	529
13.47.2.12create_ringbuffer() . . . . .	529

13.47.2.13	connect_port()	529
13.47.2.14	register_port()	530
13.47.2.15	open_client()	530
13.47.2.16	api_connect()	530
13.47.2.17	api_init_out()	531
13.47.2.18	api_init_in()	531
13.47.2.19	api_init_out_sub()	532
13.47.2.20	api_init_in_sub()	532
13.47.2.21	api_deinit_in()	532
13.47.2.22	api_get_midi_event()	532
13.47.2.23	api_poll_for_midi()	533
13.47.2.24	api_play()	533
13.47.2.25	api_sysex()	533
13.47.2.26	api_flush()	533
13.47.2.27	api_continue_from()	534
13.47.2.28	api_start()	534
13.47.2.29	api_stop()	534
13.47.2.30	api_clock()	534
13.47.2.31	api_set_ppqn()	534
13.47.2.32	api_set_beats_per_minute()	535
13.47.2.33	api_get_port_name()	535
13.47.2.34	set_virtual_name()	535
13.47.3	Friends And Related Function Documentation	535
13.47.3.1	midi_jack_info	535
13.47.4	Field Documentation	536
13.47.4.1	m_multi_client	536
13.47.4.2	m_remote_port_name	536
13.47.4.3	m_jack_info	536
13.47.4.4	m_jack_data	536
13.48	seq64::midi_jack_data Struct Reference	536

13.48.1 Detailed Description . . . . .	537
13.48.2 Constructor & Destructor Documentation . . . . .	537
13.48.2.1 midi_jack_data() . . . . .	537
13.48.2.2 ~midi_jack_data() . . . . .	537
13.48.3 Member Function Documentation . . . . .	537
13.48.3.1 valid_buffer() . . . . .	537
13.48.4 Field Documentation . . . . .	538
13.48.4.1 m_jack_client . . . . .	538
13.48.4.2 m_jack_port . . . . .	538
13.48.4.3 m_jack_buffsize . . . . .	538
13.48.4.4 m_jack_buffmessage . . . . .	538
13.48.4.5 m_jack_lasttime . . . . .	538
13.48.4.6 m_jack_rtmidiin . . . . .	538
13.49seq64::midi_jack_info Class Reference . . . . .	539
13.49.1 Constructor & Destructor Documentation . . . . .	541
13.49.1.1 midi_jack_info() . . . . .	541
13.49.1.2 ~midi_jack_info() . . . . .	541
13.49.2 Member Function Documentation . . . . .	541
13.49.2.1 multi_client() . . . . .	541
13.49.2.2 client_handle() [1/2] . . . . .	542
13.49.2.3 api_get_midi_event() . . . . .	542
13.49.2.4 api_connect() . . . . .	542
13.49.2.5 api_poll_for_midi() . . . . .	542
13.49.2.6 api_set_ppqn() . . . . .	542
13.49.2.7 api_set_beats_per_minute() . . . . .	543
13.49.2.8 api_port_start() . . . . .	543
13.49.2.9 api_flush() . . . . .	544
13.49.2.10get_all_port_info() . . . . .	544
13.49.2.11client_handle() [2/2] . . . . .	545
13.49.2.12connect() . . . . .	545

13.49.2.13 disconnect()	545
13.49.2.14 extract_names()	545
13.49.2.15 add()	545
13.49.3 Friends And Related Function Documentation	545
13.49.3.1 midi_jack	545
13.49.3.2 jack_process_io	546
13.49.4 Field Documentation	546
13.49.4.1 m_multi_client	546
13.49.4.2 m_jack_ports	546
13.49.4.3 m_jack_client	546
13.50 seq64::midi_list Class Reference	546
13.50.1 Member Typedef Documentation	548
13.50.1.1 CharList	548
13.50.2 Constructor & Destructor Documentation	548
13.50.2.1 midi_list()	548
13.50.2.2 ~midi_list()	549
13.50.3 Member Function Documentation	549
13.50.3.1 size()	549
13.50.3.2 done()	549
13.50.3.3 put()	549
13.50.3.4 get()	549
13.50.3.5 clear()	550
13.50.4 Field Documentation	550
13.50.4.1 m_char_list	550
13.51 seq64::midi_measures Class Reference	550
13.51.1 Detailed Description	551
13.51.2 Constructor & Destructor Documentation	551
13.51.2.1 midi_measures() [1/2]	551
13.51.2.2 midi_measures() [2/2]	551
13.51.3 Member Function Documentation	551



13.51.3.1 measures() [1/2]	551
13.51.3.2 measures() [2/2]	551
13.51.3.3 beats() [1/2]	552
13.51.3.4 beats() [2/2]	552
13.51.3.5 divisions() [1/2]	552
13.51.3.6 divisions() [2/2]	552
13.51.4 Field Documentation	552
13.51.4.1 m_measures	552
13.51.4.2 m_beats	553
13.51.4.3 m_divisions	553
13.52seq64::midi_message Class Reference	553
13.52.1 Detailed Description	554
13.52.2 Member Typedef Documentation	554
13.52.2.1 container	554
13.52.3 Constructor & Destructor Documentation	554
13.52.3.1 midi_message()	554
13.52.4 Member Function Documentation	554
13.52.4.1 operator[]()	554
13.52.4.2 at() [1/2]	554
13.52.4.3 at() [2/2]	555
13.52.4.4 array()	555
13.52.4.5 count()	555
13.52.4.6 empty()	555
13.52.4.7 push()	555
13.52.4.8 timestamp() [1/2]	555
13.52.4.9 timestamp() [2/2]	555
13.52.5 Field Documentation	555
13.52.5.1 m_bytes	556
13.52.5.2 m_timestamp	556
13.53seq64::midi_out_alsa Class Reference	556

13.53.1 Constructor & Destructor Documentation . . . . .	558
13.53.1.1 midi_out_alsa() . . . . .	558
13.54seq64::midi_out_jack Class Reference . . . . .	558
13.54.1 Constructor & Destructor Documentation . . . . .	560
13.54.1.1 midi_out_jack() . . . . .	560
13.54.1.2 ~midi_out_jack() . . . . .	560
13.54.2 Member Function Documentation . . . . .	560
13.54.2.1 send_message() . . . . .	560
13.54.2.2 open_client() . . . . .	561
13.55seq64::midi_port_info Class Reference . . . . .	561
13.55.1 Constructor & Destructor Documentation . . . . .	562
13.55.1.1 midi_port_info() . . . . .	562
13.55.2 Member Function Documentation . . . . .	562
13.55.2.1 add() [1/2] . . . . .	562
13.55.2.2 add() [2/2] . . . . .	563
13.55.2.3 clear() . . . . .	563
13.55.2.4 get_port_count() . . . . .	563
13.55.2.5 get_bus_id() . . . . .	563
13.55.2.6 get_bus_name() . . . . .	563
13.55.2.7 get_port_id() . . . . .	563
13.55.2.8 get_port_name() . . . . .	563
13.55.2.9 get_input() . . . . .	564
13.55.2.10get_virtual() . . . . .	564
13.55.2.11get_system() . . . . .	564
13.55.2.12get_queue_number() . . . . .	564
13.55.2.13connect_name() . . . . .	564
13.55.3 Field Documentation . . . . .	564
13.55.3.1 m_port_count . . . . .	564
13.55.3.2 m_port_container . . . . .	565
13.56seq64::midi_queue Class Reference . . . . .	565

13.56.1 Detailed Description . . . . .	565
13.56.2 Constructor & Destructor Documentation . . . . .	566
13.56.2.1 midi_queue() . . . . .	566
13.56.2.2 ~midi_queue() . . . . .	566
13.56.3 Member Function Documentation . . . . .	566
13.56.3.1 empty() . . . . .	566
13.56.3.2 count() . . . . .	566
13.56.3.3 full() . . . . .	566
13.56.3.4 add() . . . . .	566
13.56.3.5 pop() . . . . .	567
13.56.3.6 pop_front() . . . . .	567
13.56.3.7 allocate() . . . . .	567
13.56.3.8 deallocate() . . . . .	567
13.56.3.9 front() . . . . .	567
13.56.4 Field Documentation . . . . .	568
13.56.4.1 m_front . . . . .	568
13.56.4.2 m_back . . . . .	568
13.56.4.3 m_size . . . . .	568
13.56.4.4 m_ring_size . . . . .	568
13.56.4.5 m_ring . . . . .	568
13.57seq64::midi_splitter Class Reference . . . . .	568
13.57.1 Detailed Description . . . . .	569
13.57.2 Constructor & Destructor Documentation . . . . .	570
13.57.2.1 midi_splitter() . . . . .	570
13.57.2.2 ~midi_splitter() . . . . .	571
13.57.3 Member Function Documentation . . . . .	571
13.57.3.1 log_main_sequence() . . . . .	571
13.57.3.2 initialize() . . . . .	571
13.57.3.3 increment() . . . . .	571
13.57.3.4 split() . . . . .	572

13.57.3.5 ppqn()	572
13.57.3.6 count()	572
13.57.3.7 split_channel()	573
13.57.4 Field Documentation	573
13.57.4.1 m_ppqn	573
13.57.4.2 m_use_default_ppqn	573
13.57.4.3 m_smf0_channels_count	574
13.57.4.4 m_smf0_channels	574
13.57.4.5 m_smf0_main_sequence	574
13.57.4.6 m_smf0_seq_number	574
13.58seq64::midi_timing Class Reference	574
13.58.1 Detailed Description	575
13.58.2 Constructor & Destructor Documentation	575
13.58.2.1 midi_timing() [1/2]	575
13.58.2.2 midi_timing() [2/2]	575
13.58.3 Member Function Documentation	575
13.58.3.1 beats_per_minute() [1/2]	576
13.58.3.2 beats_per_minute() [2/2]	576
13.58.3.3 beats_per_measure() [1/2]	576
13.58.3.4 beats_per_measure() [2/2]	576
13.58.3.5 beat_width() [1/2]	576
13.58.3.6 beat_width() [2/2]	576
13.58.3.7 ppqn() [1/2]	577
13.58.3.8 ppqn() [2/2]	577
13.58.4 Field Documentation	577
13.58.4.1 m_beats_per_minute	577
13.58.4.2 m_beats_per_measure	577
13.58.4.3 m_beat_width	577
13.58.4.4 m_ppqn	578
13.59seq64::midi_vector Class Reference	578

13.59.1 Member Typedef Documentation . . . . .	580
13.59.1.1 CharVector . . . . .	580
13.59.2 Constructor & Destructor Documentation . . . . .	580
13.59.2.1 midi_vector() . . . . .	580
13.59.2.2 ~midi_vector() . . . . .	581
13.59.3 Member Function Documentation . . . . .	581
13.59.3.1 size() . . . . .	581
13.59.3.2 done() . . . . .	581
13.59.3.3 put() . . . . .	581
13.59.3.4 get() . . . . .	582
13.59.3.5 clear() . . . . .	582
13.59.4 Field Documentation . . . . .	582
13.59.4.1 m_char_vector . . . . .	582
13.60seq64::midibase Class Reference . . . . .	582
13.60.1 Constructor & Destructor Documentation . . . . .	587
13.60.1.1 midibase() . . . . .	588
13.60.1.2 ~midibase() . . . . .	589
13.60.2 Member Function Documentation . . . . .	589
13.60.2.1 show_bus_values() . . . . .	589
13.60.2.2 display_name() [1/2] . . . . .	589
13.60.2.3 bus_name() [1/2] . . . . .	589
13.60.2.4 port_name() [1/2] . . . . .	589
13.60.2.5 connect_name() . . . . .	589
13.60.2.6 get_bus_index() . . . . .	589
13.60.2.7 get_bus_id() . . . . .	590
13.60.2.8 get_port_id() . . . . .	590
13.60.2.9 ppqn() . . . . .	590
13.60.2.10 bpm() . . . . .	590
13.60.2.11 match() . . . . .	590
13.60.2.12 s_virtual_port() [1/2] . . . . .	590

13.60.2.13s_virtual_port() [2/2] . . . . .	590
13.60.2.14s_input_port() [1/2] . . . . .	591
13.60.2.15s_output_port() . . . . .	591
13.60.2.16s_input_port() [2/2] . . . . .	591
13.60.2.17s_system_port() . . . . .	591
13.60.2.18set_system_port_flag() . . . . .	591
13.60.2.19set_clock() . . . . .	591
13.60.2.20get_clock() . . . . .	591
13.60.2.21set_clock_status() . . . . .	592
13.60.2.22get_input() . . . . .	592
13.60.2.23set_input_status() . . . . .	592
13.60.2.24queue_number() . . . . .	592
13.60.2.25set_bus_id() . . . . .	592
13.60.2.26set_name() . . . . .	592
13.60.2.27set_alt_name() . . . . .	593
13.60.2.28set_multi_name() . . . . .	593
13.60.2.29set_clock_mod() . . . . .	594
13.60.2.30get_clock_mod() . . . . .	594
13.60.2.31poll_for_midi() . . . . .	594
13.60.2.32get_midi_event() . . . . .	594
13.60.2.33nit_out() . . . . .	595
13.60.2.34nit_in() . . . . .	595
13.60.2.35deinit_in() . . . . .	595
13.60.2.36nit_out_sub() . . . . .	595
13.60.2.37nit_in_sub() . . . . .	596
13.60.2.38play() . . . . .	596
13.60.2.39sysex() . . . . .	596
13.60.2.40flush() . . . . .	596
13.60.2.41start() . . . . .	596
13.60.2.42stop() . . . . .	597

13.60.2.43	clock()	597
13.60.2.44	continue_from()	597
13.60.2.45	init_clock()	597
13.60.2.46	print()	598
13.60.2.47	set_input()	598
13.60.2.48	display_name() [2/2]	598
13.60.2.49	bus_name() [2/2]	598
13.60.2.50	port_name() [2/2]	598
13.60.2.51	set_port_id()	598
13.60.2.52	api_poll_for_midi()	599
13.60.2.53	api_get_midi_event()	599
13.60.2.54	api_init_in_sub()	599
13.60.2.55	api_init_out_sub()	599
13.60.2.56	api_deinit_in()	599
13.60.2.57	api_play()	600
13.60.2.58	api_sysex()	600
13.60.2.59	api_flush()	600
13.60.2.60	api_init_in()	600
13.60.2.61	api_init_out()	600
13.60.2.62	api_continue_from()	601
13.60.2.63	api_start()	601
13.60.2.64	api_stop()	601
13.60.2.65	api_clock()	601
13.60.3	Friends And Related Function Documentation	601
13.60.3.1	mastermidibus	601
13.60.4	Field Documentation	601
13.60.4.1	m_clock_mod	602
13.60.4.2	m_bus_index	602
13.60.4.3	m_bus_id	602
13.60.4.4	m_port_id	602

13.60.4.5 m_clock_type . . . . .	602
13.60.4.6 m_inputing . . . . .	602
13.60.4.7 m_ppqn . . . . .	602
13.60.4.8 m_bpm . . . . .	602
13.60.4.9 m_queue . . . . .	603
13.60.4.10m_display_name . . . . .	603
13.60.4.11m_bus_name . . . . .	603
13.60.4.12m_port_name . . . . .	603
13.60.4.13m_lasttick . . . . .	603
13.60.4.14m_is_virtual_port . . . . .	603
13.60.4.15m_is_input_port . . . . .	603
13.60.4.16m_is_system_port . . . . .	604
13.60.4.17m_mutex . . . . .	604
13.61seq64::midibus Class Reference . . . . .	604
13.61.1 Detailed Description . . . . .	607
13.61.2 Constructor & Destructor Documentation . . . . .	607
13.61.2.1 midibus() [1/3] . . . . .	607
13.61.2.2 midibus() [2/3] . . . . .	608
13.61.2.3 ~midibus() [1/2] . . . . .	608
13.61.2.4 midibus() [3/3] . . . . .	608
13.61.2.5 ~midibus() [2/2] . . . . .	608
13.61.3 Member Function Documentation . . . . .	609
13.61.3.1 get_client() . . . . .	609
13.61.3.2 get_port() . . . . .	609
13.61.3.3 api_init_out() [1/2] . . . . .	609
13.61.3.4 api_init_in() [1/2] . . . . .	609
13.61.3.5 api_init_out_sub() [1/2] . . . . .	610
13.61.3.6 api_init_in_sub() [1/2] . . . . .	610
13.61.3.7 api_deinit_in() [1/2] . . . . .	610
13.61.3.8 api_play() [1/2] . . . . .	610



13.61.3.9 <a href="#">api_sysex()</a> . . . . .	611
13.61.3.10 <a href="#">api_flush()</a> . . . . .	611
13.61.3.11 <a href="#">api_continue_from()</a> [1/2] . . . . .	611
13.61.3.12 <a href="#">api_start()</a> [1/2] . . . . .	611
13.61.3.13 <a href="#">api_stop()</a> [1/2] . . . . .	612
13.61.3.14 <a href="#">api_clock()</a> [1/2] . . . . .	612
13.61.3.15 <a href="#">set_virtual_name()</a> . . . . .	612
13.61.3.16 <a href="#">api_connect()</a> . . . . .	612
13.61.3.17 <a href="#">api_init_in()</a> [2/2] . . . . .	613
13.61.3.18 <a href="#">api_init_in_sub()</a> [2/2] . . . . .	613
13.61.3.19 <a href="#">api_init_out()</a> [2/2] . . . . .	613
13.61.3.20 <a href="#">api_init_out_sub()</a> [2/2] . . . . .	613
13.61.3.21 <a href="#">api_deinit_in()</a> [2/2] . . . . .	613
13.61.3.22 <a href="#">api_get_midi_event()</a> . . . . .	613
13.61.3.23 <a href="#">api_poll_for_midi()</a> . . . . .	614
13.61.3.24 <a href="#">api_continue_from()</a> [2/2] . . . . .	614
13.61.3.25 <a href="#">api_start()</a> [2/2] . . . . .	614
13.61.3.26 <a href="#">api_stop()</a> [2/2] . . . . .	614
13.61.3.27 <a href="#">api_clock()</a> [2/2] . . . . .	615
13.61.3.28 <a href="#">api_play()</a> [2/2] . . . . .	615
13.61.4 Friends And Related Function Documentation . . . . .	615
13.61.4.1 <a href="#">mastermidibus</a> . . . . .	615
13.61.5 Field Documentation . . . . .	615
13.61.5.1 <a href="#">m_seq</a> . . . . .	615
13.61.5.2 <a href="#">m_dest_addr_client</a> . . . . .	615
13.61.5.3 <a href="#">m_dest_addr_port</a> . . . . .	616
13.61.5.4 <a href="#">m_local_addr_client</a> . . . . .	616
13.61.5.5 <a href="#">m_local_addr_port</a> . . . . .	616
13.61.5.6 <a href="#">m_input_port_name</a> . . . . .	616
13.61.5.7 <a href="#">m_rt_midi</a> . . . . .	616

13.61.5.8 m_master_info . . . . .	616
13.62seq64::midifile Class Reference . . . . .	616
13.62.1 Detailed Description . . . . .	619
13.62.2 Constructor & Destructor Documentation . . . . .	619
13.62.2.1 midifile() . . . . .	619
13.62.2.2 ~midifile() . . . . .	620
13.62.3 Member Function Documentation . . . . .	620
13.62.3.1 parse() . . . . .	620
13.62.3.2 write() . . . . .	621
13.62.3.3 write_song() . . . . .	622
13.62.3.4 error_message() . . . . .	622
13.62.3.5 error_is_fatal() . . . . .	622
13.62.3.6 ppqn() . . . . .	622
13.62.3.7 parse_smf_0() . . . . .	622
13.62.3.8 parse_smf_1() . . . . .	623
13.62.3.9 parse_prop_header() . . . . .	623
13.62.3.10parse_proprietary_track() . . . . .	624
13.62.3.11pow2() . . . . .	625
13.62.3.12checklen() . . . . .	625
13.62.3.13add_trigger() . . . . .	626
13.62.3.14read_long() . . . . .	626
13.62.3.15read_short() . . . . .	626
13.62.3.16read_byte() . . . . .	627
13.62.3.17read_varinum() . . . . .	627
13.62.3.18write_long() . . . . .	627
13.62.3.19write_short() . . . . .	627
13.62.3.20read_byte_array() . . . . .	628
13.62.3.21write_byte() . . . . .	628
13.62.3.22write_varinum() . . . . .	628
13.62.3.23write_track_name() . . . . .	629

13.62.3.24	read_track_name()	629
13.62.3.25	write_seq_number()	629
13.62.3.26	read_seq_number()	630
13.62.3.27	write_track_end()	630
13.62.3.28	write_header()	630
13.62.3.29	write_prop_header()	631
13.62.3.30	write_proprietary_track()	631
13.62.3.31	varinum_size()	632
13.62.3.32	prop_item_size()	632
13.62.3.33	track_name_size()	633
13.62.3.34	errdump() [1/2]	633
13.62.3.35	errdump() [2/2]	633
13.62.3.36	write_track()	634
13.62.3.37	seq_number_size()	634
13.62.3.38	track_end_size()	634
13.62.3.39	s_sysex_special_id()	634
13.62.4	Field Documentation	635
13.62.4.1	m_mutex	635
13.62.4.2	m_file_size	635
13.62.4.3	m_error_message	635
13.62.4.4	m_error_is_fatal	635
13.62.4.5	m_disable_reported	635
13.62.4.6	m_pos	635
13.62.4.7	m_name	635
13.62.4.8	m_data	636
13.62.4.9	m_char_list	636
13.62.4.10	m_new_format	636
13.62.4.11	m_global_bgsequence	636
13.62.4.12	m_ppqn	636
13.62.4.13	m_use_default_ppqn	636

13.62.4.14m_smf0_splitter . . . . .	636
13.63seq64::mutex Class Reference . . . . .	637
13.63.1 Constructor & Destructor Documentation . . . . .	638
13.63.1.1 mutex() . . . . .	638
13.63.2 Member Function Documentation . . . . .	638
13.63.2.1 lock() . . . . .	638
13.63.2.2 unlock() . . . . .	638
13.63.3 Field Documentation . . . . .	638
13.63.3.1 sm_recursive_mutex . . . . .	638
13.63.3.2 m_mutex_lock . . . . .	638
13.64seq64::editable_event::name_value_t Struct Reference . . . . .	638
13.64.1 Field Documentation . . . . .	639
13.64.1.1 event_value . . . . .	639
13.64.1.2 event_name . . . . .	639
13.65seq64::options Class Reference . . . . .	639
13.65.1 Member Enumeration Documentation . . . . .	640
13.65.1.1 button . . . . .	641
13.65.2 Constructor & Destructor Documentation . . . . .	641
13.65.2.1 options() . . . . .	641
13.65.3 Member Function Documentation . . . . .	641
13.65.3.1 perf() . . . . .	641
13.65.3.2 clock_callback_off() . . . . .	642
13.65.3.3 clock_callback_on() . . . . .	642
13.65.3.4 clock_callback_mod() . . . . .	642
13.65.3.5 clock_mod_callback() . . . . .	642
13.65.3.6 input_callback() . . . . .	642
13.65.3.7 filter_callback() . . . . .	642
13.65.3.8 transport_callback() . . . . .	643
13.65.3.9 mouse_seq24_callback() . . . . .	643
13.65.3.10mouse_fruity_callback() . . . . .	643

13.65.3.11	<code>mouse_mod4_callback()</code>	643
13.65.3.12	<code>mouse_snap_split_callback()</code>	643
13.65.3.13	<code>mouse_click_edit_callback()</code>	643
13.65.3.14	<code>ash_support_callback()</code>	643
13.65.3.15	<code>add_midi_clock_page()</code>	644
13.65.3.16	<code>add_midi_input_page()</code>	644
13.65.3.17	<code>add_keyboard_page()</code>	644
13.65.3.18	<code>add_extended_keys_page()</code>	644
13.65.3.19	<code>add_mouse_page()</code>	644
13.65.3.20	<code>add_jack_sync_page()</code>	644
13.65.4	Field Documentation	644
13.65.4.1	<code>m_tooltips</code>	644
13.65.4.2	<code>m_mainperf</code>	645
13.65.4.3	<code>m_button_ok</code>	645
13.65.4.4	<code>m_button_jack_transport</code>	645
13.65.4.5	<code>m_button_jack_master</code>	645
13.65.4.6	<code>m_button_jack_master_cond</code>	645
13.65.4.7	<code>m_button_jack_connect</code>	645
13.65.4.8	<code>m_button_jack_disconnect</code>	645
13.65.4.9	<code>m_notebook</code>	645
13.66	<code>seq64::optionsfile</code> Class Reference	646
13.66.1	Detailed Description	647
13.66.2	Constructor & Destructor Documentation	647
13.66.2.1	<code>optionsfile()</code>	647
13.66.2.2	<code>~optionsfile()</code>	647
13.66.3	Member Function Documentation	647
13.66.3.1	<code>parse()</code>	647
13.66.3.2	<code>write()</code>	649
13.66.3.3	<code>error_message()</code>	649
13.67	<code>seq64::perfedit</code> Class Reference	650

13.67.1 Detailed Description . . . . .	654
13.67.2 Constructor & Destructor Documentation . . . . .	654
13.67.2.1 perfedit() . . . . .	654
13.67.2.2 ~perfedit() . . . . .	655
13.67.3 Member Function Documentation . . . . .	655
13.67.3.1 init_before_show() . . . . .	655
13.67.3.2 enqueue_draw() . . . . .	655
13.67.3.3 zoom_check() . . . . .	655
13.67.3.4 enregister_peer() . . . . .	656
13.67.3.5 set_zoom() . . . . .	656
13.67.3.6 get_toggle_jack() . . . . .	656
13.67.3.7 toggle_jack() . . . . .	656
13.67.3.8 rewind() . . . . .	657
13.67.3.9 fast_forward() . . . . .	657
13.67.3.10set_follow_transport() . . . . .	657
13.67.3.11toggle_follow_transport() . . . . .	657
13.67.3.12set_jack_mode() . . . . .	657
13.67.3.13set_transpose() . . . . .	657
13.67.3.14transpose_button_callback() . . . . .	658
13.67.3.15set_beats_per_bar() . . . . .	658
13.67.3.16set_beat_width() . . . . .	658
13.67.3.17set_snap() . . . . .	659
13.67.3.18set_guides() . . . . .	659
13.67.3.19grow() . . . . .	659
13.67.3.20set_looped() . . . . .	659
13.67.3.21expand() . . . . .	659
13.67.3.22collapse() . . . . .	659
13.67.3.23copy() . . . . .	660
13.67.3.24undo() . . . . .	660
13.67.3.25redo() . . . . .	660

13.67.3.26	<a href="#">popup_menu()</a>	660
13.67.3.27	<a href="#">draw_sequences()</a>	660
13.67.3.28	<a href="#">timeout()</a>	660
13.67.3.29	<a href="#">set_image()</a>	660
13.67.3.30	<a href="#">start_playing()</a>	661
13.67.3.31	<a href="#">pause_playing()</a>	661
13.67.3.32	<a href="#">stop_playing()</a>	661
13.67.3.33	<a href="#">toggle_playing()</a>	661
13.67.3.34	<a href="#">on_realize()</a>	661
13.67.3.35	<a href="#">on_key_press_event()</a>	661
13.67.3.36	<a href="#">on_key_release_event()</a>	662
13.67.3.37	<a href="#">on_delete_event()</a>	662
13.67.4	<a href="#">Friends And Related Function Documentation</a>	662
13.67.4.1	<a href="#">update_perfedit_sequences</a>	662
13.67.5	<a href="#">Field Documentation</a>	662
13.67.5.1	<a href="#">m_peer_perfedit</a>	662
13.67.5.2	<a href="#">m_table</a>	663
13.67.5.3	<a href="#">m_vadjust</a>	663
13.67.5.4	<a href="#">m_hadjust</a>	663
13.67.5.5	<a href="#">m_vscroll</a>	663
13.67.5.6	<a href="#">m_hscroll</a>	663
13.67.5.7	<a href="#">m_perfnames</a>	663
13.67.5.8	<a href="#">m_perfroll</a>	663
13.67.5.9	<a href="#">m_perftime</a>	663
13.67.5.10	<a href="#">m_menu_snap</a>	664
13.67.5.11	<a href="#">m_menu_xpose</a>	664
13.67.5.12	<a href="#">m_button_xpose</a>	664
13.67.5.13	<a href="#">m_entry_xpose</a>	664
13.67.5.14	<a href="#">m_image_play</a>	664
13.67.5.15	<a href="#">m_button_snap</a>	664

13.67.5.16m_entry_snap . . . . .	664
13.67.5.17m_button_stop . . . . .	664
13.67.5.18m_button_play . . . . .	665
13.67.5.19m_button_loop . . . . .	665
13.67.5.20m_button_expand . . . . .	665
13.67.5.21m_button_collapse . . . . .	665
13.67.5.22m_button_copy . . . . .	665
13.67.5.23m_button_grow . . . . .	665
13.67.5.24m_button_undo . . . . .	665
13.67.5.25m_button_redo . . . . .	665
13.67.5.26m_button_jack . . . . .	666
13.67.5.27m_button_follow . . . . .	666
13.67.5.28m_button_bpm . . . . .	666
13.67.5.29m_entry_bpm . . . . .	666
13.67.5.30m_button_bw . . . . .	666
13.67.5.31m_entry_bw . . . . .	666
13.67.5.32m_hbox . . . . .	666
13.67.5.33m_hlbox . . . . .	666
13.67.5.34m_tooltips . . . . .	667
13.67.5.35m_menu_bpm . . . . .	667
13.67.5.36m_menu_bw . . . . .	667
13.67.5.37m_snap . . . . .	667
13.67.5.38m_bpm . . . . .	667
13.67.5.39m_bw . . . . .	667
13.67.5.40m_ppqn . . . . .	667
13.67.5.41m_is_running . . . . .	667
13.67.5.42m_standard_bpm . . . . .	668
13.68seq64::perfnames Class Reference . . . . .	668
13.68.1 Detailed Description . . . . .	670
13.68.2 Constructor & Destructor Documentation . . . . .	670



13.68.2.1	<a href="#">perfnames()</a>	670
13.68.2.2	<a href="#">~perfnames()</a>	671
13.68.3	<a href="#">Member Function Documentation</a>	671
13.68.3.1	<a href="#">redraw_dirty_sequences()</a>	671
13.68.3.2	<a href="#">enqueue_draw()</a>	671
13.68.3.3	<a href="#">convert_y()</a>	671
13.68.3.4	<a href="#">draw_sequences()</a>	671
13.68.3.5	<a href="#">draw_sequence()</a>	672
13.68.3.6	<a href="#">change_vert()</a>	672
13.68.3.7	<a href="#">redraw()</a>	672
13.68.3.8	<a href="#">on_realize()</a>	672
13.68.3.9	<a href="#">on_expose_event()</a>	673
13.68.3.10	<a href="#">on_button_press_event()</a>	673
13.68.3.11	<a href="#">on_button_release_event()</a>	673
13.68.3.12	<a href="#">on_size_allocate()</a>	674
13.68.3.13	<a href="#">on_scroll_event()</a>	674
13.68.4	<a href="#">Friends And Related Function Documentation</a>	674
13.68.4.1	<a href="#">perfedit</a>	674
13.68.5	<a href="#">Field Documentation</a>	674
13.68.5.1	<a href="#">m_parent</a>	675
13.68.5.2	<a href="#">m_names_chars</a>	675
13.68.5.3	<a href="#">m_char_w</a>	675
13.68.5.4	<a href="#">m_setbox_w</a>	675
13.68.5.5	<a href="#">m_namebox_w</a>	675
13.68.5.6	<a href="#">m_names_x</a>	675
13.68.5.7	<a href="#">m_names_y</a>	675
13.68.5.8	<a href="#">m_xy_offset</a>	676
13.68.5.9	<a href="#">m_seqs_in_set</a>	676
13.68.5.10	<a href="#">m_sequence_max</a>	676
13.68.5.11	<a href="#">m_sequence_offset</a>	676

13.68.5.12m_sequence_active . . . . .	676
13.69seq64::perform Class Reference . . . . .	676
13.69.1 Detailed Description . . . . .	689
13.69.2 Member Enumeration Documentation . . . . .	690
13.69.2.1 mute_op_t . . . . .	690
13.69.2.2 ff_rw_button_t . . . . .	691
13.69.3 Constructor & Destructor Documentation . . . . .	691
13.69.3.1 perform() . . . . .	691
13.69.3.2 ~perform() . . . . .	692
13.69.4 Member Function Documentation . . . . .	692
13.69.4.1 is_modified() [1/2] . . . . .	692
13.69.4.2 modify() . . . . .	692
13.69.4.3 ppqn() . . . . .	692
13.69.4.4 sequence_count() . . . . .	692
13.69.4.5 sequence_max() . . . . .	692
13.69.4.6 is_control_status() . . . . .	692
13.69.4.7 set_edit_sequence() . . . . .	692
13.69.4.8 unset_edit_sequence() . . . . .	693
13.69.4.9 is_edit_sequence() . . . . .	693
13.69.4.10get_beats_per_bar() . . . . .	693
13.69.4.11set_beats_per_bar() . . . . .	693
13.69.4.12get_beat_width() . . . . .	694
13.69.4.13set_beat_width() . . . . .	694
13.69.4.14clocks_per_metronome() [1/2] . . . . .	694
13.69.4.15clocks_per_metronome() [2/2] . . . . .	694
13.69.4.16set_32nds_per_quarter() . . . . .	694
13.69.4.17get_32nds_per_quarter() . . . . .	694
13.69.4.18is_per_quarter_note() [1/2] . . . . .	695
13.69.4.19is_per_quarter_note() [2/2] . . . . .	695
13.69.4.20gui() [1/2] . . . . .	695

13.69.4.21	<a href="#">gui()</a> [2/2]	695
13.69.4.22	<a href="#">keys()</a> [1/2]	695
13.69.4.23	<a href="#">keys()</a> [2/2]	695
13.69.4.24	<a href="#">master_bus()</a>	695
13.69.4.25	<a href="#">filter_by_channel()</a>	696
13.69.4.26	<a href="#">s_running()</a>	696
13.69.4.27	<a href="#">s_pattern_playing()</a> [1/2]	696
13.69.4.28	<a href="#">toggle_song_start_mode()</a>	696
13.69.4.29	<a href="#">song_start_mode()</a> [1/2]	696
13.69.4.30	<a href="#">song_start_mode()</a> [2/2]	696
13.69.4.31	<a href="#">is_jack_running()</a>	696
13.69.4.32	<a href="#">s_jack_master()</a>	696
13.69.4.33	<a href="#">enregister()</a>	696
13.69.4.34	<a href="#">toggle_jack_mode()</a>	697
13.69.4.35	<a href="#">set_jack_mode()</a>	697
13.69.4.36	<a href="#">get_toggle_jack()</a>	697
13.69.4.37	<a href="#">set_jack_stop_tick()</a>	697
13.69.4.38	<a href="#">combine_bytes()</a>	698
13.69.4.39	<a href="#">FF_rewind()</a>	698
13.69.4.40	<a href="#">FF_RW_timeout()</a>	698
13.69.4.41	<a href="#">start_from_perfectedit()</a> [1/2]	698
13.69.4.42	<a href="#">start_from_perfectedit()</a> [2/2]	699
13.69.4.43	<a href="#">set_follow_transport()</a>	699
13.69.4.44	<a href="#">get_follow_transport()</a>	699
13.69.4.45	<a href="#">toggle_follow_transport()</a>	699
13.69.4.46	<a href="#">set_reposition()</a>	699
13.69.4.47	<a href="#">ff_rw_type()</a> [1/2]	699
13.69.4.48	<a href="#">ff_rw_type()</a> [2/2]	699
13.69.4.49	<a href="#">rewind()</a>	699
13.69.4.50	<a href="#">fast_forward()</a>	700

13.69.4.51	reposition()	700
13.69.4.52	clear_all()	700
13.69.4.53	launch()	701
13.69.4.54	new_sequence()	701
13.69.4.55	add_sequence()	701
13.69.4.56	delete_sequence()	702
13.69.4.57	is_sequence_in_edit()	702
13.69.4.58	clear_sequence_triggers()	703
13.69.4.59	print_triggers()	703
13.69.4.60	finish()	703
13.69.4.61	get_tick()	703
13.69.4.62	set_tick()	703
13.69.4.63	get_jack_tick()	703
13.69.4.64	set_jack_tick()	703
13.69.4.65	set_left_tick()	704
13.69.4.66	get_left_tick()	704
13.69.4.67	set_start_tick()	704
13.69.4.68	get_start_tick()	704
13.69.4.69	set_right_tick()	705
13.69.4.70	get_right_tick()	705
13.69.4.71	left_right_size()	705
13.69.4.72	is_active()	705
13.69.4.73	apply_song_transpose()	706
13.69.4.74	set_transpose()	706
13.69.4.75	get_transpose()	706
13.69.4.76	get_beats_per_minute()	706
13.69.4.77	set_sequence_control_status()	706
13.69.4.78	unset_sequence_control_status()	706
13.69.4.79	sequence_playing_toggle()	708
13.69.4.80	sequence_playing_change()	708

13.69.4.81sequence_playing_on()	709
13.69.4.82sequence_playing_off()	709
13.69.4.83mute_all_tracks()	709
13.69.4.84toggle_all_tracks()	709
13.69.4.85armed_saved()	709
13.69.4.86toggle_playing_tracks()	710
13.69.4.87mute_screenset()	710
13.69.4.88output_func()	710
13.69.4.89input_func()	711
13.69.4.90set_group_mute_state()	711
13.69.4.91get_group_mute_state()	711
13.69.4.92set_offset()	712
13.69.4.93get_offset()	712
13.69.4.94save_playing_state()	712
13.69.4.95restore_playing_state()	712
13.69.4.96key_name()	712
13.69.4.97get_key_events()	713
13.69.4.98get_key_groups()	713
13.69.4.99get_key_events_rev()	713
13.69.4.100get_key_groups_rev()	713
13.69.4.101show_ui_sequence_key() [1/2]	713
13.69.4.102show_ui_sequence_key() [2/2]	713
13.69.4.103show_ui_sequence_number() [1/2]	714
13.69.4.104show_ui_sequence_number() [2/2]	714
13.69.4.105okup_keyevent_key()	714
13.69.4.106okup_keyevent_seq()	714
13.69.4.107okup_keygroup_key()	715
13.69.4.108okup_keygroup_group()	715
13.69.4.109start_playing()	715
13.69.4.110pause_playing()	716

13.69.4.114	<a href="#">stop_playing()</a> . . . . .	717
13.69.4.115	<a href="#">start_key()</a> . . . . .	717
13.69.4.116	<a href="#">pause_key()</a> . . . . .	717
13.69.4.117	<a href="#">stop_key()</a> . . . . .	717
13.69.4.118	<a href="#">barn_toggle()</a> . . . . .	717
13.69.4.119	<a href="#">decrement_beats_per_minute()</a> . . . . .	718
13.69.4.120	<a href="#">increment_beats_per_minute()</a> . . . . .	718
13.69.4.121	<a href="#">page_decrement_beats_per_minute()</a> . . . . .	718
13.69.4.122	<a href="#">page_increment_beats_per_minute()</a> . . . . .	718
13.69.4.123	<a href="#">decrement_screenset()</a> . . . . .	719
13.69.4.124	<a href="#">increment_screenset()</a> . . . . .	719
13.69.4.125	<a href="#">highlight()</a> . . . . .	719
13.69.4.126	<a href="#">smf_0()</a> . . . . .	719
13.69.4.127	<a href="#">set_sequence()</a> [1/2] . . . . .	719
13.69.4.128	<a href="#">set_sequence()</a> [2/2] . . . . .	720
13.69.4.129	<a href="#">sequence_key()</a> . . . . .	720
13.69.4.130	<a href="#">sequence_label()</a> . . . . .	720
13.69.4.131	<a href="#">set_input_bus()</a> . . . . .	721
13.69.4.132	<a href="#">set_clock_bus()</a> . . . . .	721
13.69.4.133	<a href="#">mainwnd_key_event()</a> . . . . .	722
13.69.4.134	<a href="#">perfrroll_key_event()</a> . . . . .	722
13.69.4.135	<a href="#">playback_key_event()</a> . . . . .	722
13.69.4.136	<a href="#">move_triggers()</a> . . . . .	723
13.69.4.137	<a href="#">copy_triggers()</a> . . . . .	723
13.69.4.138	<a href="#">push_trigger_undo()</a> . . . . .	723
13.69.4.139	<a href="#">pop_trigger_undo()</a> . . . . .	724
13.69.4.140	<a href="#">pop_trigger_redo()</a> . . . . .	724
13.69.4.141	<a href="#">dirty_main()</a> . . . . .	724
13.69.4.142	<a href="#">dirty_edit()</a> . . . . .	724
13.69.4.143	<a href="#">dirty_perf()</a> . . . . .	725

13.69.4.141	<a href="#">is_dirty_names()</a>	725
13.69.4.142	<a href="#">is_exportable()</a>	725
13.69.4.143	<a href="#">set_screenset()</a>	726
13.69.4.144	<a href="#">get_screenset()</a>	726
13.69.4.145	<a href="#">set_playing_screenset()</a>	726
13.69.4.146	<a href="#">toggle_other_seqs()</a>	726
13.69.4.147	<a href="#">toggle_other_names()</a>	727
13.69.4.148	<a href="#">have_undo()</a>	727
13.69.4.149	<a href="#">set_have_undo()</a>	727
13.69.4.150	<a href="#">have_redo()</a>	727
13.69.4.151	<a href="#">set_have_redo()</a>	728
13.69.4.152	<a href="#">split_trigger()</a>	728
13.69.4.153	<a href="#">set_max_trigger()</a>	728
13.69.4.154	<a href="#">collapse()</a>	728
13.69.4.155	<a href="#">copy()</a>	728
13.69.4.156	<a href="#">expand()</a>	728
13.69.4.157	<a href="#">midi_control_toggle()</a>	729
13.69.4.158	<a href="#">midi_control_on()</a>	729
13.69.4.159	<a href="#">midi_control_off()</a>	729
13.69.4.160	<a href="#">midi_control_event()</a>	730
13.69.4.161	<a href="#">handle_midi_control()</a>	730
13.69.4.162	<a href="#">handle_midi_control_ex()</a>	731
13.69.4.163	<a href="#">get_screen_set_notepad()</a>	731
13.69.4.164	<a href="#">current_screen_set_notepad()</a>	731
13.69.4.165	<a href="#">set_screen_set_notepad()</a> [1/2]	732
13.69.4.166	<a href="#">set_screen_set_notepad()</a> [2/2]	732
13.69.4.167	<a href="#">set_playing_screenset()</a>	732
13.69.4.168	<a href="#">any_group_unmutes()</a>	732
13.69.4.169	<a href="#">mute_group_tracks()</a>	733
13.69.4.170	<a href="#">select_and_mute_group()</a>	733

13.69.4.174	<a href="#">set_song_mute()</a> . . . . .	733
13.69.4.175	<a href="#">set_mode_group_mute()</a> . . . . .	733
13.69.4.176	<a href="#">unset_mode_group_mute()</a> . . . . .	734
13.69.4.177	<a href="#">select_group_mute()</a> . . . . .	734
13.69.4.178	<a href="#">set_mode_group_learn()</a> . . . . .	734
13.69.4.179	<a href="#">unset_mode_group_learn()</a> . . . . .	734
13.69.4.180	<a href="#">group_learning()</a> . . . . .	734
13.69.4.181	<a href="#">set_and_copy_mute_group()</a> . . . . .	735
13.69.4.182	<a href="#">activate()</a> . . . . .	735
13.69.4.183	<a href="#">start()</a> . . . . .	735
13.69.4.184	<a href="#">stop()</a> . . . . .	735
13.69.4.185	<a href="#">start_jack()</a> . . . . .	736
13.69.4.186	<a href="#">stop_jack()</a> . . . . .	736
13.69.4.187	<a href="#">position_jack()</a> . . . . .	736
13.69.4.188	<a href="#">off_sequences()</a> . . . . .	737
13.69.4.189	<a href="#">all_notes_off()</a> . . . . .	737
13.69.4.190	<a href="#">set_active()</a> . . . . .	737
13.69.4.191	<a href="#">set_was_active()</a> . . . . .	737
13.69.4.192	<a href="#">set_sequences()</a> . . . . .	737
13.69.4.193	<a href="#">play()</a> . . . . .	738
13.69.4.194	<a href="#">set_orig_ticks()</a> . . . . .	738
13.69.4.195	<a href="#">set_beats_per_minute()</a> . . . . .	738
13.69.4.196	<a href="#">set_looping()</a> . . . . .	739
13.69.4.197	<a href="#">max_active_set()</a> . . . . .	739
13.69.4.198	<a href="#">launch_input_thread()</a> . . . . .	739
13.69.4.199	<a href="#">launch_output_thread()</a> . . . . .	739
13.69.4.200	<a href="#">init_jack_transport()</a> . . . . .	740
13.69.4.201	<a href="#">reinit_jack_transport()</a> . . . . .	740
13.69.4.202	<a href="#">seq_in_playing_screen()</a> . . . . .	740
13.69.4.203	<a href="#">modified()</a> [ 2 / 2 ] . . . . .	740



13.69.4.201	valid_midi_control_seq()	741
13.69.4.202	screenset_valid()	741
13.69.4.203	set_running()	741
13.69.4.204	pattern_playing() [2/2]	742
13.69.4.205	set_playback_mode()	742
13.69.4.206	route_group_offset()	742
13.69.4.207	seq_valid()	742
13.69.4.208	mseq_valid()	743
13.69.4.209	stall_sequence()	743
13.69.4.210	timer_start()	744
13.69.4.211	timer_stop()	744
13.69.4.212	dump_track()	744
13.69.4.213	set_key_event()	745
13.69.4.214	set_key_group()	745
13.69.4.215	create_master_bus()	745
13.69.4.216	add_clock()	746
13.69.4.217	set_clock()	746
13.69.4.218	add_input()	746
13.69.4.219	set_input()	746
13.69.4.220	get_input()	747
13.69.4.221	set_input_system_port()	747
13.69.5	Friends And Related Function Documentation	747
13.69.5.1	jack_assistant	747
13.69.5.2	keybindentry	747
13.69.5.3	mainwnd	747
13.69.5.4	midifile	747
13.69.5.5	optionsfile	748
13.69.5.6	options	748
13.69.5.7	perfedit	748
13.69.5.8	perfroll	748

13.69.5.9 input_thread_func . . . . .	748
13.69.5.10 output_thread_func . . . . .	748
13.69.5.11 jack_sync_callback . . . . .	749
13.69.5.12 jack_transport_callback . . . . .	749
13.69.5.13 jack_shutdown . . . . .	750
13.69.5.14 jack_timebase_callback . . . . .	750
13.69.5.15 get_current_jack_position . . . . .	751
13.69.6 Field Documentation . . . . .	751
13.69.6.1 sm_mc_dummy . . . . .	751
13.69.6.2 m_song_start_mode . . . . .	751
13.69.6.3 m_start_from_perfedit . . . . .	752
13.69.6.4 m_reposition . . . . .	752
13.69.6.5 m_excell_FF_RW . . . . .	752
13.69.6.6 m_FF_RW_button_type . . . . .	752
13.69.6.7 m_mute_group . . . . .	752
13.69.6.8 m_armed_saved . . . . .	752
13.69.6.9 m_armed_statuses . . . . .	752
13.69.6.10 m_tracks_mute_state . . . . .	753
13.69.6.11 m_mode_group . . . . .	753
13.69.6.12 m_mode_group_learn . . . . .	753
13.69.6.13 m_mute_group_selected . . . . .	753
13.69.6.14 m_playing_screen . . . . .	753
13.69.6.15 m_playscreen_offset . . . . .	753
13.69.6.16 m_seqs . . . . .	753
13.69.6.17 m_seqs_active . . . . .	754
13.69.6.18 m_was_active_main . . . . .	754
13.69.6.19 m_was_active_edit . . . . .	754
13.69.6.20 m_was_active_perf . . . . .	754
13.69.6.21 m_was_active_names . . . . .	754
13.69.6.22 m_sequence_state . . . . .	754

13.69.6.23m_transpose . . . . .	754
13.69.6.24m_out_thread . . . . .	755
13.69.6.25m_in_thread . . . . .	755
13.69.6.26m_out_thread_launched . . . . .	755
13.69.6.27m_in_thread_launched . . . . .	755
13.69.6.28m_running . . . . .	755
13.69.6.29m_is_pattern_playing . . . . .	755
13.69.6.30m_inputting . . . . .	755
13.69.6.31m_outputting . . . . .	756
13.69.6.32m_looping . . . . .	756
13.69.6.33m_playback_mode . . . . .	756
13.69.6.34m_ppqn . . . . .	756
13.69.6.35m_bpm . . . . .	756
13.69.6.36m_beats_per_bar . . . . .	756
13.69.6.37m_beat_width . . . . .	756
13.69.6.38m_clocks_per_metronome . . . . .	757
13.69.6.39m_32nds_per_quarter . . . . .	757
13.69.6.40m_us_per_quarter_note . . . . .	757
13.69.6.41m_master_bus . . . . .	757
13.69.6.42m_master_clocks . . . . .	757
13.69.6.43m_master_inputs . . . . .	757
13.69.6.44m_one_measure . . . . .	757
13.69.6.45m_left_tick . . . . .	758
13.69.6.46m_right_tick . . . . .	758
13.69.6.47m_starting_tick . . . . .	758
13.69.6.48m_tick . . . . .	758
13.69.6.49m_jack_tick . . . . .	758
13.69.6.50m_usemidiclock . . . . .	758
13.69.6.51m_midiclockrunning . . . . .	758
13.69.6.52m_midiclocktick . . . . .	759

13.69.6.53m_midiclockpos . . . . .	759
13.69.6.54m_dont_reset_ticks . . . . .	759
13.69.6.55m_screen_set_notepad . . . . .	759
13.69.6.56m_midi_cc_toggle . . . . .	759
13.69.6.57m_midi_cc_on . . . . .	759
13.69.6.58m_midi_cc_off . . . . .	759
13.69.6.59m_offset . . . . .	759
13.69.6.60m_control_status . . . . .	760
13.69.6.61m_screenset . . . . .	760
13.69.6.62m_seqs_in_set . . . . .	760
13.69.6.63m_max_sets . . . . .	760
13.69.6.64m_sequence_count . . . . .	760
13.69.6.65m_sequence_max . . . . .	760
13.69.6.66m_sequence_high . . . . .	760
13.69.6.67m_edit_sequence . . . . .	761
13.69.6.68m_is_modified . . . . .	761
13.69.6.69m_condition_var . . . . .	761
13.69.6.70m_jack_asst . . . . .	761
13.69.6.71m_have_undo . . . . .	761
13.69.6.72m_undo_vect . . . . .	761
13.69.6.73m_have_redo . . . . .	761
13.69.6.74m_redo_vect . . . . .	762
13.69.6.75m_notify . . . . .	762
13.69.6.76m_gui_support . . . . .	762
13.70seq64::performcallback Struct Reference . . . . .	762
13.70.1 Detailed Description . . . . .	763
13.70.2 Member Function Documentation . . . . .	764
13.70.2.1 on_grouplearnchange() . . . . .	764
13.71seq64::perfull Class Reference . . . . .	764
13.71.1 Constructor & Destructor Documentation . . . . .	769

13.71.1.1	<code>perfull()</code>	769
13.71.1.2	<code>~perfull()</code>	769
13.71.2	Member Function Documentation	769
13.71.2.1	<code>set_guides()</code>	769
13.71.2.2	<code>update_sizes()</code>	770
13.71.2.3	<code>init_before_show()</code>	770
13.71.2.4	<code>fill_background_pixmap()</code>	770
13.71.2.5	<code>increment_size()</code>	771
13.71.2.6	<code>draw_all()</code>	771
13.71.2.7	<code>follow_progress()</code>	771
13.71.2.8	<code>redraw_progress()</code>	771
13.71.2.9	<code>draw_progress()</code>	771
13.71.2.10	<code>redraw_dirty_sequences()</code>	771
13.71.2.11	<code>set_ppqn()</code>	772
13.71.2.12	<code>convert_xy()</code>	772
13.71.2.13	<code>convert_x()</code>	772
13.71.2.14	<code>snap_x()</code>	773
13.71.2.15	<code>draw_sequence_on()</code>	773
13.71.2.16	<code>draw_background_on()</code>	773
13.71.2.17	<code>draw_drawable_row()</code>	773
13.71.2.18	<code>change_horz()</code>	773
13.71.2.19	<code>change_vert()</code>	774
13.71.2.20	<code>split_trigger()</code>	774
13.71.2.21	<code>enqueue_draw()</code>	774
13.71.2.22	<code>set_zoom()</code>	774
13.71.2.23	<code>convert_drop_xy()</code>	774
13.71.2.24	<code>horizontal_adjust()</code>	774
13.71.2.25	<code>vertical_adjust()</code>	775
13.71.2.26	<code>horizontal_set()</code>	775
13.71.2.27	<code>vertical_set()</code>	775

13.71.2.28	<code>on_realize()</code>	776
13.71.2.29	<code>on_expose_event()</code>	776
13.71.2.30	<code>on_button_press_event()</code>	776
13.71.2.31	<code>on_button_release_event()</code>	777
13.71.2.32	<code>on_motion_notify_event()</code>	777
13.71.2.33	<code>on_scroll_event()</code>	777
13.71.2.34	<code>on_focus_in_event()</code>	777
13.71.2.35	<code>on_focus_out_event()</code>	777
13.71.2.36	<code>on_size_allocate()</code>	778
13.71.2.37	<code>on_key_press_event()</code>	778
13.71.2.38	<code>on_size_request()</code>	778
13.71.3	Friends And Related Function Documentation	778
13.71.3.1	<code>FruityPerfInput</code>	778
13.71.3.2	<code>Seq24PerfInput</code>	779
13.71.3.3	<code>perfedit</code>	779
13.71.4	Field Documentation	779
13.71.4.1	<code>m_parent</code>	779
13.71.4.2	<code>m_h_page_increment</code>	779
13.71.4.3	<code>m_v_page_increment</code>	779
13.71.4.4	<code>m_snap</code>	779
13.71.4.5	<code>m_ppqn</code>	779
13.71.4.6	<code>m_page_factor</code>	780
13.71.4.7	<code>m_divs_per_beat</code>	780
13.71.4.8	<code>m_ticks_per_bar</code>	780
13.71.4.9	<code>m_perf_scale_x</code>	780
13.71.4.10	<code>m_zoom</code>	780
13.71.4.11	<code>m_names_y</code>	780
13.71.4.12	<code>m_background_x</code>	780
13.71.4.13	<code>m_size_box_w</code>	781
13.71.4.14	<code>m_measure_length</code>	781

13.71.4.15m_beat_length . . . . .	781
13.71.4.16m_old_progress_ticks . . . . .	781
13.71.4.17m_have_button_press . . . . .	781
13.71.4.18m_transport_follow . . . . .	781
13.71.4.19m_trans_button_press . . . . .	781
13.71.4.20m_4bar_offset . . . . .	782
13.71.4.21m_sequence_offset . . . . .	782
13.71.4.22m_roll_length_ticks . . . . .	782
13.71.4.23m_drop_tick . . . . .	782
13.71.4.24m_drop_tick_trigger_offset . . . . .	782
13.71.4.25m_drop_sequence . . . . .	782
13.71.4.26m_sequence_max . . . . .	782
13.71.4.27m_sequence_active . . . . .	783
13.71.4.28m_fruity_interaction . . . . .	783
13.71.4.29m_seq24_interaction . . . . .	783
13.71.4.30m_interaction . . . . .	783
13.71.4.31m_moving . . . . .	783
13.71.4.32m_growing . . . . .	783
13.71.4.33m_grow_direction . . . . .	783
13.72seq64::perftime Class Reference . . . . .	784
13.72.1 Constructor & Destructor Documentation . . . . .	786
13.72.1.1 perftime() . . . . .	786
13.72.1.2 ~perftime() . . . . .	787
13.72.2 Member Function Documentation . . . . .	787
13.72.2.1 reset() . . . . .	787
13.72.2.2 set_scale() . . . . .	787
13.72.2.3 set_guides() . . . . .	787
13.72.2.4 increment_size() . . . . .	788
13.72.2.5 enqueue_draw() . . . . .	788
13.72.2.6 set_zoom() . . . . .	788

13.72.2.7 draw_background()	788
13.72.2.8 draw_progress_on_window()	788
13.72.2.9 change_horz()	788
13.72.2.10set_ppqn()	789
13.72.2.11tick_to_pixel()	789
13.72.2.12pixel_to_tick()	789
13.72.2.13tick_offset()	790
13.72.2.14update_sizes()	790
13.72.2.15dle_progress()	790
13.72.2.16update_pixmap()	790
13.72.2.17draw_pixmap_on_window()	790
13.72.2.18on_realize()	790
13.72.2.19on_expose_event()	790
13.72.2.20on_button_press_event()	791
13.72.2.21on_size_allocate()	791
13.72.2.22on_button_release_event()	791
13.72.2.23key_press_event()	792
13.72.3 Friends And Related Function Documentation	792
13.72.3.1 perfedit	792
13.72.4 Field Documentation	792
13.72.4.1 m_parent	792
13.72.4.2 m_4bar_offset	792
13.72.4.3 m_tick_offset	792
13.72.4.4 m_ppqn	793
13.72.4.5 m_snap	793
13.72.4.6 m_measure_length	793
13.72.4.7 m_left_marker_tick	793
13.72.4.8 m_right_marker_tick	793
13.72.4.9 m_perf_scale_x	793
13.72.4.10m_timearea_y	793



13.73seq64::midi_port_info::port_info_t Struct Reference . . . . .	793
13.73.1 Detailed Description . . . . .	794
13.73.2 Field Documentation . . . . .	794
13.73.2.1 m_client_number . . . . .	794
13.73.2.2 m_client_name . . . . .	794
13.73.2.3 m_port_number . . . . .	794
13.73.2.4 m_port_name . . . . .	795
13.73.2.5 m_queue_number . . . . .	795
13.73.2.6 m_is_input . . . . .	795
13.73.2.7 m_is_virtual . . . . .	795
13.73.2.8 m_is_system . . . . .	795
13.74seq64::rc_settings Class Reference . . . . .	795
13.74.1 Detailed Description . . . . .	800
13.74.2 Constructor & Destructor Documentation . . . . .	800
13.74.2.1 rc_settings() [1/2] . . . . .	800
13.74.2.2 rc_settings() [2/2] . . . . .	800
13.74.3 Member Function Documentation . . . . .	801
13.74.3.1 operator=() . . . . .	801
13.74.3.2 config_filespec() . . . . .	801
13.74.3.3 user_filespec() . . . . .	801
13.74.3.4 set_defaults() . . . . .	802
13.74.3.5 auto_option_save() [1/2] . . . . .	802
13.74.3.6 legacy_format() [1/2] . . . . .	802
13.74.3.7 lash_support() [1/2] . . . . .	802
13.74.3.8 allow_mod4_mode() [1/2] . . . . .	802
13.74.3.9 allow_snap_split() [1/2] . . . . .	802
13.74.3.10allow_click_edit() [1/2] . . . . .	802
13.74.3.11show_midi() [1/2] . . . . .	802
13.74.3.12priority() [1/2] . . . . .	803
13.74.3.13stats() [1/2] . . . . .	803

13.74.3.14	<a href="#">pass_sysex()</a> [1/2]	803
13.74.3.15	<a href="#">with_jack_transport()</a> [1/2]	803
13.74.3.16	<a href="#">with_jack_master()</a> [1/2]	803
13.74.3.17	<a href="#">with_jack_master_cond()</a> [1/2]	803
13.74.3.18	<a href="#">with_jack_midi()</a> [1/2]	803
13.74.3.19	<a href="#">with_jack()</a>	803
13.74.3.20	<a href="#">filter_by_channel()</a> [1/2]	804
13.74.3.21	<a href="#">manual_alsa_ports()</a> [1/2]	804
13.74.3.22	<a href="#">reveal_alsa_ports()</a> [1/2]	804
13.74.3.23	<a href="#">print_keys()</a> [1/2]	804
13.74.3.24	<a href="#">device_ignore()</a> [1/2]	804
13.74.3.25	<a href="#">device_ignore_num()</a> [1/2]	804
13.74.3.26	<a href="#">interaction_method()</a> [1/2]	804
13.74.3.27	<a href="#">filename()</a> [1/2]	804
13.74.3.28	<a href="#">jack_session_uuid()</a> [1/2]	805
13.74.3.29	<a href="#">last_used_dir()</a> [1/2]	805
13.74.3.30	<a href="#">config_directory()</a> [1/2]	805
13.74.3.31	<a href="#">config_filename()</a> [1/2]	805
13.74.3.32	<a href="#">user_filename()</a> [1/2]	805
13.74.3.33	<a href="#">config_filename_alt()</a> [1/2]	805
13.74.3.34	<a href="#">user_filename_alt()</a> [1/2]	805
13.74.3.35	<a href="#">application_name()</a>	805
13.74.3.36	<a href="#">app_client_name()</a>	806
13.74.3.37	<a href="#">auto_option_save()</a> [2/2]	806
13.74.3.38	<a href="#">legacy_format()</a> [2/2]	806
13.74.3.39	<a href="#">ash_support()</a> [2/2]	806
13.74.3.40	<a href="#">allow_mod4_mode()</a> [2/2]	806
13.74.3.41	<a href="#">allow_snap_split()</a> [2/2]	806
13.74.3.42	<a href="#">allow_click_edit()</a> [2/2]	806
13.74.3.43	<a href="#">show_midi()</a> [2/2]	807

13.74.3.44	<a href="#">priority()</a> [2/2]	807
13.74.3.45	<a href="#">stats()</a> [2/2]	807
13.74.3.46	<a href="#">pass_sysex()</a> [2/2]	807
13.74.3.47	<a href="#">with_jack_transport()</a> [2/2]	807
13.74.3.48	<a href="#">with_jack_master()</a> [2/2]	807
13.74.3.49	<a href="#">with_jack_master_cond()</a> [2/2]	807
13.74.3.50	<a href="#">with_jack_midi()</a> [2/2]	808
13.74.3.51	<a href="#">filter_by_channel()</a> [2/2]	808
13.74.3.52	<a href="#">manual_alsa_ports()</a> [2/2]	808
13.74.3.53	<a href="#">reveal_alsa_ports()</a> [2/2]	808
13.74.3.54	<a href="#">print_keys()</a> [2/2]	808
13.74.3.55	<a href="#">device_ignore()</a> [2/2]	808
13.74.3.56	<a href="#">device_ignore_num()</a> [2/2]	808
13.74.3.57	<a href="#">interaction_method()</a> [2/2]	809
13.74.3.58	<a href="#">filename()</a> [2/2]	809
13.74.3.59	<a href="#">jack_session_uuid()</a> [2/2]	809
13.74.3.60	<a href="#">last_used_dir()</a> [2/2]	809
13.74.3.61	<a href="#">config_directory()</a> [2/2]	810
13.74.3.62	<a href="#">set_config_files()</a>	810
13.74.3.63	<a href="#">config_filename()</a> [2/2]	810
13.74.3.64	<a href="#">user_filename()</a> [2/2]	810
13.74.3.65	<a href="#">config_filename_alt()</a> [2/2]	811
13.74.3.66	<a href="#">user_filename_alt()</a> [2/2]	811
13.74.3.67	<a href="#">home_config_directory()</a>	811
13.74.4	<a href="#">Friends And Related Function Documentation</a>	811
13.74.4.1	<a href="#">optionsfile</a>	812
13.74.4.2	<a href="#">options</a>	812
13.74.4.3	<a href="#">mainwnd</a>	812
13.74.4.4	<a href="#">rtmidi_info</a>	812
13.74.4.5	<a href="#">parse_command_line_options</a>	812

13.74.4.6 help_check . . . . .	813
13.74.5 Field Documentation . . . . .	813
13.74.5.1 m_auto_option_save . . . . .	813
13.74.5.2 m_legacy_format . . . . .	813
13.74.5.3 m_lash_support . . . . .	813
13.74.5.4 m_allow_mod4_mode . . . . .	813
13.74.5.5 m_allow_snap_split . . . . .	814
13.74.5.6 m_allow_click_edit . . . . .	814
13.74.5.7 m_show_midi . . . . .	814
13.74.5.8 m_priority . . . . .	814
13.74.5.9 m_stats . . . . .	814
13.74.5.10m_pass_sysex . . . . .	814
13.74.5.11m_with_jack_transport . . . . .	814
13.74.5.12m_with_jack_master . . . . .	814
13.74.5.13m_with_jack_master_cond . . . . .	815
13.74.5.14m_with_jack_midi . . . . .	815
13.74.5.15m_filter_by_channel . . . . .	815
13.74.5.16m_manual_alsa_ports . . . . .	815
13.74.5.17m_reveal_alsa_ports . . . . .	815
13.74.5.18m_print_keys . . . . .	815
13.74.5.19m_device_ignore . . . . .	815
13.74.5.20m_device_ignore_num . . . . .	815
13.74.5.21m_interaction_method . . . . .	816
13.74.5.22m_filename . . . . .	816
13.74.5.23m_jack_session_uuid . . . . .	816
13.74.5.24m_last_used_dir . . . . .	816
13.74.5.25m_config_directory . . . . .	816
13.74.5.26m_config_filename . . . . .	816
13.74.5.27m_user_filename . . . . .	816
13.74.5.28m_config_filename_alt . . . . .	816

13.74.5.29m_user_filename_alt . . . . .	817
13.74.5.30m_application_name . . . . .	817
13.74.5.31m_app_client_name . . . . .	817
13.75seq64::rect Class Reference . . . . .	817
13.75.1 Field Documentation . . . . .	817
13.75.1.1 x . . . . .	817
13.75.1.2 y . . . . .	818
13.75.1.3 height . . . . .	818
13.75.1.4 width . . . . .	818
13.76seq64::gui_drawingarea_gtk2::rect Struct Reference . . . . .	818
13.76.1 Field Documentation . . . . .	818
13.76.1.1 x . . . . .	818
13.76.1.2 y . . . . .	818
13.76.1.3 height . . . . .	819
13.76.1.4 width . . . . .	819
13.77seq64::rterror Class Reference . . . . .	819
13.77.1 Detailed Description . . . . .	820
13.77.2 Member Enumeration Documentation . . . . .	820
13.77.2.1 Type . . . . .	820
13.77.3 Constructor & Destructor Documentation . . . . .	820
13.77.3.1 rterror() . . . . .	820
13.77.3.2 ~rterror() . . . . .	820
13.77.4 Member Function Documentation . . . . .	821
13.77.4.1 print_message() . . . . .	821
13.77.4.2 getType() . . . . .	821
13.77.4.3 get_message() . . . . .	821
13.77.4.4 what() . . . . .	821
13.77.5 Field Documentation . . . . .	821
13.77.5.1 m_message . . . . .	821
13.77.5.2 m_type . . . . .	821

13.78seq64::rtmidi Class Reference . . . . .	822
13.78.1 Detailed Description . . . . .	824
13.78.2 Constructor & Destructor Documentation . . . . .	824
13.78.2.1 rtmidi() . . . . .	824
13.78.2.2 ~rtmidi() . . . . .	824
13.78.3 Member Function Documentation . . . . .	824
13.78.3.1 api_connect() . . . . .	824
13.78.3.2 api_play() . . . . .	825
13.78.3.3 api_continue_from() . . . . .	825
13.78.3.4 api_start() . . . . .	825
13.78.3.5 api_stop() . . . . .	825
13.78.3.6 api_clock() . . . . .	825
13.78.3.7 api_set_ppqn() . . . . .	825
13.78.3.8 api_set_beats_per_minute() . . . . .	826
13.78.3.9 api_init_out() . . . . .	826
13.78.3.10api_init_out_sub() . . . . .	826
13.78.3.11api_init_in() . . . . .	826
13.78.3.12api_init_in_sub() . . . . .	826
13.78.3.13api_deinit_in() . . . . .	826
13.78.3.14api_get_midi_event() . . . . .	827
13.78.3.15api_poll_for_midi() . . . . .	827
13.78.3.16api_sysex() . . . . .	827
13.78.3.17api_flush() . . . . .	827
13.78.3.18s_port_open() . . . . .	827
13.78.3.19get_bus_id() . . . . .	827
13.78.3.20get_bus_name() . . . . .	828
13.78.3.21get_port_id() . . . . .	828
13.78.3.22get_port_name() . . . . .	828
13.78.3.23get_port_count() . . . . .	828
13.78.3.24full_port_count() . . . . .	828

13.78.3.25	<a href="#">get_api()</a> [1/2]	829
13.78.3.26	<a href="#">get_api()</a> [2/2]	829
13.78.3.27	<a href="#">set_api()</a>	829
13.78.3.28	<a href="#">delete_api()</a>	829
13.78.4	<a href="#">Friends And Related Function Documentation</a>	829
13.78.4.1	<a href="#">midibus</a>	829
13.78.5	<a href="#">Field Documentation</a>	829
13.78.5.1	<a href="#">m_midi_info</a>	829
13.78.5.2	<a href="#">m_midi_api</a>	830
13.79	<a href="#">seq64::rtmidi_in Class Reference</a>	830
13.79.1	<a href="#">Detailed Description</a>	832
13.79.2	<a href="#">Constructor &amp; Destructor Documentation</a>	832
13.79.2.1	<a href="#">rtmidi_in()</a>	832
13.79.2.2	<a href="#">~rtmidi_in()</a>	833
13.79.3	<a href="#">Member Function Documentation</a>	833
13.79.3.1	<a href="#">user_callback()</a>	833
13.79.3.2	<a href="#">cancel_callback()</a>	833
13.79.3.3	<a href="#">openmidi_api()</a>	833
13.80	<a href="#">seq64::rtmidi_in_data Class Reference</a>	834
13.80.1	<a href="#">Detailed Description</a>	835
13.80.2	<a href="#">Constructor &amp; Destructor Documentation</a>	835
13.80.2.1	<a href="#">rtmidi_in_data()</a>	835
13.80.3	<a href="#">Member Function Documentation</a>	835
13.80.3.1	<a href="#">queue()</a> [1/2]	835
13.80.3.2	<a href="#">queue()</a> [2/2]	835
13.80.3.3	<a href="#">message()</a> [1/2]	835
13.80.3.4	<a href="#">message()</a> [2/2]	835
13.80.3.5	<a href="#">ignore_flags()</a> [1/2]	835
13.80.3.6	<a href="#">test_ignore_flags()</a>	836
13.80.3.7	<a href="#">ignore_flags()</a> [2/2]	836

13.80.3.8 do_input() [1/2]	836
13.80.3.9 do_input() [2/2]	836
13.80.3.10first_message() [1/2]	836
13.80.3.11first_message() [2/2]	836
13.80.3.12continue_sysex() [1/2]	836
13.80.3.13continue_sysex() [2/2]	837
13.80.3.14using_callback() [1/2]	837
13.80.3.15using_callback() [2/2]	837
13.80.3.16api_data() [1/3]	837
13.80.3.17api_data() [2/3]	837
13.80.3.18api_data() [3/3]	837
13.80.3.19user_data() [1/3]	837
13.80.3.20user_data() [2/3]	838
13.80.3.21user_data() [3/3]	838
13.80.3.22user_callback() [1/2]	838
13.80.3.23user_callback() [2/2]	838
13.80.4 Field Documentation	838
13.80.4.1 m_queue	838
13.80.4.2 m_message	838
13.80.4.3 m_ignore_flags	838
13.80.4.4 m_do_input	839
13.80.4.5 m_first_message	839
13.80.4.6 m_api_data	839
13.80.4.7 m_using_callback	839
13.80.4.8 m_user_callback	839
13.80.4.9 m_user_data	839
13.80.4.10m_continue_sysex	839
13.81 seq64::rtmidi_info Class Reference	839
13.81.1 Detailed Description	841
13.81.2 Constructor & Destructor Documentation	842



13.81.2.1	rtmidi_info()	842
13.81.2.2	~rtmidi_info()	842
13.81.3	Member Function Documentation	842
13.81.3.1	get_version()	842
13.81.3.2	get_compiled_api()	842
13.81.3.3	midi_mode() [1/2]	842
13.81.3.4	midi_mode() [2/2]	843
13.81.3.5	clear()	843
13.81.3.6	add_input()	843
13.81.3.7	add_output()	843
13.81.3.8	add_bus()	843
13.81.3.9	get_bus_id()	843
13.81.3.10	get_bus_name()	844
13.81.3.11	get_port_count()	844
13.81.3.12	full_port_count()	844
13.81.3.13	get_port_id()	844
13.81.3.14	get_port_name()	844
13.81.3.15	get_input()	845
13.81.3.16	get_virtual()	845
13.81.3.17	get_system()	845
13.81.3.18	get_all_port_info()	845
13.81.3.19	queue_number()	845
13.81.3.20	app_name()	845
13.81.3.21	global_queue()	845
13.81.3.22	ppqn()	846
13.81.3.23	api_set_ppqn()	846
13.81.3.24	bpm()	846
13.81.3.25	api_set_beats_per_minute()	846
13.81.3.26	api_port_start()	846
13.81.3.27	api_get_midi_event()	846

13.81.3.28	<a href="#">api_flush()</a>	846
13.81.3.29	<a href="#">api_poll_for_midi()</a>	847
13.81.3.30	<a href="#">port_list()</a>	847
13.81.3.31	<a href="#">selected_api()</a> [1/2]	847
13.81.3.32	<a href="#">get_api_info()</a> [1/2]	847
13.81.3.33	<a href="#">get_api_info()</a> [2/2]	847
13.81.3.34	<a href="#">api_connect()</a>	847
13.81.3.35	<a href="#">selected_api()</a> [2/2]	847
13.81.3.36	<a href="#">set_api_info()</a>	848
13.81.3.37	<a href="#">delete_api()</a>	848
13.81.3.38	<a href="#">openmidi_api()</a>	848
13.81.4	<a href="#">Friends And Related Function Documentation</a>	848
13.81.4.1	<a href="#">mastermidibus</a>	848
13.81.4.2	<a href="#">midibus</a>	849
13.81.4.3	<a href="#">rtmidi_in</a>	849
13.81.4.4	<a href="#">rtmidi_out</a>	849
13.81.5	<a href="#">Field Documentation</a>	849
13.81.5.1	<a href="#">m_info_api</a>	849
13.81.5.2	<a href="#">sm_selected_api</a>	849
13.82	<a href="#">seq64::rtmidi_out Class Reference</a>	850
13.82.1	<a href="#">Detailed Description</a>	851
13.82.2	<a href="#">Constructor &amp; Destructor Documentation</a>	851
13.82.2.1	<a href="#">rtmidi_out()</a>	851
13.82.2.2	<a href="#">~rtmidi_out()</a>	851
13.82.3	<a href="#">Member Function Documentation</a>	852
13.82.3.1	<a href="#">openmidi_api()</a>	852
13.83	<a href="#">seq64::Seq24PerfInput Class Reference</a>	852
13.83.1	<a href="#">Constructor &amp; Destructor Documentation</a>	853
13.83.1.1	<a href="#">Seq24PerfInput()</a>	853
13.83.2	<a href="#">Member Function Documentation</a>	853

13.83.2.1 on_button_press_event()	854
13.83.2.2 on_button_release_event()	854
13.83.2.3 on_motion_notify_event()	854
13.83.2.4 activate_adding()	855
13.83.2.5 handle_motion_key()	855
13.83.3 Friends And Related Function Documentation	855
13.83.3.1 perfroll	855
13.83.4 Field Documentation	856
13.83.4.1 m_effective_tick	856
13.84seq64::Seq24SeqEventInput Struct Reference	856
13.84.1 Constructor & Destructor Documentation	856
13.84.1.1 Seq24SeqEventInput()	856
13.84.2 Member Function Documentation	856
13.84.2.1 set_adding()	856
13.84.2.2 on_button_press_event()	857
13.84.2.3 on_button_release_event()	857
13.84.2.4 on_motion_notify_event()	857
13.84.3 Field Documentation	858
13.84.3.1 m_adding	858
13.85seq64::seqdata Class Reference	858
13.85.1 Constructor & Destructor Documentation	861
13.85.1.1 seqdata()	861
13.85.1.2 ~seqdata()	862
13.85.2 Member Function Documentation	862
13.85.2.1 reset()	862
13.85.2.2 redraw()	862
13.85.2.3 set_zoom()	862
13.85.2.4 set_data_type()	863
13.85.2.5 idle_redraw()	863
13.85.2.6 update_sizes()	863

13.85.2.7 update_pixmap()	863
13.85.2.8 draw_line_on_window()	863
13.85.2.9 xy_to_rect()	864
13.85.2.10 draw_events_on()	865
13.85.2.11 change_horz()	865
13.85.2.12 convert_x()	866
13.85.2.13 render_number()	866
13.85.2.14 draw_events_on_pixmap()	866
13.85.2.15 draw_pixmap_on_window()	866
13.85.2.16 on_realize()	866
13.85.2.17 on_expose_event()	866
13.85.2.18 on_button_press_event()	867
13.85.2.19 on_button_release_event()	867
13.85.2.20 on_motion_notify_event()	868
13.85.2.21 on_leave_notify_event()	868
13.85.2.22 on_scroll_event()	868
13.85.2.23 on_size_allocate()	869
13.85.3 Friends And Related Function Documentation	869
13.85.3.1 lfownd	869
13.85.3.2 sequevent	869
13.85.3.3 seqroll	869
13.85.4 Field Documentation	869
13.85.4.1 m_seq	869
13.85.4.2 m_zoom	869
13.85.4.3 m_scroll_offset_ticks	870
13.85.4.4 m_scroll_offset_x	870
13.85.4.5 m_number_w	870
13.85.4.6 m_number_h	870
13.85.4.7 m_number_offset_y	870
13.85.4.8 m_status	870

13.85.4.9 m_cc . . . . .	870
13.85.4.10m_numbers . . . . .	870
13.85.4.11m_old . . . . .	871
13.85.4.12m_drag_handle . . . . .	871
13.85.4.13m_dragging . . . . .	871
13.86seq64::seqedit Class Reference . . . . .	871
13.86.1 Detailed Description . . . . .	878
13.86.2 Constructor & Destructor Documentation . . . . .	879
13.86.2.1 seqedit() . . . . .	879
13.86.2.2 ~seqedit() . . . . .	879
13.86.3 Member Function Documentation . . . . .	879
13.86.3.1 set_zoom() . . . . .	879
13.86.3.2 set_snap() . . . . .	880
13.86.3.3 set_note_length() . . . . .	880
13.86.3.4 set_beats_per_bar() . . . . .	880
13.86.3.5 set_beat_width() . . . . .	881
13.86.3.6 set_transpose_image() . . . . .	881
13.86.3.7 set_rec_vol() . . . . .	881
13.86.3.8 horizontal_adjust() . . . . .	882
13.86.3.9 vertical_adjust() . . . . .	882
13.86.3.10horizontal_set() . . . . .	882
13.86.3.11vertical_set() . . . . .	882
13.86.3.12set_measures() . . . . .	883
13.86.3.13apply_length() . . . . .	883
13.86.3.14get_measures() . . . . .	883
13.86.3.15set_midi_channel() . . . . .	883
13.86.3.16set_midi_bus() . . . . .	884
13.86.3.17set_scale() . . . . .	884
13.86.3.18set_chord() . . . . .	884
13.86.3.19set_key() . . . . .	884

13.86.3.20	set_background_sequence()	885
13.86.3.21	transpose_change_callback()	885
13.86.3.22	name_change_callback()	885
13.86.3.23	play_change_callback()	885
13.86.3.24	record_change_callback()	885
13.86.3.25	q_rec_change_callback()	885
13.86.3.26	thru_change_callback()	886
13.86.3.27	undo_callback()	886
13.86.3.28	redo_callback()	886
13.86.3.29	set_data_type()	886
13.86.3.30	update_all_windows()	886
13.86.3.31	fill_top_bar()	887
13.86.3.32	create_menus()	887
13.86.3.33	popup_menu()	887
13.86.3.34	popup_event_menu()	887
13.86.3.35	popup_midibus_menu()	888
13.86.3.36	popup_sequence_menu()	888
13.86.3.37	popup_tool_menu()	888
13.86.3.38	popup_midich_menu()	888
13.86.3.39	create_menu_image()	888
13.86.3.40	timeout()	888
13.86.3.41	do_action()	889
13.86.3.42	mouse_action()	889
13.86.3.43	start_playing()	889
13.86.3.44	stop_playing()	889
13.86.3.45	change_focus()	889
13.86.3.46	handle_close()	889
13.86.3.47	on_realize()	890
13.86.3.48	on_set_focus()	890
13.86.3.49	on_focus_in_event()	890

13.86.3.50	<code>on_focus_out_event()</code>	890
13.86.3.51	<code>on_delete_event()</code>	890
13.86.3.52	<code>on_scroll_event()</code>	891
13.86.3.53	<code>on_key_press_event()</code>	891
13.86.4	Field Documentation	892
13.86.4.1	<code>seqmenu</code>	892
13.86.4.2	<code>m_initial_snap</code>	892
13.86.4.3	<code>m_initial_note_length</code>	892
13.86.4.4	<code>m_initial_chord</code>	892
13.86.4.5	<code>m_initial_zoom</code>	893
13.86.4.6	<code>m_zoom</code>	893
13.86.4.7	<code>m_snap</code>	893
13.86.4.8	<code>m_note_length</code>	893
13.86.4.9	<code>m_scale</code>	893
13.86.4.10	<code>m_chord</code>	893
13.86.4.11	<code>m_key</code>	893
13.86.4.12	<code>m_bgsequence</code>	893
13.86.4.13	<code>m_measures</code>	894
13.86.4.14	<code>m_ppqn</code>	894
13.86.4.15	<code>m_pp_whole</code>	894
13.86.4.16	<code>m_pp_eighth</code>	894
13.86.4.17	<code>m_pp_sixteenth</code>	894
13.86.4.18	<code>m_seq</code>	894
13.86.4.19	<code>m_menubar</code>	894
13.86.4.20	<code>m_menu_tools</code>	895
13.86.4.21	<code>m_menu_zoom</code>	895
13.86.4.22	<code>m_menu_snap</code>	895
13.86.4.23	<code>m_menu_note_length</code>	895
13.86.4.24	<code>m_menu_length</code>	895
13.86.4.25	<code>m_toggle_transpose</code>	895

13.86.4.26m_image_transpose . . . . .	895
13.86.4.27m_menu_midich . . . . .	895
13.86.4.28m_menu_midibus . . . . .	896
13.86.4.29m_menu_data . . . . .	896
13.86.4.30m_menu_key . . . . .	896
13.86.4.31m_menu_scale . . . . .	896
13.86.4.32m_menu_chords . . . . .	896
13.86.4.33m_menu_sequences . . . . .	896
13.86.4.34m_menu_bpm . . . . .	896
13.86.4.35m_menu_bw . . . . .	896
13.86.4.36m_menu_rec_vol . . . . .	897
13.86.4.37m_vadjust . . . . .	897
13.86.4.38m_hadjust . . . . .	897
13.86.4.39m_vscroll_new . . . . .	897
13.86.4.40m_hscroll_new . . . . .	897
13.86.4.41m_seqkeys_wid . . . . .	897
13.86.4.42m_seqtime_wid . . . . .	897
13.86.4.43m_seqdata_wid . . . . .	897
13.86.4.44m_seqevent_wid . . . . .	898
13.86.4.45m_seqroll_wid . . . . .	898
13.86.4.46m_button_lfo . . . . .	898
13.86.4.47m_lfo_wnd . . . . .	898
13.86.4.48m_table . . . . .	898
13.86.4.49m_vbox . . . . .	898
13.86.4.50m_hbox . . . . .	898
13.86.4.51m_hbox2 . . . . .	898
13.86.4.52m_button_undo . . . . .	899
13.86.4.53m_button_redo . . . . .	899
13.86.4.54m_button_quantize . . . . .	899
13.86.4.55m_button_tools . . . . .	899



13.86.4.56m_button_sequence . . . . .	899
13.86.4.57m_entry_sequence . . . . .	899
13.86.4.58m_button_bus . . . . .	899
13.86.4.59m_entry_bus . . . . .	899
13.86.4.60m_button_channel . . . . .	900
13.86.4.61m_entry_channel . . . . .	900
13.86.4.62m_button_snap . . . . .	900
13.86.4.63m_entry_snap . . . . .	900
13.86.4.64m_button_note_length . . . . .	900
13.86.4.65m_entry_note_length . . . . .	900
13.86.4.66m_button_zoom . . . . .	900
13.86.4.67m_entry_zoom . . . . .	900
13.86.4.68m_button_length . . . . .	901
13.86.4.69m_entry_length . . . . .	901
13.86.4.70m_button_key . . . . .	901
13.86.4.71m_entry_key . . . . .	901
13.86.4.72m_button_scale . . . . .	901
13.86.4.73m_entry_scale . . . . .	901
13.86.4.74m_button_chord . . . . .	901
13.86.4.75m_entry_chord . . . . .	901
13.86.4.76m_tooltips . . . . .	902
13.86.4.77m_button_data . . . . .	902
13.86.4.78m_entry_data . . . . .	902
13.86.4.79m_button_bpm . . . . .	902
13.86.4.80m_entry_bpm . . . . .	902
13.86.4.81m_button_bw . . . . .	902
13.86.4.82m_entry_bw . . . . .	902
13.86.4.83m_button_rec_vol . . . . .	902
13.86.4.84m_toggle_play . . . . .	903
13.86.4.85m_toggle_record . . . . .	903

13.86.4.86m_toggle_q_rec . . . . .	903
13.86.4.87m_toggle_thru . . . . .	903
13.86.4.88m_entry_name . . . . .	903
13.86.4.89m_editing_status . . . . .	903
13.86.4.90m_editing_cc . . . . .	903
13.86.4.91m_have_focus . . . . .	903
13.87seq64::sequevent Class Reference . . . . .	904
13.87.1 Constructor & Destructor Documentation . . . . .	907
13.87.1.1 sequevent() . . . . .	907
13.87.1.2 ~sequevent() . . . . .	908
13.87.2 Member Function Documentation . . . . .	908
13.87.2.1 reset() . . . . .	908
13.87.2.2 redraw() . . . . .	908
13.87.2.3 set_zoom() . . . . .	908
13.87.2.4 set_snap() . . . . .	908
13.87.2.5 set_data_type() . . . . .	908
13.87.2.6 update_sizes() . . . . .	909
13.87.2.7 draw_background() . . . . .	909
13.87.2.8 draw_events_on_pixmap() . . . . .	909
13.87.2.9 draw_pixmap_on_window() . . . . .	909
13.87.2.10draw_selection_on_window() . . . . .	909
13.87.2.11update_pixmap() . . . . .	910
13.87.2.12force_draw() . . . . .	910
13.87.2.13dle_redraw() . . . . .	910
13.87.2.14x_to_w() . . . . .	910
13.87.2.15drop_event() . . . . .	910
13.87.2.16draw_events_on() . . . . .	911
13.87.2.17start_paste() . . . . .	911
13.87.2.18change_horz() . . . . .	911
13.87.2.19convert_x() . . . . .	911

13.87.2.20	<code>convert_t()</code>	912
13.87.2.21	<code>snap_y()</code>	912
13.87.2.22	<code>snap_x()</code>	912
13.87.2.23	<code>on_realize()</code>	913
13.87.2.24	<code>on_expose_event()</code>	913
13.87.2.25	<code>on_button_press_event()</code>	913
13.87.2.26	<code>on_button_release_event()</code>	913
13.87.2.27	<code>on_motion_notify_event()</code>	914
13.87.2.28	<code>on_focus_in_event()</code>	914
13.87.2.29	<code>on_focus_out_event()</code>	914
13.87.2.30	<code>on_key_press_event()</code>	915
13.87.2.31	<code>on_size_allocate()</code>	915
13.87.3	Friends And Related Function Documentation	915
13.87.3.1	<code>FruitySeqEventInput</code>	915
13.87.3.2	<code>Seq24SeqEventInput</code>	915
13.87.4	Field Documentation	916
13.87.4.1	<code>m_fruity_interaction</code>	916
13.87.4.2	<code>m_seq24_interaction</code>	916
13.87.4.3	<code>m_seq</code>	916
13.87.4.4	<code>m_zoom</code>	916
13.87.4.5	<code>m_snap</code>	916
13.87.4.6	<code>m_ppqn</code>	916
13.87.4.7	<code>m_old</code>	916
13.87.4.8	<code>m_selected</code>	917
13.87.4.9	<code>m_scroll_offset_ticks</code>	917
13.87.4.10	<code>m_scroll_offset_x</code>	917
13.87.4.11	<code>m_seqdata_wid</code>	917
13.87.4.12	<code>m_selecting</code>	917
13.87.4.13	<code>m_moving_init</code>	917
13.87.4.14	<code>m_moving</code>	917

13.87.4.15m_growing . . . . .	917
13.87.4.16m_painting . . . . .	918
13.87.4.17m_paste . . . . .	918
13.87.4.18m_move_snap_offset_x . . . . .	918
13.87.4.19m_status . . . . .	918
13.87.4.20m_cc . . . . .	918
13.88seq64::seqkeys Class Reference . . . . .	918
13.88.1 Detailed Description . . . . .	921
13.88.2 Constructor & Destructor Documentation . . . . .	921
13.88.2.1 seqkeys() . . . . .	921
13.88.2.2 ~seqkeys() . . . . .	922
13.88.3 Member Function Documentation . . . . .	922
13.88.3.1 set_scale() . . . . .	922
13.88.3.2 set_key() . . . . .	922
13.88.3.3 set_hint_key() . . . . .	922
13.88.3.4 set_hint_state() . . . . .	923
13.88.3.5 force_draw() . . . . .	923
13.88.3.6 set_listen_button_press() . . . . .	923
13.88.3.7 set_listen_button_release() . . . . .	923
13.88.3.8 set_listen_motion_notify() . . . . .	924
13.88.3.9 draw_area() . . . . .	924
13.88.3.10update_pixmap() . . . . .	924
13.88.3.11convert_y() . . . . .	924
13.88.3.12draw_key() . . . . .	924
13.88.3.13change_vert() . . . . .	925
13.88.3.14update_sizes() . . . . .	925
13.88.3.15reset() . . . . .	925
13.88.3.16s_black_key() . . . . .	925
13.88.3.17on_realize() . . . . .	926
13.88.3.18on_expose_event() . . . . .	926

13.88.3.19	<a href="#">on_button_press_event()</a>	926
13.88.3.20	<a href="#">on_button_release_event()</a>	926
13.88.3.21	<a href="#">on_motion_notify_event()</a>	927
13.88.3.22	<a href="#">on_enter_notify_event()</a>	927
13.88.3.23	<a href="#">on_leave_notify_event()</a>	927
13.88.3.24	<a href="#">on_scroll_event()</a>	927
13.88.3.25	<a href="#">on_size_allocate()</a>	928
13.88.4	<a href="#">Friends And Related Function Documentation</a>	928
13.88.4.1	<a href="#">seqroll</a>	928
13.88.4.2	<a href="#">FruitySeqRollInput</a>	928
13.88.5	<a href="#">Field Documentation</a>	928
13.88.5.1	<a href="#">m_seq</a>	928
13.88.5.2	<a href="#">m_scroll_offset_key</a>	929
13.88.5.3	<a href="#">m_scroll_offset_y</a>	929
13.88.5.4	<a href="#">m_hint_state</a>	929
13.88.5.5	<a href="#">m_hint_key</a>	929
13.88.5.6	<a href="#">m_keying</a>	929
13.88.5.7	<a href="#">m_keying_note</a>	929
13.88.5.8	<a href="#">m_scale</a>	929
13.88.5.9	<a href="#">m_key</a>	929
13.88.5.10	<a href="#">m_show_octave_letters</a>	930
13.89	<a href="#">seq64::seqmenu Class Reference</a>	930
13.89.1	<a href="#">Detailed Description</a>	934
13.89.2	<a href="#">Member Typedef Documentation</a>	934
13.89.2.1	<a href="#">SeqeditMap</a>	934
13.89.2.2	<a href="#">SeqeditPair</a>	934
13.89.2.3	<a href="#">iterator</a>	934
13.89.2.4	<a href="#">const_iterator</a>	934
13.89.3	<a href="#">Constructor &amp; Destructor Documentation</a>	934
13.89.3.1	<a href="#">seqmenu()</a>	934

13.89.3.2 ~seqmenu()	935
13.89.4 Member Function Documentation	935
13.89.4.1 current_seq() [1/2]	935
13.89.4.2 is_modified() [1/2]	935
13.89.4.3 current_seq() [2/2]	935
13.89.4.4 set_edit_sequence()	935
13.89.4.5 unset_edit_sequence()	936
13.89.4.6 is_edit_sequence()	936
13.89.4.7 is_modified() [2/2]	936
13.89.4.8 get_current_sequence()	936
13.89.4.9 get_sequence()	936
13.89.4.10 is_current_seq_active()	936
13.89.4.11 is_current_seq_in_edit()	936
13.89.4.12 new_current_sequence()	937
13.89.4.13 new_sequence()	937
13.89.4.14 delete_current_sequence()	937
13.89.4.15 toggle_current_sequence()	937
13.89.4.16 popup_menu()	937
13.89.4.17 seq_edit()	937
13.89.4.18 seq_event_edit()	938
13.89.4.19 create_seqedit()	938
13.89.4.20 remove_seqedit()	938
13.89.4.21 seq_set_and_edit()	938
13.89.4.22 seq_set_and_eventedit()	939
13.89.4.23 redraw()	939
13.89.4.24 seq_new()	939
13.89.4.25 seq_copy()	939
13.89.4.26 seq_cut()	940
13.89.4.27 seq_paste()	940
13.89.4.28 seq_clear_perf()	940

13.89.4.29	set_bus_and_midi_channel()	940
13.89.4.30	set_transposable()	940
13.89.4.31	mute_all_tracks()	941
13.89.4.32	unmute_all_tracks()	941
13.89.4.33	toggle_all_tracks()	941
13.89.4.34	on_realize()	941
13.89.5	Friends And Related Function Documentation	941
13.89.5.1	mainwnd	941
13.89.5.2	seqedit	941
13.89.6	Field Documentation	942
13.89.6.1	sm_seqedit_list	942
13.89.6.2	m_menu	942
13.89.6.3	m_mainperf	942
13.89.6.4	m_clipboard	942
13.89.6.5	m_seqedit	942
13.89.6.6	m_eventedit	942
13.89.6.7	m_current_seq	943
13.89.6.8	m_modified	943
13.90	seq64::seqroll Class Reference	943
13.90.1	Constructor & Destructor Documentation	949
13.90.1.1	seqroll()	949
13.90.1.2	~seqroll()	950
13.90.2	Member Function Documentation	950
13.90.2.1	set_snap()	950
13.90.2.2	set_zoom()	950
13.90.2.3	set_note_length()	951
13.90.2.4	note_off_length()	951
13.90.2.5	add_note()	951
13.90.2.6	add_chord()	951
13.90.2.7	set_key()	952

13.90.2.8 <code>set_scale()</code> . . . . .	952
13.90.2.9 <code>set_chord()</code> . . . . .	952
13.90.2.10 <code>set_data_type()</code> . . . . .	952
13.90.2.11 <code>set_background_sequence()</code> . . . . .	953
13.90.2.12 <code>update_pixmap()</code> . . . . .	953
13.90.2.13 <code>update_sizes()</code> . . . . .	953
13.90.2.14 <code>update_background()</code> . . . . .	953
13.90.2.15 <code>draw_background_on_pixmap()</code> . . . . .	953
13.90.2.16 <code>draw_events_on_pixmap()</code> . . . . .	954
13.90.2.17 <code>draw_selection_on_window()</code> . . . . .	954
13.90.2.18 <code>draw_progress_on_window()</code> . . . . .	954
13.90.2.19 <code>reset()</code> . . . . .	954
13.90.2.20 <code>update_and_draw()</code> . . . . .	954
13.90.2.21 <code>redraw()</code> . . . . .	955
13.90.2.22 <code>redraw_events()</code> . . . . .	955
13.90.2.23 <code>start_paste()</code> . . . . .	955
13.90.2.24 <code>complete_paste()</code> [1/2] . . . . .	955
13.90.2.25 <code>complete_paste()</code> [2/2] . . . . .	955
13.90.2.26 <code>follow_progress()</code> . . . . .	955
13.90.2.27 <code>force_draw()</code> . . . . .	955
13.90.2.28 <code>horizontal_adjust()</code> . . . . .	955
13.90.2.29 <code>vertical_adjust()</code> . . . . .	956
13.90.2.30 <code>snap_y()</code> . . . . .	956
13.90.2.31 <code>snap_x()</code> . . . . .	956
13.90.2.32 <code>convert_xy()</code> . . . . .	957
13.90.2.33 <code>convert_tn()</code> . . . . .	957
13.90.2.34 <code>xy_to_rect()</code> . . . . .	957
13.90.2.35 <code>convert_tn_box_to_rect()</code> . . . . .	958
13.90.2.36 <code>convert_sel_box_to_rect()</code> . . . . .	958
13.90.2.37 <code>get_selected_box()</code> . . . . .	959



13.90.2.38	<code>draw_events_on()</code>	959
13.90.2.39	<code>idle_redraw()</code>	959
13.90.2.40	<code>idle_progress()</code>	960
13.90.2.41	<code>change_horz()</code>	960
13.90.2.42	<code>change_vert()</code>	960
13.90.2.43	<code>move_selection_box()</code>	960
13.90.2.44	<code>move_selected_notes()</code>	960
13.90.2.45	<code>grow_selected_notes()</code>	961
13.90.2.46	<code>set_adding()</code>	961
13.90.2.47	<code>update_mouse_pointer()</code>	961
13.90.2.48	<code>button_press_initial()</code>	962
13.90.2.49	<code>align_selection()</code>	962
13.90.2.50	<code>button_press()</code>	962
13.90.2.51	<code>button_release()</code>	962
13.90.2.52	<code>motion_notify()</code>	963
13.90.2.53	<code>clear_selected()</code>	963
13.90.2.54	<code>clear_old()</code>	963
13.90.2.55	<code>clear_flags()</code>	963
13.90.2.56	<code>scroll_offset_x()</code>	964
13.90.2.57	<code>scroll_offset_y()</code>	965
13.90.2.58	<code>set_current_offset_x_y()</code>	965
13.90.2.59	<code>adding()</code>	966
13.90.2.60	<code>selecting()</code>	966
13.90.2.61	<code>growing()</code>	966
13.90.2.62	<code>normal_action()</code>	966
13.90.2.63	<code>select_action()</code>	966
13.90.2.64	<code>drop_action()</code>	966
13.90.2.65	<code>moving()</code>	967
13.90.2.66	<code>on_realize()</code>	967
13.90.2.67	<code>on_expose_event()</code>	967

13.90.2.68	<code>on_button_press_event()</code>	967
13.90.2.69	<code>on_button_release_event()</code>	967
13.90.2.70	<code>on_motion_notify_event()</code>	968
13.90.2.71	<code>on_focus_in_event()</code>	968
13.90.2.72	<code>on_focus_out_event()</code>	969
13.90.2.73	<code>on_key_press_event()</code>	969
13.90.2.74	<code>on_scroll_event()</code>	969
13.90.2.75	<code>on_size_allocate()</code>	970
13.90.2.76	<code>on_leave_notify_event()</code>	970
13.90.2.77	<code>on_enter_notify_event()</code>	970
13.90.3	Friends And Related Function Documentation	971
13.90.3.1	<code>FruitySeqRollInput</code>	971
13.90.4	Field Documentation	971
13.90.4.1	<code>m_horizontal_adjust</code>	971
13.90.4.2	<code>m_vertical_adjust</code>	971
13.90.4.3	<code>m_old</code>	971
13.90.4.4	<code>m_selected</code>	971
13.90.4.5	<code>m_seq</code>	972
13.90.4.6	<code>m_seqkeys_wid</code>	972
13.90.4.7	<code>m_fruity_interaction</code>	972
13.90.4.8	<code>m_pos</code>	972
13.90.4.9	<code>m_zoom</code>	972
13.90.4.10	<code>m_snap</code>	972
13.90.4.11	<code>m_ppqn</code>	972
13.90.4.12	<code>m_note_length</code>	972
13.90.4.13	<code>m_scale</code>	973
13.90.4.14	<code>m_chord</code>	973
13.90.4.15	<code>m_key</code>	973
13.90.4.16	<code>m_adding</code>	973
13.90.4.17	<code>m_selecting</code>	973

13.90.4.18m_moving . . . . .	973
13.90.4.19m_moving_init . . . . .	973
13.90.4.20m_growing . . . . .	973
13.90.4.21m_painting . . . . .	974
13.90.4.22m_paste . . . . .	974
13.90.4.23m_is_drag_pasting . . . . .	974
13.90.4.24m_is_drag_pasting_start . . . . .	974
13.90.4.25m_justselected_one . . . . .	974
13.90.4.26m_move_delta_x . . . . .	974
13.90.4.27m_move_delta_y . . . . .	974
13.90.4.28m_move_snap_offset_x . . . . .	974
13.90.4.29m_progress_x . . . . .	975
13.90.4.30m_scroll_offset_ticks . . . . .	975
13.90.4.31m_scroll_offset_key . . . . .	975
13.90.4.32m_scroll_offset_x . . . . .	975
13.90.4.33m_scroll_offset_y . . . . .	975
13.90.4.34m_transport_follow . . . . .	975
13.90.4.35m_trans_button_press . . . . .	975
13.90.4.36m_background_sequence . . . . .	975
13.90.4.37m_drawing_background_seq . . . . .	976
13.90.4.38m_status . . . . .	976
13.90.4.39m_cc . . . . .	976
13.91seq64::seqtime Class Reference . . . . .	976
13.91.1 Constructor & Destructor Documentation . . . . .	978
13.91.1.1 seqtime() . . . . .	979
13.91.1.2 ~seqtime() . . . . .	979
13.91.2 Member Function Documentation . . . . .	979
13.91.2.1 reset() . . . . .	979
13.91.2.2 redraw() . . . . .	979
13.91.2.3 set_zoom() . . . . .	979

13.91.2.4 draw_pixmap_on_window()	979
13.91.2.5 draw_progress_on_window()	980
13.91.2.6 update_pixmap()	980
13.91.2.7 change_horz()	980
13.91.2.8 update_sizes()	981
13.91.2.9 idle_progress()	981
13.91.2.10 on_realize()	981
13.91.2.11 on_expose_event()	981
13.91.2.12 on_size_allocate()	981
13.91.2.13 on_button_press_event()	981
13.91.2.14 on_button_release_event()	981
13.91.3 Field Documentation	982
13.91.3.1 m_seq	982
13.91.3.2 m_scroll_offset_ticks	982
13.91.3.3 m_scroll_offset_x	982
13.91.3.4 m_zoom	982
13.91.3.5 m_ppqn	982
13.92seq64::sequence Class Reference	982
13.92.1 Detailed Description	993
13.92.2 Member Typedef Documentation	993
13.92.2.1 EventStack	993
13.92.3 Member Enumeration Documentation	993
13.92.3.1 select_action_e	993
13.92.4 Constructor & Destructor Documentation	994
13.92.4.1 sequence()	994
13.92.4.2 ~sequence()	994
13.92.5 Member Function Documentation	994
13.92.5.1 operator=()	994
13.92.5.2 partial_assign()	994
13.92.5.3 events() [1/2]	995

13.92.5.4 events() [2/2]	995
13.92.5.5 any_selected_notes()	995
13.92.5.6 triggerlist() [1/2]	995
13.92.5.7 triggerlist() [2/2]	995
13.92.5.8 get_trigger_count()	995
13.92.5.9 set_trigger_paste_tick()	995
13.92.5.10get_trigger_paste_tick()	996
13.92.5.11number() [1/2]	996
13.92.5.12number() [2/2]	996
13.92.5.13modify()	996
13.92.5.14event_count()	996
13.92.5.15set_hold_undo()	996
13.92.5.16get_hold_undo()	997
13.92.5.17set_have_undo()	997
13.92.5.18have_undo()	997
13.92.5.19set_have_redo()	997
13.92.5.20have_redo()	997
13.92.5.21push_undo()	997
13.92.5.22pop_undo()	998
13.92.5.23pop_redo()	998
13.92.5.24push_trigger_undo()	998
13.92.5.25pop_trigger_undo()	998
13.92.5.26pop_trigger_redo()	998
13.92.5.27set_name() [1/2]	998
13.92.5.28set_name() [2/2]	999
13.92.5.29set_measures()	999
13.92.5.30get_measures()	999
13.92.5.31get_ppqn()	999
13.92.5.32set_beats_per_bar()	999
13.92.5.33get_beats_per_bar()	999

13.92.5.34	<code>set_beat_width()</code>	1000
13.92.5.35	<code>get_beat_width()</code>	1000
13.92.5.36	<code>measures_to_ticks()</code>	1000
13.92.5.37	<code>clocks_per_metronome()</code> [1/2]	1000
13.92.5.38	<code>clocks_per_metronome()</code> [2/2]	1000
13.92.5.39	<code>set_32nds_per_quarter()</code>	1000
13.92.5.40	<code>get_32nds_per_quarter()</code>	1001
13.92.5.41	<code>lus_per_quarter_note()</code> [1/2]	1001
13.92.5.42	<code>lus_per_quarter_note()</code> [2/2]	1001
13.92.5.43	<code>set_rec_vol()</code>	1001
13.92.5.44	<code>set_song_mute()</code>	1001
13.92.5.45	<code>toggle_song_mute()</code>	1001
13.92.5.46	<code>get_song_mute()</code>	1002
13.92.5.47	<code>apply_song_transpose()</code>	1002
13.92.5.48	<code>set_transposable()</code>	1002
13.92.5.49	<code>get_transposable()</code>	1002
13.92.5.50	<code>get_name()</code>	1002
13.92.5.51	<code>name()</code>	1002
13.92.5.52	<code>set_editing()</code>	1002
13.92.5.53	<code>get_editing()</code>	1003
13.92.5.54	<code>set_raise()</code>	1003
13.92.5.55	<code>get_raise()</code>	1003
13.92.5.56	<code>set_length()</code>	1003
13.92.5.57	<code>get_length()</code>	1004
13.92.5.58	<code>get_last_tick()</code>	1004
13.92.5.59	<code>set_last_tick()</code>	1004
13.92.5.60	<code>mod_last_tick()</code>	1004
13.92.5.61	<code>set_playing()</code>	1004
13.92.5.62	<code>get_playing()</code>	1005
13.92.5.63	<code>toggle_playing()</code>	1005

13.92.5.64	<a href="#">toggle_queued()</a>	1005
13.92.5.65	<a href="#">off_queued()</a>	1005
13.92.5.66	<a href="#">on_queued()</a>	1005
13.92.5.67	<a href="#">get_queued()</a>	1005
13.92.5.68	<a href="#">get_queued_tick()</a>	1006
13.92.5.69	<a href="#">check_queued_tick()</a>	1006
13.92.5.70	<a href="#">set_recording()</a>	1006
13.92.5.71	<a href="#">get_recording()</a>	1006
13.92.5.72	<a href="#">set_snap_tick()</a>	1006
13.92.5.73	<a href="#">set_quantized_rec()</a>	1006
13.92.5.74	<a href="#">get_quantized_rec()</a>	1006
13.92.5.75	<a href="#">set_thru()</a>	1007
13.92.5.76	<a href="#">get_thru()</a>	1007
13.92.5.77	<a href="#">s_dirty_main()</a>	1007
13.92.5.78	<a href="#">s_dirty_edit()</a>	1007
13.92.5.79	<a href="#">s_dirty_perf()</a>	1007
13.92.5.80	<a href="#">s_dirty_names()</a>	1008
13.92.5.81	<a href="#">set_dirty_mp()</a>	1008
13.92.5.82	<a href="#">set_dirty()</a>	1008
13.92.5.83	<a href="#">get_midi_channel()</a>	1008
13.92.5.84	<a href="#">s_smf_0()</a>	1008
13.92.5.85	<a href="#">set_midi_channel()</a>	1008
13.92.5.86	<a href="#">print()</a>	1009
13.92.5.87	<a href="#">print_triggers()</a>	1009
13.92.5.88	<a href="#">play()</a>	1009
13.92.5.89	<a href="#">play_queue()</a>	1010
13.92.5.90	<a href="#">add_note()</a>	1010
13.92.5.91	<a href="#">add_event()</a> [1/2]	1011
13.92.5.92	<a href="#">add_chord()</a>	1012
13.92.5.93	<a href="#">add_event()</a> [2/2]	1012

13.92.5.94	<a href="#">append_event()</a> . . . . .	1013
13.92.5.95	<a href="#">sort_events()</a> . . . . .	1013
13.92.5.96	<a href="#">add_trigger()</a> . . . . .	1013
13.92.5.97	<a href="#">split_trigger()</a> . . . . .	1013
13.92.5.98	<a href="#">grow_trigger()</a> . . . . .	1014
13.92.5.99	<a href="#">del_trigger()</a> . . . . .	1014
13.92.5.100	<a href="#">get_trigger_state()</a> . . . . .	1014
13.92.5.101	<a href="#">select_trigger()</a> . . . . .	1015
13.92.5.102	<a href="#">get_triggers()</a> . . . . .	1015
13.92.5.103	<a href="#">deselect_triggers()</a> . . . . .	1015
13.92.5.104	<a href="#">intersect_triggers()</a> . . . . .	1016
13.92.5.105	<a href="#">intersect_notes()</a> . . . . .	1016
13.92.5.106	<a href="#">intersect_events()</a> . . . . .	1017
13.92.5.107	<a href="#">del_selected_trigger()</a> . . . . .	1017
13.92.5.108	<a href="#">cut_selected_trigger()</a> . . . . .	1017
13.92.5.109	<a href="#">copy_selected_trigger()</a> . . . . .	1017
13.92.5.110	<a href="#">paste_trigger()</a> . . . . .	1017
13.92.5.111	<a href="#">move_selected_triggers_to()</a> . . . . .	1018
13.92.5.112	<a href="#">selected_trigger_start()</a> . . . . .	1018
13.92.5.113	<a href="#">selected_trigger_end()</a> . . . . .	1019
13.92.5.114	<a href="#">get_max_trigger()</a> . . . . .	1019
13.92.5.115	<a href="#">move_triggers()</a> . . . . .	1019
13.92.5.116	<a href="#">copy_triggers()</a> . . . . .	1019
13.92.5.117	<a href="#">clear_triggers()</a> . . . . .	1020
13.92.5.118	<a href="#">get_trigger_offset()</a> . . . . .	1020
13.92.5.119	<a href="#">set_midi_bus()</a> . . . . .	1020
13.92.5.120	<a href="#">get_midi_bus()</a> . . . . .	1020
13.92.5.121	<a href="#">set_master_midi_bus()</a> . . . . .	1020
13.92.5.122	<a href="#">select_note_events()</a> . . . . .	1021
13.92.5.123	<a href="#">select_events()</a> [ 1 / 3 ] . . . . .	1021



13.92.5.124	<a href="#">select_events() [2/3]</a>	1022
13.92.5.125	<a href="#">select_events() [3/3]</a>	1023
13.92.5.126	<a href="#">select_event_handle()</a>	1023
13.92.5.127	<a href="#">select_linked()</a>	1023
13.92.5.128	<a href="#">select_even_or_odd_notes()</a>	1024
13.92.5.129	<a href="#">select_all_notes()</a>	1024
13.92.5.130	<a href="#">get_num_selected_notes()</a>	1024
13.92.5.131	<a href="#">get_num_selected_events()</a>	1025
13.92.5.132	<a href="#">select_all()</a>	1025
13.92.5.133	<a href="#">copy_selected()</a>	1025
13.92.5.134	<a href="#">cut_selected()</a>	1025
13.92.5.135	<a href="#">paste_selected()</a>	1026
13.92.5.136	<a href="#">get_selected_box()</a>	1027
13.92.5.137	<a href="#">get_clipboard_box()</a>	1027
13.92.5.138	<a href="#">adjust_timestamp()</a>	1027
13.92.5.139	<a href="#">trim_timestamp()</a>	1028
13.92.5.140	<a href="#">flip_timestamp()</a>	1028
13.92.5.141	<a href="#">move_selected_notes()</a>	1029
13.92.5.142	<a href="#">stream_event()</a>	1030
13.92.5.143	<a href="#">change_event_data_range()</a>	1030
13.92.5.144	<a href="#">change_event_data_lfo()</a>	1031
13.92.5.145	<a href="#">increment_selected()</a>	1032
13.92.5.146	<a href="#">decrement_selected()</a>	1032
13.92.5.147	<a href="#">grow_selected()</a>	1033
13.92.5.148	<a href="#">stretch_selected()</a>	1033
13.92.5.149	<a href="#">remove_marked()</a>	1034
13.92.5.150	<a href="#">mark_selected()</a>	1034
13.92.5.151	<a href="#">remove_selected()</a>	1034
13.92.5.152	<a href="#">repaint_all()</a>	1034
13.92.5.153	<a href="#">unselect()</a>	1034

13.92.5.154	<a href="#">verify_and_link()</a> . . . . .	1035
13.92.5.155	<a href="#">link_new()</a> . . . . .	1035
13.92.5.156	<a href="#">zero_markers()</a> . . . . .	1035
13.92.5.157	<a href="#">play_note_on()</a> . . . . .	1035
13.92.5.158	<a href="#">play_note_off()</a> . . . . .	1035
13.92.5.159	<a href="#">stop_playing_notes()</a> . . . . .	1036
13.92.5.160	<a href="#">stop()</a> . . . . .	1036
13.92.5.161	<a href="#">pause()</a> . . . . .	1036
13.92.5.162	<a href="#">reset_draw_marker()</a> . . . . .	1036
13.92.5.163	<a href="#">reset_draw_trigger_marker()</a> . . . . .	1036
13.92.5.164	<a href="#">get_next_note_event()</a> . . . . .	1037
13.92.5.165	<a href="#">get_minmax_note_events()</a> . . . . .	1038
13.92.5.166	<a href="#">get_next_event()</a> [1/2] . . . . .	1038
13.92.5.167	<a href="#">get_next_event()</a> [2/2] . . . . .	1039
13.92.5.168	<a href="#">get_next_trigger()</a> . . . . .	1039
13.92.5.169	<a href="#">quantize_events()</a> . . . . .	1039
13.92.5.170	<a href="#">push_quantize()</a> . . . . .	1040
13.92.5.171	<a href="#">transpose_notes()</a> . . . . .	1040
13.92.5.172	<a href="#">shift_notes()</a> . . . . .	1041
13.92.5.173	<a href="#">multiply_pattern()</a> . . . . .	1041
13.92.5.174	<a href="#">musical_key()</a> [1/2] . . . . .	1041
13.92.5.175	<a href="#">musical_key()</a> [2/2] . . . . .	1041
13.92.5.176	<a href="#">musical_scale()</a> [1/2] . . . . .	1041
13.92.5.177	<a href="#">musical_scale()</a> [2/2] . . . . .	1041
13.92.5.178	<a href="#">background_sequence()</a> [1/2] . . . . .	1042
13.92.5.179	<a href="#">background_sequence()</a> [2/2] . . . . .	1042
13.92.5.180	<a href="#">show_events()</a> . . . . .	1042
13.92.5.181	<a href="#">copy_events()</a> . . . . .	1042
13.92.5.182	<a href="#">note_off_margin()</a> . . . . .	1042
13.92.5.183	<a href="#">event_in_range()</a> . . . . .	1043

13.92.5.184	<a href="#">set_parent()</a>	1043
13.92.5.185	<a href="#">put_event_on_bus()</a>	1043
13.92.5.186	<a href="#">set_trigger_offset()</a>	1044
13.92.5.187	<a href="#">adjust_trigger_offsets_to_length()</a>	1044
13.92.5.188	<a href="#">adjust_offset()</a>	1044
13.92.5.189	<a href="#">remove()</a> [1/2]	1044
13.92.5.190	<a href="#">remove()</a> [2/2]	1045
13.92.5.191	<a href="#">remove_all()</a>	1045
13.92.5.192	<a href="#">channel_match()</a>	1045
13.92.6	Friends And Related Function Documentation	1045
13.92.6.1	<a href="#">perform</a>	1046
13.92.6.2	<a href="#">triggers</a>	1046
13.92.7	Field Documentation	1046
13.92.7.1	<a href="#">m_events_clipboard</a>	1046
13.92.7.2	<a href="#">m_parent</a>	1046
13.92.7.3	<a href="#">m_events</a>	1046
13.92.7.4	<a href="#">m_triggers</a>	1046
13.92.7.5	<a href="#">m_events_undo_hold</a>	1046
13.92.7.6	<a href="#">m_have_undo</a>	1047
13.92.7.7	<a href="#">m_have_redo</a>	1047
13.92.7.8	<a href="#">m_events_undo</a>	1047
13.92.7.9	<a href="#">m_events_redo</a>	1047
13.92.7.10	<a href="#">m_iterator_draw</a>	1047
13.92.7.11	<a href="#">m_channel_match</a>	1047
13.92.7.12	<a href="#">m_midi_channel</a>	1047
13.92.7.13	<a href="#">m_bus</a>	1048
13.92.7.14	<a href="#">m_song_mute</a>	1048
13.92.7.15	<a href="#">m_transposable</a>	1048
13.92.7.16	<a href="#">m_notes_on</a>	1048
13.92.7.17	<a href="#">m_masterbus</a>	1048

13.92.7.18m_playing_notes . . . . .	1048
13.92.7.19m_was_playing . . . . .	1048
13.92.7.20m_playing . . . . .	1048
13.92.7.21m_recording . . . . .	1049
13.92.7.22m_quantized_rec . . . . .	1049
13.92.7.23m_thru . . . . .	1049
13.92.7.24m_queued . . . . .	1049
13.92.7.25m_dirty_main . . . . .	1049
13.92.7.26m_dirty_edit . . . . .	1049
13.92.7.27m_dirty_perf . . . . .	1049
13.92.7.28m_dirty_names . . . . .	1049
13.92.7.29m_editing . . . . .	1050
13.92.7.30m_raise . . . . .	1050
13.92.7.31m_name . . . . .	1050
13.92.7.32m_last_tick . . . . .	1050
13.92.7.33m_queued_tick . . . . .	1050
13.92.7.34m_trigger_offset . . . . .	1050
13.92.7.35m_maxbeats . . . . .	1050
13.92.7.36m_ppqn . . . . .	1050
13.92.7.37m_seq_number . . . . .	1051
13.92.7.38m_length . . . . .	1051
13.92.7.39m_snap_tick . . . . .	1051
13.92.7.40m_time_beats_per_measure . . . . .	1051
13.92.7.41m_time_beat_width . . . . .	1051
13.92.7.42m_clocks_per_metronome . . . . .	1051
13.92.7.43m_32nds_per_quarter . . . . .	1051
13.92.7.44m_us_per_quarter_note . . . . .	1052
13.92.7.45m_rec_vol . . . . .	1052
13.92.7.46m_note_on_velocity . . . . .	1052
13.92.7.47m_note_off_velocity . . . . .	1052

13.92.7.48	m_musical_key	1052
13.92.7.49	m_musical_scale	1052
13.92.7.50	m_background_sequence	1052
13.92.7.51	m_mutex	1053
13.92.7.52	m_note_off_margin	1053
13.93	seq64::trigger Class Reference	1053
13.93.1	Detailed Description	1054
13.93.2	Constructor & Destructor Documentation	1054
13.93.2.1	trigger()	1054
13.93.3	Member Function Documentation	1054
13.93.3.1	operator<()	1054
13.93.3.2	length()	1054
13.93.3.3	tick_start() [1/2]	1055
13.93.3.4	tick_start() [2/2]	1055
13.93.3.5	increment_tick_start()	1055
13.93.3.6	decrement_tick_start()	1055
13.93.3.7	tick_end() [1/2]	1055
13.93.3.8	tick_end() [2/2]	1055
13.93.3.9	increment_tick_end()	1056
13.93.3.10	decrement_tick_end()	1056
13.93.3.11	offset() [1/2]	1056
13.93.3.12	offset() [2/2]	1056
13.93.3.13	increment_offset()	1056
13.93.3.14	decrement_offset()	1056
13.93.3.15	selected() [1/2]	1056
13.93.3.16	selected() [2/2]	1057
13.93.4	Field Documentation	1057
13.93.4.1	m_tick_start	1057
13.93.4.2	m_tick_end	1057
13.93.4.3	m_offset	1057

13.93.4.4 m_selected . . . . .	1057
13.94seq64::triggers Class Reference . . . . .	1057
13.94.1 Member Typedef Documentation . . . . .	1060
13.94.1.1 List . . . . .	1060
13.94.1.2 Stack . . . . .	1060
13.94.2 Member Enumeration Documentation . . . . .	1060
13.94.2.1 grow_edit_t . . . . .	1060
13.94.3 Constructor & Destructor Documentation . . . . .	1061
13.94.3.1 triggers() . . . . .	1061
13.94.3.2 ~triggers() . . . . .	1061
13.94.4 Member Function Documentation . . . . .	1061
13.94.4.1 operator=() . . . . .	1061
13.94.4.2 set_ppqn() . . . . .	1062
13.94.4.3 set_length() . . . . .	1062
13.94.4.4 triggerlist() [1/2] . . . . .	1062
13.94.4.5 triggerlist() [2/2] . . . . .	1062
13.94.4.6 push_undo() . . . . .	1062
13.94.4.7 pop_undo() . . . . .	1062
13.94.4.8 pop_redo() . . . . .	1062
13.94.4.9 print() . . . . .	1062
13.94.4.10play() . . . . .	1063
13.94.4.11add() . . . . .	1063
13.94.4.12adjust_offsets_to_length() . . . . .	1064
13.94.4.13split() [1/2] . . . . .	1064
13.94.4.14grow() . . . . .	1064
13.94.4.15remove() . . . . .	1065
13.94.4.16get_state() . . . . .	1065
13.94.4.17select() . . . . .	1065
13.94.4.18unselect() . . . . .	1066
13.94.4.19intersect() . . . . .	1066

13.94.4.20	<a href="#">remove_selected()</a>	1066
13.94.4.21	<a href="#">copy_selected()</a>	1066
13.94.4.22	<a href="#">paste()</a>	1067
13.94.4.23	<a href="#">move_selected()</a>	1067
13.94.4.24	<a href="#">get_selected_start()</a>	1067
13.94.4.25	<a href="#">get_selected_end()</a>	1068
13.94.4.26	<a href="#">get_maximum()</a>	1068
13.94.4.27	<a href="#">move()</a>	1068
13.94.4.28	<a href="#">copy()</a>	1068
13.94.4.29	<a href="#">clear()</a>	1070
13.94.4.30	<a href="#">next()</a>	1070
13.94.4.31	<a href="#">next_trigger()</a>	1070
13.94.4.32	<a href="#">reset_draw_trigger_marker()</a>	1070
13.94.4.33	<a href="#">set_trigger_paste_tick()</a>	1071
13.94.4.34	<a href="#">get_trigger_paste_tick()</a>	1071
13.94.4.35	<a href="#">adjust_offset()</a>	1071
13.94.4.36	<a href="#">split()</a> [ 2 / 2 ]	1071
13.94.5	<a href="#">Friends And Related Function Documentation</a>	1071
13.94.5.1	<a href="#">midi_container</a>	1072
13.94.5.2	<a href="#">midifile</a>	1072
13.94.5.3	<a href="#">sequence</a>	1072
13.94.5.4	<a href="#">Seq24PerfInput</a>	1072
13.94.5.5	<a href="#">FruityPerfInput</a>	1072
13.94.6	<a href="#">Field Documentation</a>	1072
13.94.6.1	<a href="#">m_parent</a>	1072
13.94.6.2	<a href="#">m_triggers</a>	1072
13.94.6.3	<a href="#">m_clipboard</a>	1073
13.94.6.4	<a href="#">m_undo_stack</a>	1073
13.94.6.5	<a href="#">m_redo_stack</a>	1073
13.94.6.6	<a href="#">m_iterator_play_trigger</a>	1073

13.94.6.7 m_iterator_draw_trigger . . . . .	1073
13.94.6.8 m_trigger_copied . . . . .	1073
13.94.6.9 m_paste_tick . . . . .	1073
13.94.6.10m_ppqn . . . . .	1073
13.94.6.11m_length . . . . .	1074
13.95seq64::user_instrument Class Reference . . . . .	1074
13.95.1 Detailed Description . . . . .	1075
13.95.2 Constructor & Destructor Documentation . . . . .	1075
13.95.2.1 user_instrument() [1/2] . . . . .	1075
13.95.2.2 user_instrument() [2/2] . . . . .	1075
13.95.3 Member Function Documentation . . . . .	1075
13.95.3.1 operator=() . . . . .	1076
13.95.3.2 is_valid() . . . . .	1076
13.95.3.3 set_defaults() . . . . .	1076
13.95.3.4 name() . . . . .	1076
13.95.3.5 controller_count() . . . . .	1076
13.95.3.6 controller_max() . . . . .	1076
13.95.3.7 controller_name() . . . . .	1076
13.95.3.8 controller_active() . . . . .	1077
13.95.3.9 set_controller() . . . . .	1077
13.95.3.10set_name() . . . . .	1077
13.95.3.11copy_definitions() . . . . .	1078
13.95.4 Field Documentation . . . . .	1078
13.95.4.1 m_is_valid . . . . .	1078
13.95.4.2 m_controller_count . . . . .	1078
13.95.4.3 m_instrument_def . . . . .	1078
13.96seq64::user_instrument_t Struct Reference . . . . .	1078
13.96.1 Field Documentation . . . . .	1079
13.96.1.1 instrument . . . . .	1079
13.96.1.2 controllers . . . . .	1079



13.96.1.3 controllers_active . . . . .	1079
13.97seq64::user_midi_bus Class Reference . . . . .	1079
13.97.1 Detailed Description . . . . .	1080
13.97.2 Constructor & Destructor Documentation . . . . .	1080
13.97.2.1 user_midi_bus() [1/2] . . . . .	1080
13.97.2.2 user_midi_bus() [2/2] . . . . .	1081
13.97.3 Member Function Documentation . . . . .	1081
13.97.3.1 operator=() . . . . .	1081
13.97.3.2 is_valid() . . . . .	1081
13.97.3.3 set_defaults() . . . . .	1081
13.97.3.4 name() . . . . .	1082
13.97.3.5 channel_count() . . . . .	1082
13.97.3.6 channel_max() . . . . .	1082
13.97.3.7 instrument() . . . . .	1082
13.97.3.8 set_instrument() . . . . .	1082
13.97.3.9 set_name() . . . . .	1083
13.97.3.10copy_definitions() . . . . .	1083
13.97.4 Field Documentation . . . . .	1083
13.97.4.1 m_is_valid . . . . .	1083
13.97.4.2 m_channel_count . . . . .	1083
13.97.4.3 m_midi_bus_def . . . . .	1083
13.98seq64::user_midi_bus_t Struct Reference . . . . .	1084
13.98.1 Field Documentation . . . . .	1084
13.98.1.1 alias . . . . .	1084
13.98.1.2 instrument . . . . .	1084
13.99seq64::user_settings Class Reference . . . . .	1084
13.99.1 Detailed Description . . . . .	1092
13.99.2 Member Typedef Documentation . . . . .	1092
13.99.2.1 Busses . . . . .	1092
13.99.2.2 Busslterator . . . . .	1092

13.99.2.3 BussConstlterator . . . . .	1093
13.99.2.4 Instruments . . . . .	1093
13.99.2.5 Instrumentlterator . . . . .	1093
13.99.2.6 InstrumentConstlterator . . . . .	1093
13.99.3 Member Enumeration Documentation . . . . .	1093
13.99.3.1 mainwid_grid_style_t . . . . .	1093
13.99.4 Constructor & Destructor Documentation . . . . .	1093
13.99.4.1 user_settings() [1/2] . . . . .	1094
13.99.4.2 user_settings() [2/2] . . . . .	1094
13.99.5 Member Function Documentation . . . . .	1094
13.99.5.1 operator=() . . . . .	1094
13.99.5.2 set_defaults() . . . . .	1094
13.99.5.3 normalize() . . . . .	1094
13.99.5.4 add_bus() . . . . .	1094
13.99.5.5 add_instrument() . . . . .	1095
13.99.5.6 bus() . . . . .	1095
13.99.5.7 instrument() . . . . .	1095
13.99.5.8 bus_count() . . . . .	1095
13.99.5.9 set_bus_instrument() . . . . .	1095
13.99.5.10bus_instrument() . . . . .	1095
13.99.5.11bus_name() . . . . .	1095
13.99.5.12nstrument_count() . . . . .	1096
13.99.5.13set_instrument_controllers() . . . . .	1096
13.99.5.14nstrument_name() [1/2] . . . . .	1096
13.99.5.15nstrument_name() [2/2] . . . . .	1096
13.99.5.16nstrument_controller_active() . . . . .	1096
13.99.5.17controller_active() . . . . .	1096
13.99.5.18nstrument_controller_name() . . . . .	1097
13.99.5.19controller_name() . . . . .	1097
13.99.5.20grid_style() [1/2] . . . . .	1097

13.99.5.21	grid_is_normal()	1097
13.99.5.22	grid_is_white()	1097
13.99.5.23	grid_is_black()	1097
13.99.5.24	grid_brackets() [1/2]	1097
13.99.5.25	mainwnd_rows() [1/2]	1098
13.99.5.26	mainwnd_cols() [1/2]	1098
13.99.5.27	seqs_in_set()	1098
13.99.5.28	mute_tracks()	1098
13.99.5.29	max_sets() [1/2]	1098
13.99.5.30	max_sequence()	1098
13.99.5.31	text_x() [1/2]	1098
13.99.5.32	text_y() [1/2]	1098
13.99.5.33	seqchars_x() [1/2]	1099
13.99.5.34	seqchars_y() [1/2]	1099
13.99.5.35	seqarea_x() [1/2]	1099
13.99.5.36	seqarea_y() [1/2]	1099
13.99.5.37	seqarea_seq_x() [1/2]	1099
13.99.5.38	seqarea_seq_y() [1/2]	1099
13.99.5.39	mainwid_border() [1/2]	1099
13.99.5.40	mainwid_spacing() [1/2]	1099
13.99.5.41	mainwid_x()	1100
13.99.5.42	mainwid_y()	1100
13.99.5.43	control_height() [1/2]	1100
13.99.5.44	zoom() [1/2]	1100
13.99.5.45	zoom() [2/2]	1100
13.99.5.46	global_seq_feature() [1/2]	1100
13.99.5.47	global_seq_feature() [2/2]	1100
13.99.5.48	seqedit_scale() [1/2]	1101
13.99.5.49	seqedit_scale() [2/2]	1101
13.99.5.50	seqedit_key() [1/2]	1101

13.99.5.51	<code>seqedit_key()</code> [2/2]	1101
13.99.5.52	<code>seqedit_bgsequence()</code> [1/2]	1101
13.99.5.53	<code>seqedit_bgsequence()</code> [2/2]	1101
13.99.5.54	<code>use_new_font()</code> [1/2]	1101
13.99.5.55	<code>allow_two_perfedits()</code> [1/2]	1102
13.99.5.56	<code>perf_h_page_increment()</code> [1/2]	1102
13.99.5.57	<code>perf_v_page_increment()</code> [1/2]	1102
13.99.5.58	<code>progress_bar_colored()</code> [1/2]	1102
13.99.5.59	<code>progress_bar_thick()</code> [1/2]	1102
13.99.5.60	<code>inverse_colors()</code> [1/2]	1102
13.99.5.61	<code>window_redraw_rate()</code> [1/2]	1102
13.99.5.62	<code>use_more_icons()</code> [1/2]	1102
13.99.5.63	<code>save_user_config()</code> [1/2]	1103
13.99.5.64	<code>save_user_config()</code> [2/2]	1103
13.99.5.65	<code>grid_brackets()</code> [2/2]	1103
13.99.5.66	<code>grid_style()</code> [2/2]	1103
13.99.5.67	<code>mainwnd_rows()</code> [2/2]	1103
13.99.5.68	<code>mainwnd_cols()</code> [2/2]	1103
13.99.5.69	<code>max_sets()</code> [2/2]	1103
13.99.5.70	<code>text_x()</code> [2/2]	1104
13.99.5.71	<code>text_y()</code> [2/2]	1104
13.99.5.72	<code>seqchars_x()</code> [2/2]	1104
13.99.5.73	<code>seqchars_y()</code> [2/2]	1104
13.99.5.74	<code>seqarea_x()</code> [2/2]	1104
13.99.5.75	<code>seqarea_y()</code> [2/2]	1104
13.99.5.76	<code>seqarea_seq_x()</code> [2/2]	1104
13.99.5.77	<code>seqarea_seq_y()</code> [2/2]	1105
13.99.5.78	<code>mainwid_border()</code> [2/2]	1105
13.99.5.79	<code>mainwid_spacing()</code> [2/2]	1105
13.99.5.80	<code>control_height()</code> [2/2]	1105

13.99.5.81	<code>dump_summary()</code>	1105
13.99.5.82	<code>midi_ppqn()</code> [1/2]	1105
13.99.5.83	<code>midi_beats_per_bar()</code> [1/2]	1105
13.99.5.84	<code>midi_beats_per_minute()</code> [1/2]	1106
13.99.5.85	<code>midi_beat_width()</code> [1/2]	1106
13.99.5.86	<code>midi_buss_override()</code> [1/2]	1106
13.99.5.87	<code>velocity_override()</code> [1/2]	1106
13.99.5.88	<code>bpm_precision()</code> [1/2]	1106
13.99.5.89	<code>bpm_step_increment()</code> [1/2]	1106
13.99.5.90	<code>bpm_page_increment()</code> [1/2]	1106
13.99.5.91	<code>min_zoom()</code>	1106
13.99.5.92	<code>max_zoom()</code>	1107
13.99.5.93	<code>baseline_ppqn()</code>	1107
13.99.5.94	<code>use_new_font()</code> [2/2]	1107
13.99.5.95	<code>allow_two_perfedits()</code> [2/2]	1107
13.99.5.96	<code>perf_h_page_increment()</code> [2/2]	1107
13.99.5.97	<code>perf_v_page_increment()</code> [2/2]	1107
13.99.5.98	<code>progress_bar_colored()</code> [2/2]	1107
13.99.5.99	<code>progress_bar_thick()</code> [2/2]	1108
13.99.5.100	<code>verse_colors()</code> [2/2]	1108
13.99.5.101	<code>window_redraw_rate()</code> [2/2]	1108
13.99.5.102	<code>use_more_icons()</code> [2/2]	1108
13.99.5.103	<code>midi_ppqn()</code> [2/2]	1108
13.99.5.104	<code>midi_buss_override()</code> [2/2]	1108
13.99.5.105	<code>velocity_override()</code> [2/2]	1108
13.99.5.106	<code>bpm_precision()</code> [2/2]	1109
13.99.5.107	<code>bpm_step_increment()</code> [2/2]	1109
13.99.5.108	<code>bpm_page_increment()</code> [2/2]	1109
13.99.5.109	<code>midi_beats_per_bar()</code> [2/2]	1109
13.99.5.110	<code>midi_beats_per_minute()</code> [2/2]	1109

13.99.5.11midi_beat_width() [2/2] . . . . .	1109
13.99.5.12private_bus() . . . . .	1109
13.99.5.13private_instrument() . . . . .	1110
13.99.6 Friends And Related Function Documentation . . . . .	1110
13.99.6.1 userfile . . . . .	1110
13.99.7 Field Documentation . . . . .	1110
13.99.7.1 m_midi_buses . . . . .	1110
13.99.7.2 m_instruments . . . . .	1110
13.99.7.3 m_grid_style . . . . .	1110
13.99.7.4 m_grid_brackets . . . . .	1111
13.99.7.5 m_mainwnd_rows . . . . .	1111
13.99.7.6 m_mainwnd_cols . . . . .	1111
13.99.7.7 m_max_sets . . . . .	1111
13.99.7.8 m_mainwid_border . . . . .	1111
13.99.7.9 m_mainwid_spacing . . . . .	1111
13.99.7.10m_control_height . . . . .	1111
13.99.7.11m_current_zoom . . . . .	1112
13.99.7.12m_global_seq_feature_save . . . . .	1112
13.99.7.13m_seqedit_scale . . . . .	1112
13.99.7.14m_seqedit_key . . . . .	1112
13.99.7.15m_seqedit_bgsequence . . . . .	1112
13.99.7.16m_use_new_font . . . . .	1113
13.99.7.17m_allow_two_perfedits . . . . .	1113
13.99.7.18m_h_perf_page_increment . . . . .	1113
13.99.7.19m_v_perf_page_increment . . . . .	1113
13.99.7.20m_progress_bar_colored . . . . .	1113
13.99.7.21m_progress_bar_thick . . . . .	1113
13.99.7.22m_inverse_colors . . . . .	1113
13.99.7.23m_window_redraw_rate_ms . . . . .	1114
13.99.7.24m_use_more_icons . . . . .	1114

13.99.7.25m_text_x . . . . .	1114
13.99.7.26m_text_y . . . . .	1114
13.99.7.27m_seqchars_x . . . . .	1114
13.99.7.28m_seqchars_y . . . . .	1115
13.99.7.29m_midi_ppqn . . . . .	1115
13.99.7.30m_midi_beats_per_measure . . . . .	1115
13.99.7.31m_midi_beats_per_minute . . . . .	1115
13.99.7.32m_midi_beat_width . . . . .	1115
13.99.7.33m_midi_buss_override . . . . .	1115
13.99.7.34m_velocity_override . . . . .	1116
13.99.7.35m_bpm_precision . . . . .	1116
13.99.7.36m_bpm_step_increment . . . . .	1116
13.99.7.37m_bpm_page_increment . . . . .	1116
13.99.7.38m_total_seqs . . . . .	1116
13.99.7.39m_seqs_in_set . . . . .	1117
13.99.7.40m_gmute_tracks . . . . .	1117
13.99.7.41m_max_sequence . . . . .	1117
13.99.7.42m_seqarea_x . . . . .	1117
13.99.7.43m_seqarea_y . . . . .	1117
13.99.7.44m_seqarea_seq_x . . . . .	1117
13.99.7.45m_seqarea_seq_y . . . . .	1118
13.99.7.46m_mainwid_x . . . . .	1118
13.99.7.47m_mainwid_y . . . . .	1118
13.99.7.48m_save_user_config . . . . .	1118
13.99.7.49mc_min_zoom . . . . .	1118
13.99.7.50mc_max_zoom . . . . .	1118
13.99.7.51mc_baseline_ppqn . . . . .	1119
13.100eq64::userfile Class Reference . . . . .	1119
13.100.1Constructor & Destructor Documentation . . . . .	1120
13.100.1.1userfile() . . . . .	1120
13.100.1.2~userfile() . . . . .	1120
13.100.2Member Function Documentation . . . . .	1120
13.100.2.1parse() . . . . .	1120
13.100.2.2write() . . . . .	1121
13.100.2.3dump_setting_summary() . . . . .	1121





# Chapter 1

## Sequencer64

**Author(s)** Chris Ahlstrom 2016-10-23

### 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

# MIDI File Parsing in Sequencer64

**Author(s)** Chris Ahlstrom 2016-02-13

### 2.1 Introduction

This section describes the parsing of a MIDI file (and a few other topics). We wanted to add the reading of SMF 0 files to *Sequencer64*. We started with the main format that is supported, SMF 1. Once we understood that we, we figured out how to split a SMF 0 tracks correctly.

We split the `midifile::parse()` function into two sections. The first section analyzes the header of the MIDI. Then, based on whether the file is SMF 1 (the normal case) or SMF 0, either the `parse_smf_1()` function or the `parse_smf_0()` function is called. The `parse_smf_0()` function creates one sequence object per channel present in the SMF 0 file, plus the original track. The last pattern slot (sequence 16) will contain the original track data, and the rest will contain common data and then channel data for each channel. After the parsing is done, all the tracks (including the original track) will be added to the performance. The user then has the option of deleting the original track, which will be the last track.

### 2.2 SMF 1 Parsing

This section describes the parsing of the header chunk, MThd, and the track chunk, MTrk.

The `midifile::parse()` function starts by opening the MIDI file, getting its file-size, pre-allocating the data vector to that size, reading all of the characters into that vector, and then closing the file.

#### 2.2.1 MIDI File Header, MThd

The data of the header is read:

Header ID:	"MThd"	<code>read_long()</code>	4 bytes
MThd length:	6	<code>read_long()</code>	4 bytes
Format:	0, 1, 2	<code>read_short()</code>	2 bytes
No. of track:	1 or more	<code>read_short()</code>	2 bytes
PPQN:	192	<code>read_short()</code>	2 bytes

The header ID and it's length are always the same values. The formats that Sequencer64 supports are 0 or 1. SMF 0 has only one track, while SMF 1 can support an arbitrary number of tracks. The last value in the header is the PPQN value, which specifies the "pulses per quarter note", which is the basic time-resolution of events in the MIDI file. Common values are 96 or 192, but higher values are also common. Sequencer64 and its precursor, Seq24, default to 192.

### 2.2.2 MIDI Track, MTrk

Sequencer64 next reads the tracks specified in the file. Each track is assumed to cover a different MIDI channel, but always the same MIDI buss. (The MIDI buss is not a data item in standard MIDI files, but it is a special data item in Seq24/Sequencer64 MIDI files.) Each track is tagged by a standard chunk marker, "MTrk". Other markers are possible, and are to be ignored, if nothing else. Here are the values read at the beginning of a track:

Track ID:	"MTrk"	read_long()	4 bytes
Track length:	varies	read_long()	4 bytes

The track length is the number of bytes that need to be read in order to get all of the data in the track.

Next, a new sequence object is created, with the PPQN value passed to its constructor. The sequence then is hooked to the master MIDI buss object. The "RunningTime" accumulator is set to 0 for that track.

Next, the parse() function loops through the rest of the track, reading data and logging it to the sequence. Let's go through the loop, which is the meat of the processing.

TODO: An empty event is created before track processing, and re-used for every track and event. This seems dangerous. We moved the event constructor two levels of nesting deeper, and it seems to work fine.

**Delta time.** The amount time that passes from one event to the next is the *delta time*. For some events, the time doesn't matter, and is set to 0. This values is a *variable length value*, also known as a "VLV" or a "varinum". It provides a way of encoding arbitrarily large values, a byte at a time. For now, just note that a varinum is 1 or more bytes, and MIDI provides a way to tell when the varinum is complete.

Delta time:	varies	read_varinum()	1 or more bytes
-------------	--------	----------------	-----------------

#### 2.2.2.1 Channel Events

**Status.** The byte after the delta time is examined by masking it against 0x80 to check the high bit. If not set, it is a "running status", it is replaced with the "last status", which is 0 at first.

Status byte:	varies	read_byte()	1 byte
--------------	--------	-------------	--------

If the high bit is set, it is a status, and is passed to the setter `event::set_status()`.

The "RunningTime" accumulator is incremented by the delta-time. The current time is adjusted as per the PPQN ratio, if needed, and passed to the setter `event::set_timestamp()`.

Now what does the status mean? First, the channel part of the status is masked out using the 0xF0 mask.

If it is a 2-data-byte event (note on, note off, aftertouch, control-change, or pitch-wheel), then the two data bytes are read:

Data byte 0:	varies	read_byte()	1 byte
Data byte 1:	varies	read_byte()	1 byte

If the status is a note-on event, with `data[1] = 0`, then it is converted to a note-off event, a fix for the output quirks of some MIDI devices, and the status of the event is amended to `EVENT_NOTE_OFF`.

If it is a 1-data-byte event (program change or channel pressure), then only data byte 0 is read.

Then the one or two data bytes are added to the event by overloads of `event::set_data()`, the event is added to the current sequence by `sequence::add_event()`, and the MIDI channel of the sequence is set by `sequence::set_midi_channel()`.

Note that this is the point where parsing could detect a change in channel, and select a new sequence to support that channel, and add the events to that sequence, if the file were SMF 0.

Also note that the channel of the sequence is set every a new channel event/status is read. This should be done once, and then simply warned about if a non-matching channel occurs.

Lastly, note that it might be better to do the sequence function calls at the end of processing the event.

## 2.2.2.2 Meta Events

If the event status masks off to 0xF0 (0xF0 to 0xFF), then it is a meta event. If the status is 0xFF, it is called a "Sequencer-specific", or "SeqSpec" event. For this kind of event, then a type byte and the length of the event are read.

Meta type:	varies	<code>read_byte()</code>	1 byte
Meta length:	varies	<code>read_varinum()</code>	1 or more bytes

If the type of the SeqSpec (0xFF) meta event is 0x7F, parsing checks to see if it is one of the Seq24 "proprietary" events. These events are tagged with various values that mask off to 0x24240000. The parser reads the tag:

Prop tag:	0x242400nn	<code>read_long()</code>	4 bytes
-----------	------------	--------------------------	---------

These tags provide a way to save and recover Seq24/Sequencer64 properties from the MIDI file: MIDI buss, MIDI channel, time signature, sequence triggers, and (new), the key, scale, and background sequence to use with the track/sequence. Any leftover data for the tagged event is let go. Unknown tags are skipped.

If the type of the SeqSpec (0xFF) meta event is 0x2F, then it is the End-of-Track marker. The current time is set using `sequence::set_length()` and then `sequence::zero_markers()` is called, and parsing is done for that track.

If the type of the SeqSpec (0xFF) meta event is 0x03, then it is the sequence name. The "length" number of bytes are read, and loaded by `sequence::set_name()`.

If the type of the SeqSpec (0xFF) meta event is 0x00, then it is the sequence number, which is read:

Seq number:	varies	<code>read_short()</code>	2 bytes
-------------	--------	---------------------------	---------

Note that the sequence number might be modified later to account for the current screenset in force for a file import operation.

Anything other SeqSpec type is simply skipped by reading the "length" number of bytes.

To summarize the process, here are the relevant event and sequence setter calls typically made while parsing a MIDI track:

1. `perform::add_sequence()`
  - (a) `sequence::sequence()`
  - (b) `sequence::set_master_midi_bus()`
  - (c) `sequence::add_event()`
    - i. `event::event()`
    - ii. `event::set_status()`
    - iii. `event::set_timestamp()`
    - iv. `event::set_data()`
  - (d) `sequence::set_midi_channel()`
  - (e) `sequence::set_length()`
  - (f) `sequence::zero_markers()`
  - (g) `sequence::set_name()`
  - (h) `sequence::set_midi_bus()`
2. `xxxxx::yyyy()`

### 2.2.3 Meta Events Summary

Here, we summarize the MIDI meta events for your edification.

1. FF 00 02 ssss: Sequence Number.
2. FF 01 len text: Text Event.
3. FF 02 len text: Copyright Notice.
4. FF 03 len text: Sequence/Track Name.
5. FF 04 len text: Instrument Name.
6. FF 05 len text: Lyric.
7. FF 06 len text: Marker.
8. FF 07 len text: Cue Point.
9. FF 08 len text: Patch/program Name.
10. FF 09 len text: Device Name.
11. FF 0A through 0F len text: Other kinds of text events.
12. FF 20 01 cc: MIDI channel (obsolete, used by Cakewalk)
13. FF 21 01 pp: MIDI port (obsolete, used by Cakewalk)
14. FF 2F 00: End of Track.
15. FF 51 03 tttttt: Set Tempo, us/qn.
16. FF 54 05 hr mn se fr ff: SMPTE Offset.
17. FF 58 04 nn dd cc bb: Time Signature.
18. FF 59 02 sf mi: Key Signature.
19. FF 7F len data: Sequencer-Specific.

The next sections describe the events that *Sequencer* tries to handle. These are

- Sequence Number (0x00)
- Track Name (0x03)
- End-of-Track (0x2F)
- Set Tempo (0x51) (Sequencer64 only)
- Time Signature (0x58) (Sequencer64 only)
- Sequencer-Specific (0x7F)
- System Exclusive (0xF0) Sort of handled, functionality incomplete..

#### 2.2.3.1 Sequence Number (0x00)

```
FF 00 02 ss ss
```

This optional event must occur at the beginning of a track, before any non-zero delta-times, and before any transmittable MIDI events. It specifies the number of a sequence.

### 2.2.3.2 Track/Sequence Name (0x03)

```
FF 03 len text
```

If in a format 0 track, or the first track in a format 1 file, the name of the sequence. Otherwise, the name of the track.

### 2.2.3.3 End of Track (0x2F)

```
FF 2F 00
```

This event is not optional. It is included so that an exact ending point may be specified for the track, so that it has an exact length, which is necessary for tracks which are looped or concatenated.

### 2.2.3.4 Set Tempo Event (0x51)

The MIDI Set Tempo meta event sets the tempo of a MIDI sequence in terms of the microseconds per quarter note. This is a meta message, so this event is never sent over MIDI ports to a MIDI device.

After the delta time, this event consists of six bytes of data:

```
FF 51 03 tt tt tt
```

Example:

```
FF 51 03 07 A1 20
```

1. 0xFF is the status byte that indicates this is a Meta event.
2. 0x51 the meta event type that signifies this is a Set Tempo event.
3. 0x03 is the length of the event, always 3 bytes.
4. The remaining three bytes carry the number of microseconds per quarter note. For example, the three bytes above form the hexadecimal value 0x07A120 (500000 decimal), which means that there are 500,000 microseconds per quarter note.

Since there are 60,000,000 microseconds per minute, the event above translates to: set the tempo to  $60,000,000 / 500,000 = 120$  quarter notes per minute (120 beats per minute). This is a 24-bit binary value, so each byte covers the full range of 0x00 to 0xFF.

This event normally appears in the first track. If not, the default tempo is 120 beats per minute. This event is important if the MIDI time division is specified in "pulses per quarter note", which does not itself define the length of the quarter note. The length of the quarter note is then determined by the Set Tempo meta event.

Representing tempos as time per beat instead of beat per time allows absolutely exact DWORD-term synchronization with a time-based sync protocol such as SMPTE time code or MIDI time code. This amount of accuracy provided by this tempo resolution allows a four-minute piece at 120 beats per minute to be accurate within 500 usec at the end of the piece.

We have now added the Tempo meta event (and the Time Signature meta event) to the track, which allows other sequencers to obtain these values from a Sequencer64 MIDI file. Here are the original headers for a normal MIDI file and its legacy (Seq24) conversion, as shown by the midicvt application:

```

hymne.asc                                hymne-ppqn-384.asc
MThd 1 4 96                              MThd 1 4 384
MTrk                                      MTrk
0 Meta SeqName "Vangelis: Hymne"        0 SeqNr 0
0 TimeSig 4/4 24 8                      0 Meta SeqName "Vangelis: Hymne"
0 Tempo 750000                          0 SeqSpec 24 24 00 08      (no triggers)
0 Meta TrkEnd                          0 SeqSpec 24 24 00 01 00  (MIDI buss 0)
TrkEnd                                  0 SeqSpec 24 24 00 06 04 04 (beats, width)
                                         0 SeqSpec 24 24 00 02 00  (MIDI ch. 0)
                                         96 Meta TrkEnd
                                         TrkEnd

```

Here is the header data that result from the new conversion, which is used if the "legacy" option is not in force:

```

MThd 1 4 192
MTrk
0 SeqNr 0
0 Meta SeqName "Vangelis: Hymne"
0 TimeSig 4/4 24 8
0 Tempo 750000
0 SeqSpec 24 24 00 08
0 SeqSpec 24 24 00 01 00
0 SeqSpec 24 24 00 06 04 04
0 SeqSpec 24 24 00 02 00
48 Meta TrkEnd
TrkEnd

```

### 2.2.3.5 Time Signature Event (0x58)

After the delta time, this event consists of seven bytes of data:

```
FF 58 04 nn dd cc bb
```

The time signature is expressed as four numbers. `nn` and `dd` represent the numerator and denominator of the time signature as it would be notated. The numerator counts the number of beats in a measure (beats per measure or beats per bar). The denominator is a negative power of two: 2 represents a quarter-note, 3 represents an eighth-note, etc. The denominator specifies the unit of the beat (e.g. 4 or 8). In Seq24/Sequencer64, this value is also called the "beat width".

The `cc` parameter expresses the number of MIDI clocks (or "ticks", or "pulses") in a metronome click. The standard MIDI clock ticks 24 times per quarter note, so a value of 6 would mean the metronome clicks every 1/8th note. A `cc` value of 6 would mean that the metronome clicks once every 1/8th of a note (quaver). This MIDI clock is different from the clock (PPQN) that determines the start time and duration of the notes.

The `bb` parameter expresses the number of notated 32nd-notes in a MIDI quarter note (24 MIDI Clocks). The usual value for this parameter is 8, though some sequencers allow the user to specify that what MIDI thinks of as a quarter note, should be notated as something else. For example, a value of 16 means that the music plays two quarter notes for each quarter note metered out by the MIDI clock, so that the music plays at double speed.

Examples:

```
FF 58 04 04 02 18 08
```

1. 0xFF is the status byte that indicates this is a Meta event.
2. 0x58 the meta event type that signifies this is a Time Signature event.



3. 0x04 is the length of the event, always 4 bytes.
4. 0x04 is the numerator of the time signature, and ranges from 0x00 to 0xFF.
5. 0x02 is the log base 2 of the denominator, and is the power to which 2 must be raised to get the denominator. Here, the denominator is 2 to 0x02, or 4, so the time signature is 4/4.
6. 0x18 is the metronome pulse in terms of the number of MIDI clock ticks per click. Assuming 24 MIDI clocks per quarter note, the value here (0x18 = 24) indicates that the metronome will tick every 24/24 quarter note. If the value of the sixth byte were 0x30 = 48, the metronome clicks every two quarter notes, i.e. every half-note.
7. 0x08 defines the number of 32nd notes per beat. This byte is usually 8 as there is usually one quarter note per beat, and one quarter note contains eight 32nd notes.

A time signature of 6/8, with a metronome click every 3rd 1/8 note, would be encoded:

```
FF 58 04 06 03 24 08
```

Remember, a 1/4 note is 24 MIDI Clocks, therefore a bar of 6/8 is 72 MIDI Clocks. Hence 3 1/8 notes is 36 (=0x24) MIDI Clocks.

There should generally be a Time Signature Meta event at the beginning of a track (at time = 0), otherwise a default 4/4 time signature will be assumed. Thereafter they can be used to effect an immediate time signature change at any point within a track.

For a format 1 MIDI file, Time Signature Meta events should only occur within the first MTrk chunk.

If a time signature event is not present in a MIDI sequence, 4/4 signature is assumed.

In *Sequencer64*, the `c_timesig SeqSpec` event is given priority. The conventional time signature is used only if the `c_timesig SeqSpec` is not present in the file. NEEDS TO BE TESTED.

### 2.2.3.6 SysEx Event (0xF0)

If the meta event status value is 0xF0, it is called a "System-exclusive", or "SysEx" event.

```
F0 len data F7
```

*Sequencer64* has some code in place to store these messages, but the data is currently not actually stored or used. Although there is some infrastructure to support storing the SysEx event within a sequence, the SysEx information is simply skipped. *Sequencer64* warns if the terminating 0xF7 SysEx terminator is not found at the expected length. Also, some malformed SysEx events have been encountered, and those are detected and skipped as well.

### 2.2.3.7 Sequencer Specific (0x7F)

This data, also known as SeqSpec data, provides a way to encode information that a specific sequencer application needs, while marking it so that other sequences can safely ignore the information.

FF 7F len data

In *Seq24* and *Sequencer64*, the data portion starts with four bytes that indicate the kind of data for a particular SeqSpec event:

c_midibus	^	0x24240001	Track buss number
c_midich	^	0x24240002	Track channel number
c_midiclocks	*	0x24240003	Track clocking
c_triggers	^	0x24240004	See c_triggers_new
c_notes	*	0x24240005	Song data, notes
c_timesig	^	0x24240006	Track time signature
c_bpmtag	*	0x24240007	Song beats/minute
c_triggers_new	^	0x24240008	Track trigger data
c_mutegroups	*	0x24240009	Song mute group data
c_midictrl	*	0x24240010	Song MIDI control
c_musickey	+	0x24240011	Track key (Sequencer64 only)
c_musicscale	+	0x24240012	Track scale (Sequencer64 only)
c_backsequence	+	0x24240013	Track background sequence (Sequencer64 only)

\* = global only; ^ = track only; + = both

In *Seq24*, these events are placed at the end of the song, but are not marked as SeqSpec data. Most MIDI applications handle this situation fine, but some (e.g. midicvt) do not. Therefore, *Sequencer64* makes sure to wrap each data item in the 0xFF 0x7F wrapper.

Also, the last three items above (key, scale, and background sequence) can also be stored (by *Sequencer64*) with a particular sequence/track, as well as at the end of the song. Not sure if this bit of extra flexibility is useful, but it is there.

### 2.2.3.8 Non-Specific End of Sequence

Any other statuses are deemed unsupportable in *Sequencer64*, and abort parsing with an error.

If the `-bus` option is in force, `sequence::set_midi_bus()` is called to override the buss number (if any) stored with the sequence.

Finally, `perform::add_sequence()` adds the sequence to the encoded tune.

## 2.3 SMF 0 Parsing

After parsing SMF 1 track data, we end up with a number of sequences, each on a different MIDI channel. With SMF 0, data for all channels is present in a single track. *Sequencer64* will read SMF 0 data, but we really need to be able to have one MIDI channel per track. So we need to take the data from the sequence and use it to make more sequences.

- `sequence::add_event()`.
- `sequence::set_midi_channel()`.
- `sequence::set_length()`.
- `sequence::set_midi_bus()`.
- `perform::add_sequence()`.

This code basically works. For now, please look at the source code for more details. Also, the reading of SMF 0 MIDI files is described in the *sequencer64-doc* project on GitHub.

## 2.4 Running Status

When we apply the `midicvt` application to a file saved by *Sequencer64*, we can end up with a successful ASCII conversion that ends with an error message:

```
$ midicvt hymne-seq64.midi -o hymne-seq64.asc
? Error at MIDI file offset 12155 [0x2f7b]
Error: Garbage at end 'readtrack(): unexpected running status'
```

Is this a problem in `midicvt` or *Sequencer4*? Let's learn about running status.

Running status is a way to speed up the sending of MIDI bytes to a synthesizer or sequencer by taking advantage of redundancy where possible. For example, if we're sending a consecutive group of Note On and Note Off messages to a particular channel, we can save some time by not sending the channel status byte after the first time. Here's an example with Note On on channel 1:

```
0x90 3C 7F
0x90 40 7F
0x90 43 F3
```

Since no change in status occurs after the first of these three events, we can drop the subsequent status bytes:

```
0x90 3C 7F
40 7F
43 F3
```

The 0x90 byte is saved in a "running status buffer" (RSB), and is filled in by the receiving device.

Here is the sequence of events for operating with running status.

1. Clear the RSB buffer (RSB = 0) to start.
2. If a **Voice Category Status** (VCS) byte is received, then set RSB = VCS. VCS bytes range from 0x80 to 0xEF. This is binary 1000000 to 11100000.
3. If a data byte is received (data bytes range from 0x00 to 0x7F, binary 0000000 to 0111111; that is, bit 7 is always 0 in a data byte):
  - (a) If RSB != 0, first insert the RSB into the incoming data stream, then insert the data byte.
  - (b) If RSB == 0, then just insert the data byte into the incoming data stream.
4. Clear the RSB buffer (RSB = 0) when a System Common Message (SCM) status byte is received. SCM bytes range from 0xF0 to 0xF7.
5. The message after an SCM **must** begin with a status byte. That is a byte with bit 7 set.
6. Do no special action when a Realtime Category Message (RCM) byte is received. RCM bytes range from 0xF8 to 0xFF.

Note that some events, such as Tempo, assume that its bytes are all data bytes.



## Chapter 3

# JACK, Live, and Song Modes in Sequencer64

**Author(s)** Chris Ahlstrom 2017-04-01

### 3.1 Introduction

This section describes the interactions between JACK settings and the Live/Song Mode settings, with an eye to describing the proper behavior of Sequencer64 with JACK settings, how the Live/Song modes are supposed to work, and what bugs or issues remain in Sequencer64's JACK handling.

Note: There is currently no description of the 0.90.x line's native JACK MIDI support, except what is found in the developer's reference manual and in the updated sequencer64-doc project at GitHub.

I'm not sure why Doxygen is applying the "code" font so often here. Weird, annoying.

### 3.2 JACK Functions

Please study the following URL and note these important points:

<http://jackaudio.org/files/docs/html/transport-design.html>

- The timebase master continuously updates position information, beats, timecode, etc. There is at most one master active at a time. If no client is registered as timebase master, frame numbers will be the only position information available.
- The timebase master registers a callback that updates position information while transport is rolling. Its output affects the following process cycle. This function is called immediately after the process callback in the same thread whenever the transport is rolling, or when any client has set a new position in the previous cycle.
- Clients that don't declare a sync callback are assumed ready immediately, anytime the transport wants to start. If a client doesn't require slow-sync processing, it can set its sync callback to NULL.
- The transport state is always valid; initially it is JackTransportStopped.
- When someone calls `jack_transport_start()`, the engine resets the poll bits and changes to a new state, JackTransportStarting.
- When all slow-sync clients are ready, the state changes to JackTransportRolling.

Does Sequencer64 need a latency callback?

[http://jackaudio.org/files/docs/html/group\\_\\_ClientCallbacks.html](http://jackaudio.org/files/docs/html/group__ClientCallbacks.html)

(We need to see why most of the following is in a monospaced font. Is there a new Doxygen feature?)

Here are summaries of the JACK functions used in the `jack_assistant` module:

### 3.2.1 jack\_client\_open()

Open a client session with a JACK server. More complex and powerful than `jack_client_new()`. Clients choose which of several servers to connect, and how to start the server automatically, if not already running. There is also an option for JACK to generate a unique client name.

```
const char *    client_name,
jack_options_t  options,
jack_status_t * status,
...
```

`client_name` of at most `jack_client_name_size()` characters. 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.

`options` formed by OR-ing together `JackOptions` bits. Only the `JackOpenOptions` bits are allowed.

`status` (if non-NULL) an address for JACK to return information from the open operation. This status word is formed by OR-ing together the relevant `JackStatus` bits.

Optional parameters: depending on corresponding [options bits] additional parameters may follow `status` (in this order).

[`JackServerName`] (`char *`) `server_name` selects from among several possible concurrent server instances. Server names are unique to each user. If unspecified, use "default" unless `$JACK_DEFAULT_SERVER` is defined in the process environment.

Returns:

Opaque client handle if successful. If this is NULL, the open operation failed, and `*status` includes `JackFailure`, and the caller is not a JACK client.

### 3.2.2 jack\_on\_shutdown()

Registers a function to call when the JACK server shuts down the client thread. It must be an asynchronous POSIX signal handler: only async-safe functions, executed from another thread. A typical function might set a flag or write to a pipe so that the rest of the application knows that the JACK client thread has shut down. Clients do not need to call this function. It only helps clients understand what is going on. It should be called before `jack_client_activate()`.

### 3.2.3 jack\_set\_sync\_callback()

Register/unregister as a slow-sync client; it can't respond immediately to transport position changes. The callback is run at the first opportunity after registration: if the client is active, this is the next process cycle, otherwise it is the first cycle after `jack_activate()`. After that, it runs as per `JackSyncCallback` rules. Clients that don't set this callback are assumed ready immediately any time the transport wants to start.

### 3.2.4 jack\_set\_process\_callback()

Tells the JACK server to call the callback whenever there is work. The function must be suitable for real-time execution, it cannot call functions that might block for a long time: `malloc()`, `free()`, `printf()`, `pthread_mutex_lock()`, `sleep()`, `wait()`, `poll()`, `select()`, `pthread_join()`, `pthread_cond_wait()`, etc. In the current class, this function is a do-nothing function.

### 3.2.5 jack\_set\_session\_callback()

Tells the JACK server to call the callback when a session event is delivered. Setting more than one session callback per process is probably a design error. For a multiclient application, it's more sensible to create a JACK client with only one session callback.

### 3.2.6 jack\_activate()

Tells the JACK server that the application is ready to start processing.

### 3.2.7 jack\_release\_timebase()

TODO

### 3.2.8 jack\_client\_close()

TODO

### 3.2.9 jack\_transport\_start()

Starts the JACK transport rolling. Any client can make this request at any time. It takes effect no sooner than the next process cycle, perhaps later if there are slow-sync clients. This function is realtime-safe. No return code.

### 3.2.10 jack\_transport\_stop()

Starts the JACK transport rolling. Any client can make this request at any time. This function is realtime-safe. No return code.

### 3.2.11 jack\_transport\_locate()

Repositions the transport to a new frame number. May be called at any time by any client. The 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.

### 3.2.12 jack\_transport\_reposition()

Request a new transport position. May be called at any time by any client. The 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. This call, made in the position() function, is currently disabled.

### 3.2.13 jack\_transport\_query()

Query the current transport state and position. This function is realtime-safe, and can be called from any thread. If called from the process thread, pos corresponds to the first frame of the current cycle and the state returned is valid for the entire cycle.

The first parameter is the client, which is a pointer to the JACK client structure.

The second parameter is a pointer to structure for returning current transport position; pos->valid will show which fields contain valid data. If pos is NULL, do not return position information.

This function returns the current transport state.

## 3.3 Modes Operation

### 3.3.1 No JACK, Live Mode

In `~/ .config/sequencer64/sequencer64.rc`, set:

- `jack_transport = 0`
- `jack_master = 0`
- `jack_master_cond = 0`
- `song_start_mode = 0`

By changing the start mode to 0 (false), Sequencer64 is put into Live Mode. With this setting, control of the muting and unmuting of patterns resides in the main window (the patterns window). One can start the playback in the performance (song) window, but it will not affect which patterns play, at all.

Note that this option is part of the *File / Options / JACK/LASH* configuration page.

### 3.3.2 No JACK, Song Mode

In `~/ .config/sequencer64/sequencer64.rc`, set:

- `jack_transport = 0`
- `jack_master = 0`
- `jack_master_cond = 0`
- `song_start_mode = 1`

By changing the start mode to 1 (true), Sequencer64 is put into Song Mode.

With this setting, control of the muting and unmuting of patterns resides in the song window (the performance window). The patterns shown in the pattern slots of the main window turn on and off whenever the progress bar is in the pattern as drawn in the performance window.

Note that this option is part of the *File / Options / JACK/LASH* configuration page.



### 3.3.3 JACK Transport

In `~/ .config/sequencer64/sequencer64.rc`, set:

- `jack_transport = 1`
- `jack_master = 0`
- `jack_master_cond = 0`
- `song_start_mode = 0` or `1` (see previous section)

The current behavior is that `qjackctl` and `sequencer64` playback/progress seem to be independent of each other.

The workaround seems to be to set `seq24/sequencer64` as JACK Master, or if another *application* (e.g. `Qtractor`) is JACK Master.

#### OLD BEHAVIOR:

Start `qjackctl`, verify that it sets up correctly, then click it's "play" button to start the transport rolling. Run `sequencer64`, load a file. Then note that starting playback (whether in the main window or in the performance window) is ineffective, but resets the time counter in `qjackctl`. Why? With JACK sync enabled by the macro:

```
[JACK transport slave]
jack sync(): zero frame rate [single report]!?
[JackTransportRolling]
[JackTransportStarting] (every time space bar pressed)
[Start playback]
. . .
```

END OF OLD BEHAVIOR.

## 3.4 Breakage

Old message about `seq24` being broken:

<http://lists.linuxaudio.org/pipermail/linux-audio-user/2010-November/073848.html>

```
i dont see the transport synchronisation working with a jack1 svn version.
you are still using only a sync callback.
```

```
and you are relying on the transport to go through the
JackTransportStarting state.
```

```
this issue should be fixed.
iirc we came to the conclusion, that seq24 is broken, and we will not
revert the changes in jack, which break it.
```

```
the quick and dirty fix on your side, would be to register an empty
process_callback.
```

```
but the issue still remains. seq24 is NOT a slow sync client. but it
registers a sync_callback.
and it even takes a lock in the sync callback.
```

```
the patch for jack-session support didnt get merged either.
```

Another one (no need for a URL):

```
I use seq24 for the majority of my projects but it isn't ideal (I should
point out that I never finish anything). I don't like seq24's pianoroll
editor, the way you do CC envelopes isn't ideal, it uses alsa-midi, there's
unnecessary complexity in switching from pattern-trigger mode to song mode,
and its insistence on being transport master while not even being able to
adjust tempo when live is annoying
```

### 3.5 JACK References

- <http://libremusicproduction.com/articles/demystifying-jack-%E2%80%93-beginners-guide>
- <http://jackaudio.org/files/docs/html/transport-design.html>
- <http://kxstudio.linuxaudio.org/Repositories>

## Chapter 4

# User Testing of Sequencer64 with Yoshimi

**Author(s)** Chris Ahlstrom 2016-03-04

### 4.1 Introduction

This section describes user testing of Sequencer64 using Yoshimi. It will expand as we work our way through all the many use-cases that can be achieved with Sequencer64 and Yoshimi.

Please note that the most advanced and recent testing can be found currently in the document `contrib/notes/jack-testing.txt`. We will eventually merge the final tests here... someday.

### 4.2 Smoke Test

Every so often we run Sequencer64 with a software synthesizer to make sure we haven't broken any functionality via our major refactoring efforts. We call it a "smoke test". We fire up the two application, and see if anything smokes.

This smoke test sets up Yoshimi with a very simple ALSA setup, and no instruments are loaded. Instead, only the "Simple Sound" is used on all channels. We've been doing this test with Yoshimi 1.3.6. The current Debian Sid ("testing") version of Yoshimi is 1.3.6-2, pulled from SourceForge. It seems to have issues, so we've been cloning and pulling the code from:

```
https://github.com/Yoshimi/yoshimi.git
```

After getting the application build and installed, the next step is to run it, using ALSA for MIDI and for audio:

```
$ yoshimi -a -A &
```

Next, fix up the configuration files for Sequencer64, `~/.config/sequencer64/sequencer64.rc` and `~/.config/sequencer64/sequencer64.usr`.

First hide `sequencer64.usr` somewhere, or delete it, as it will determine what MIDI devices are available, and we don't want that (yet). Second, make sure that `sequencer64.rc` makes the following setting:

```
[manual-alsa-ports]

# Set to 1 if you want sequencer64 to create its own ALSA ports and
# not connect to other clients

0 # number of manual ALSA ports
```

Next, run the newly-built version of Sequencer64. If desired, use the `--bus` option described below to force the buss number to the buss you need, as shown in the second version of the command:

```
$ sequencer64/sequencer64 &
$ sequencer64/sequencer64 --bus 5 &
```

In *File / Options / MIDI Clock*, observe the MIDI inputs made available by your system. Our system shows:

```
[0] 14:0 (Midi Through Port-0)
[1] 128:0 (TiMidity port 0)
[2] 128:0 (TiMidity port 1)
[3] 128:0 (TiMidity port 2)
[4] 128:0 (TiMidity port 3)
[5] 129:0 (input)
```

For some reason (a bug in Yoshimi?), input "[5]" doesn't indicate that it is Yoshimi, but it is. Take note of that input number... that is the MIDI buss number that is needed to drive Yoshimi.

Also make sure that of the clock settings for those busses are "Off".

**The next instruction still works, but it is easier to simply pass the option `--bus 5` to Sequencer64 when starting it up.**

Now open the file `sequencer64/contrib/midi/b4uacuse-GM-format.midi` in Sequencer64. For all of the patterns (slots) that have lots of data in them, right click on the pattern and select *Midi Bus / [5] 129:0 (input)* and the desired channel number. (Doesn't matter much, just use up the lower channel numbers first).

Back in Yoshimi, select each Part corresponding to the channels you selected. Make sure *Enabled* is checked for each desired channel.

Back in Sequencer64, click on each pattern you want to hear, which highlights them in black. Now click the play button (green triangle). The song should play, with each part using the "Simple Sound". Not too bad for a bunch of sine waves, eh?

Now we can test the application more fully. Note that the instructions here are very light. Detailed instructions on the usage of Sequencer64 can be found in the following project, which contains a PDF file and the LaTeX code used to build it:

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

Although it applies to an earlier version of the project, it still mostly holds true for Sequencer64.

## 4.3 Tests in the Patterns Window

The Patterns window is the inside portion of the main window, supported by the `mainwid` class. It contains a grid of boxes or slots, with each slot potentially containing a pattern, sequence, or track. Empty tracks (i.e. tracks that contain no events, like title-only tracks) are highlighted in yellow.

This window supports only a single variant of mouse-handling.

### 4.3.1 Button Clicks on a Pattern

A left-click on a pattern slot should cause the following to happen:

1. The pattern will be highlighted (white on a black background). This won't occur until the button is released.
2. During playback, the pattern will emit MIDI events and play its sequence.
3. If the pattern is dragged to another slot, whether playing is in progress or not, releasing the button in the destination slot will move the pattern to that slot.

A right-click on a pattern slot should cause the following to happen:

1. If the pattern is empty, then a pop-up menu to make a New pattern, paste a pattern, or make other selections will appear.
2. If the pattern is active, then a pop-up menu to Edit the pattern or make other selections will appear.
3. A second right-click, just off the menu, will dismiss the menu.

### 4.3.2 Patterns Window Key Shortcuts

First, note the selection of the File / Options / Keyboard / Show keys option. The tests here should work whether or not it is selected. The only difference is if the keys are shown.

We got a segfault during this test, when we weren't being systematic about it.

### 4.3.3 The Sequencer64 User File

To be discussed.

## 4.4 Tests Using Valgrind

Valgrind is a very useful tool for unearthing memory issues and other issues in an application, especially when one has the source code and can build the code with debugging information.

One runs the application from the command line, preceding its command line with valgrind and some of its options.

#### 4.4.1 Valgrind Suppressions

One problem with valgrind is that it also uncovers errors in system libraries that one has no control over. These errors clutter the output, so we suppress them using a valgrind "suppressions" file. Here's how to create one:

```
$ valgrind --gen-suppressions=yes --log-file=val.supp ./Sequencer64/sequencer64
$ valgrind --gen-suppressions=all --log-file=val.supp ./Sequencer64/sequencer64
```

As the program runs, one is asked to print a suppression. If the error is due to a system or third-party library, answer "Y return", and then copy-and-paste the suppression to a file, giving it a name. For example, we provide a file `contrib/seq64.supp` containing suppressions of errors that annoy us. There are way too many "errors" in ALSA, GTK+, gtkmm, glibc, and more.

The second command collects all the suppressions. Passing the `val.supp` file through `sed` makes it immediately usable:

```
$ sed -i -e /^==/g val.supp
```

Running valgrind like this then shows mostly the errors we care about:

```
$ valgrind --suppressions=val.supp ./Sequencer64/sequencer64
```

We've added some other suppression files to the `contrib` directory. Too much! For example:

<https://github.com/dtrebbien/GNOME.supp>

However, overall this process is very painful, and we're going to eventually do all the valgrind work on the unit-test project for Sequencer64:

<https://github.com/ahlstromcj/seq64-tests>

#### 4.4.2 Full Valgrind Leak-Checking

Here's how to capture errors, while suppressing the system errors and while generating a log file:

```
$ valgrind --suppressions=contrib/seq64.supp --leak-check=full \
  --track-origins=yes --log-file=valgrind.log --show-leak-kinds=all \
  ./Sequencer64/sequencer64
```

The errors can be also be re-routed to a log-file via the `"2> valgrind.log"` shell redirection.

Another idea is to precede the valgrind command with the following construct:

```
$ G_SLICE=debug-blocks valgrind ...
```

`G_SLICE=debug-blocks` will turn off gtk's advanced memory management to allow valgrind to show correct results. This results in an amazing plethora of invalid read and invalid write errors in GNOME-related libraries. Sheesh!

And don't forget about Valgrind's "massif" memory-tracking tool! (More to come!)

#### 4.4.2.1 Leak-Checking Basic Operation

For the first pass, just run Sequencer64, then immediately exit. Then scan the log file to see if any "errors" can be pinpointed to the application and library code.

Don't forget to run the same scenario without valgrind, in a console window, to see if any of our own debug/problem output occurs.

In any case, leakage tagged as "still reachable" isn't as bad as leakage tagged as "definitely lost" or "indirectly lost".

But good luck finding a Sequencer64 bug buried in the chaff of 3rd-party valgrind reports, even with some suppressions enabled. Apparently a lot of them have to do with data structures that are intended to last the full life of the application.

One can make the search a little easier by searching for the "seq64" namespace in the valgrind log.

## 4.5 Specific Fault Debugging

This section goes through specific debugging cases we encountered. They should be part of the regular testing of Sequencer64.

## 4.6 Snipping of a MIDI file.

In order to have a test file for the *seq64-tests* project, we loaded up the *b4uacuse-GM-format.midi* file, removed all but four of the tracks, and saved it as *b4uacuse-snipped.midi*. Loading this file into Sequencer64 caused the following:

```
$ ./Sequencer64/sequencer64
[Reading user configuration /home/ahlstrom/.config/sequencer64/sequencer64 usr]
[Reading rc configuration /home/ahlstrom/.config/sequencer64/sequencer64.rc]
get_sequence(): m_seqs[4] not null
Segmentation fault
```

First step, fire up a debugger and see what happened. We use *cgdb*, a text-based front-end for gdb with a "vi" feel.

```
$ cgdb ./Sequencer64/sequencer64
```

Just hit "r", do *File / Open*, navigate to *b4uacuse-snipped.midi*, select it, and watch what happens.

The "bt" (backtrace) command shows a pretty large stack, 52 items. Page up to the top of the stack, and select frame 1 ("fr 1"). This shows a mutex at a very low address, 0x650! Frame 2 shows we are in the automutex constructor, calling lock() on that same badly-located mutex. Frame 3 is in *sequence::event\_count()*, same bad mutex, and the *m\_events* member is at address 0x0. Obviously, we're dealing with an unallocated sequence.

Frame 4 is in *mainwid::draw\_sequence\_on\_pixmap()*, just after we've retrieved the next sequence via *perform\_get\_sequence(4)*. But that would be the fifth sequence (the sequence numbers start at 0), and we snipped all but 4 from the file before we saved it.

So, one thing we need to do is *check* the value returned by *get\_sequence()* before we try to use it. The other thing to do is figure out how we got to the fifth sequence, and fix that code as well. Using the command "*p perf().sequence\_count()*", we verify that there are indeed only 4 sequences allocated.

Frame 5 is in `mainwid::draw_sequences_on_pixmap()`. That function tries to load all sequences on the current screen-set, from 0 to 31, without checking to see how many there actually are. Inefficient and dangerous.

Frame 6 is in `mainwid::reset()`. We could pass `perf().sequence_count()` here for checking, or get it in `mainwid::draw_sequences_on_pixmap()`.

Before we fix this issue, we need to load a file that works, to see why it does not fail for most files. We will put a breakpoint at the top `mainwid::draw_sequences_on_pixmap()`.

We hit the breakpoint before even loading a file, with a `sequence_count()` of 0. The call to `valid_sequence(0)` passes the test. We may want to make `valid_sequence()` take the `sequence_count()` into account. But the call to `perf().is_active(0)` prevents anything bad from happening at startup time.

Once we load a good file, the `sequence_count()` is 14 in `mainwid::draw_sequences_on_pixmap()`. We turn on the display of "offset" using the command "display offset", and "c" (for "continue") until `offset = 14`, which means we are beyond that last sequence. That bad access is prevented by `perf().is_active(14)`.

So the fundamental problem is that `perf().is_active(4)` is not protecting the access when we load the "bad file". We need to find and fix that issue before papering over the problem with better access checks.

Start again, putting a breakpoint in the call to "new sequence(m\_ppqn)" in `midifile`. This call sets up some members and clears the list of 256 playing notes. Add another breakpoint at "a\_perf.add\_sequence()" to see what's happening there.

What we find is that the first two tracks have proper sequence numbers as read from the MIDI file, 0 and 1. But the third one preserves the number from the old file, 4. We have a disjunction between the track number and the sequence number, a conceptual problem. We can leave it as is, and beef up the error-checking, or replace the sequence number with the track number when loading the file. What to do?

- Make sure that the is-active flag for all sequences is "false", that the pointers are always null, and make sure to test both of these items (depending on context) before doing anything with the sequence.
- Convert the sequence number to the track number upon saving the MIDI file, or upon reading the MIDI file, and use that number when adding the sequence to the perform object. This might affect some seq24/sequencer64 functionality, however. It's big move.

We need information on reading and importing.

First, if we look at a file that we created long ago by importing `b4uacuse.mid`, `b4uacuse-GM-format.mid`, it has its fourteen sequence numbers identical to their track numbers. No problem.

Second, if we just read `b4uacuse.mid`, a non-seq24-created MIDI file, we see that each of its tracks have no sequence number – they are all zero. The `perform::add_sequence()` simply iterates from the beginning of `m_seqs[]` until it finds an inactive `m_seqs[i]`, and uses that element to hold the sequence pointer.

But now it also segfaults! Let's fix all the non-checked `get_sequence()` calls right away, it is too big an issue to ignore.

In the end, we have to be aware that a screen-set can have blank (null) slots interspersed amongst the active slots.



## Chapter 5

# Speed Issue of Sequencer64

**Author(s)** Chris Ahlstrom 2017-03-27

### 5.1 Introduction

This section describes some speed issues of Sequencer64. Early on in our reboot of *seq24*, we noticed that some of our larger files took a noticeable time to load. It was only a few seconds, but seemed like a long time for such small files. We fixed the issue using an alternate container implementation.

We recently added a large MIDI file to test, `b4uacuse-stress.midi`, and all hell broke loose.

### 5.2 Initial Change of Containers

In the beginning, we noticed that the MIDI container implementation used the `std::list` container, and also that it called `std::list::sort()` after each event was added to the container.

Our first thought was to replace the `std::list` with an `std::multimap`. Insertions into this container are made in the appropriate location (rather than at the end), and so are automatically sorted. We kept the old code around, but enabled the new multimap code via the `SEQ64_USE_EVENT_MAP` macro. This decreased the time of loading.

(It also exposed a small number of bugs that users of *Sequencer64* discovered and fixed.)

At the back of our minds was the possibility that the longer time needed to increment a multimap iterator versus a list iterator would prove to limit the amount of data that could be played back. Once we finally created a large file, `b4uacuse-stress.midi`, a 1.5 Mb file, we experienced the limitations of that iterator during playback. On our main development laptop, a near-gaming Intel i7 machine, there were minor artifacts in playback. On our old single-core laptop with 32-bit Debian installed, the sequence would not play, and would continually and visibly refresh the main window display.

### 5.3 Back to the Original Container

So then we re-enabled the old `seq24` list implementation, and found that the time needed to load `b4uacuse-stress.midi` was over 6 minutes on our near-gamer laptop and 13 minutes on the single-core laptop.

So, we had to find a way to get the fast loading speed of the `std::multimap` and the faster speed of the `std::list` iterator. The obvious way was to go back to the `std::list` container and stop sorting the container after every insert, when loading the MIDI file.

This worked, but had some side-effects that had to be fixed. We found that the `sequence::verify_and_link` function required that the container be sorted first, and so we had to find the places where that function was called, and make sure that the sorting had occurred.

Anyway, the current configuration for the usage of `std::multimap` versus `std::list` and the sorting of the MIDI container after every event insertion versus after all the events are now permanent and hardwired. The default is to not use the event multimap, and to not presort events. This makes loading fast, and playback able to handle more sequences. One can also try to use the multimap, or use the list with pre-sorting, if bugs appear when building the application with the default setting. However, we really want to get the post-sorted list implementation to work, to get fast loading speed and higher throughput at the same time.

The other options are available as a fallback in case one gets struck by bugs in the default, and can afford slower loads or less throughput.

### 5.4 What is Next, the Vector?

Or, what's your vector, Victor?

For playback, the `std::vector` iterator can be even faster than the `std::list` iterator, because the vector does not use stored pointers to the next element, and, since its storage layout is essentially like a standard C array, processor caching can add even more to the speed of access.

However, this change will require carefully analysis and even more careful work to correctly implement the change. We will log this as a future improvement.

Actually, now we have reverted back to the `std::list` implementation, with the key difference that, when loading a file, we do not sort until we have read all of the events. So we get a fast load time and higher maximum throughput during playback.

Now to test the hell out of the next version!

## Chapter 6

# Licenses

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

**Author(s)** Chris Ahlstrom 2015-09-10

### 6.1 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.

### 6.2 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>

## 6.3 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>

## 6.4 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>

## 6.5 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.

## 6.6 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*).



## Chapter 7

# Todo List

### File `calculations.cpp`

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

### 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 `perroll`?

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

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

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

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

### Global `seq64::mainwid::timeout ()`

We should use this callback to display the current time in the playback.

### Global `seq64::mainwnd::mainwnd (perform &p, bool allowperf2=true, int ppqn=SEQ64_USE_DEFAULT_PPQN)`

Offload most of the work into an initialization function like `options` does; make the `perform` parameter a reference; valgrind flags `m_tooltips` as lost data, but if we try to manage it ourselves, many more leaks occur.

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

Test this functionality in old and new application.

### Global `seq64::midi_jack::api_sysex (event *e24)`

Flesh out this routine.

### Global `seq64::perfedit::perfedit (perform &p, bool second_perfedit=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::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 `PPQN * 4`.

**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::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::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.

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 8

# 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.

**Global [seq64::sequence::get\\_name](#) () const**



## Chapter 9

# Namespace Index

### 9.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">Gtk</a> . . . . .	<a href="#">49</a>
<a href="#">seq64</a> Define this macro to use the new seq24 v . . . . .	<a href="#">49</a>



## Chapter 10

# Hierarchical Index

### 10.1 Class Hierarchy

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

seq64::AbstractPerfInput . . . . .	133
seq64::FruityPerfInput . . . . .	263
seq64::Seq24PerfInput . . . . .	852
seq64::automutex . . . . .	137
seq64::busarray . . . . .	138
seq64::businfo . . . . .	148
seq64::click . . . . .	154
seq64::configfile . . . . .	161
seq64::optionsfile . . . . .	646
seq64::userfile . . . . .	1119
Dialog	
seq64::options . . . . .	639
DrawingArea	
seq64::gui_palette_gtk2 . . . . .	298
seq64::gui_drawingarea_gtk2 . . . . .	280
seq64::eventslots . . . . .	239
seq64::maintime . . . . .	395
seq64::mainwid . . . . .	400
seq64::perfnames . . . . .	668
seq64::perfroll . . . . .	764
seq64::perftime . . . . .	784
seq64::seqdata . . . . .	858
seq64::seqevent . . . . .	904
seq64::seqkeys . . . . .	918
seq64::seqroll . . . . .	943
seq64::seqtime . . . . .	976
seq64::editable_events . . . . .	179
Entry	
seq64::keybindentry . . . . .	341
seq64::event . . . . .	188
seq64::editable_event . . . . .	166
seq64::event_list::event_key . . . . .	209
seq64::event_list . . . . .	211
exception	

seq64::rterror . . . . .	819
seq64::font . . . . .	258
seq64::FruitySeqEventInput . . . . .	269
seq64::FruitySeqRollInput . . . . .	272
seq64::gui_assistant . . . . .	274
seq64::gui_assistant_gtk2 . . . . .	277
seq64::jack_assistant . . . . .	314
seq64::jack_scratchpad . . . . .	338
seq64::jack_status_pair_t . . . . .	340
seq64::keys_perform . . . . .	344
seq64::keys_perform_gtk2 . . . . .	374
seq64::keys_perform_transfer . . . . .	377
seq64::keystroke . . . . .	381
seq64::lash . . . . .	386
seq64::mastermidibase . . . . .	435
seq64::mastermidibus . . . . .	456
seq64::mastermidibus . . . . .	456
seq64::midi_container . . . . .	492
seq64::midi_list . . . . .	546
seq64::midi_vector . . . . .	578
seq64::midi_control . . . . .	501
seq64::midi_info . . . . .	512
seq64::midi_alsa_info . . . . .	476
seq64::midi_jack_info . . . . .	539
seq64::midi_jack_data . . . . .	536
seq64::midi_measures . . . . .	550
seq64::midi_message . . . . .	553
seq64::midi_port_info . . . . .	561
seq64::midi_queue . . . . .	565
seq64::midi_splitter . . . . .	568
seq64::midi_timing . . . . .	574
seq64::midibase . . . . .	582
seq64::midi_api . . . . .	482
seq64::midi_alsa . . . . .	466
seq64::midi_in_alsa . . . . .	506
seq64::midi_out_alsa . . . . .	556
seq64::midi_jack . . . . .	523
seq64::midi_in_jack . . . . .	508
seq64::midi_out_jack . . . . .	558
seq64::rtmidi . . . . .	822
seq64::rtmidi_in . . . . .	830
seq64::rtmidi_out . . . . .	850
seq64::midibus . . . . .	604
seq64::midibus . . . . .	604
seq64::midifile . . . . .	616
seq64::mutex . . . . .	637
seq64::condition_var . . . . .	159
seq64::editable_event::name_value_t . . . . .	638
ObjectBase	
seq64::seqmenu . . . . .	930
seq64::mainwid . . . . .	400
seq64::perfnames . . . . .	668
seq64::perform . . . . .	676
seq64::performcallback . . . . .	762
seq64::mainwnd . . . . .	415



seq64::midi_port_info::port_info_t . . . . .	793
seq64::rc_settings . . . . .	795
seq64::rect . . . . .	817
seq64::gui_drawingarea_gtk2::rect . . . . .	818
seq64::rtmidi_in_data . . . . .	834
seq64::rtmidi_info . . . . .	839
seq64::Seq24SeqEventInput . . . . .	856
seq64::sequence . . . . .	982
seq64::trigger . . . . .	1053
seq64::triggers . . . . .	1057
seq64::user_instrument . . . . .	1074
seq64::user_instrument_t . . . . .	1078
seq64::user_midi_bus . . . . .	1079
seq64::user_midi_bus_t . . . . .	1084
seq64::user_settings . . . . .	1084
Window	
seq64::gui_window_gtk2 . . . . .	310
seq64::eventedit . . . . .	223
seq64::lfownd . . . . .	390
seq64::mainwnd . . . . .	415
seq64::perfedit . . . . .	650
seq64::seqedit . . . . .	871



## Chapter 11

# Data Structure Index

### 11.1 Data Structures

Here are the data structures with brief descriptions:

<a href="#">seq64::AbstractPerfInput</a>	Provides an abstract base class to provide the minimal interface for the various "perf input" classes . . . . .	133
<a href="#">seq64::automutex</a>	Provides a mutex that locks automatically when created, and unlocks when destroyed . . . . .	137
<a href="#">seq64::busarray</a>	Holds a number of businfo objects . . . . .	138
<a href="#">seq64::businfo</a>	A new class to consolidate a number of bus-related arrays into one array . . . . .	148
<a href="#">seq64::click</a>	Encapsulates any possible mouse click . . . . .	154
<a href="#">seq64::condition_var</a>	A mutex works best in conjunction with a condition variable . . . . .	159
<a href="#">seq64::configfile</a>	This class is the abstract base class for optionsfile and userfile . . . . .	161
<a href="#">seq64::editable_event</a>	Provides for the management of MIDI editable events . . . . .	166
<a href="#">seq64::editable_events</a>	Provides for the management of an ordered collection MIDI editable events . . . . .	179
<a href="#">seq64::event</a>	Provides events for management of MIDI events . . . . .	188
<a href="#">seq64::event_list::event_key</a>	Provides a key value for an event map . . . . .	209
<a href="#">seq64::event_list</a>	Receptable for MIDI events . . . . .	211
<a href="#">seq64::eventedit</a>	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 . . . . .	223
<a href="#">seq64::eventslots</a>	This class implements the left-side list of events in the pattern event-edit window . . . . .	239
<a href="#">seq64::font</a>	This class provides a wrapper for rendering fonts that are encoded as a 16 x 16 pixmap file in XPM format . . . . .	258
<a href="#">seq64::FruityPerfInput</a>	Implements the performance input of that certain fruity sequencer that people seem to like . . .	263

<a href="#">seq64::FruitySeqEventInput</a>	
This structure implements the interaction methods for the "fruity" mode of operation	269
<a href="#">seq64::FruitySeqRollInput</a>	
Implements the fruity mouse interaction paradigm for the seqroll	272
<a href="#">seq64::gui_assistant</a>	
This class provides an interface for some of the GUI support needed in Sequencer64	274
<a href="#">seq64::gui_assistant_gtk2</a>	
This class provides an interface for some of the Gtk/Gdk/Glib support needed in Sequencer64	277
<a href="#">seq64::gui_drawingarea_gtk2</a>	
Implements the basic drawing areas of the application	280
<a href="#">seq64::gui_palette_gtk2</a>	
Implements a stock palette of Gdk::Color elements	298
<a href="#">seq64::gui_window_gtk2</a>	
This class supports a basic interface for Gtk::Window-derived objects	310
<a href="#">seq64::jack_assistant</a>	
This class provides the performance mode JACK support	314
<a href="#">seq64::jack_scratchpad</a>	
Provide a temporary structure for passing data and results between a perform and <a href="#">jack_assistant</a> object	338
<a href="#">seq64::jack_status_pair_t</a>	
Provides an internal type to make it easier to display a specific and accurate human-readable message when a JACK operation fails	340
<a href="#">seq64::keybindentry</a>	
Class for management of application key-bindings	341
<a href="#">seq64::keys_perform</a>	
This class supports the performance mode	344
<a href="#">seq64::keys_perform_gtk2</a>	
This class supports the performance mode	374
<a href="#">seq64::keys_perform_transfer</a>	
Provides a data-transfer structure to make it easier to fill in a <a href="#">keys_perform</a> object's members using sscanf()	377
<a href="#">seq64::keystroke</a>	
Encapsulates any practical keystroke	381
<a href="#">seq64::lash</a>	
This class supports LASH operations, if compiled with LASH support (i.e.	386
<a href="#">seq64::lfownd</a>	
One LFO window class	390
<a href="#">seq64::maintime</a>	
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	395
<a href="#">seq64::mainwid</a>	
This class implements the piano roll area of the application	400
<a href="#">seq64::mainwnd</a>	
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	415
<a href="#">seq64::mastermidibase</a>	
The class that "supervises" all of the midibus objects?	435
<a href="#">seq64::mastermidibus</a>	
The class that "supervises" all of the midibus objects	456
<a href="#">seq64::midi_alsa</a>	
This class implements the ALSA version of the <a href="#">midi_api</a>	466
<a href="#">seq64::midi_alsa_info</a>	
The class for handling ALSA MIDI input	476
<a href="#">seq64::midi_api</a>	
Subclasses of midi_in_api and midi_out_api contain all API- and OS-specific code necessary to fully implement the rtmidi API	482
<a href="#">seq64::midi_container</a>	
This class is the abstract base class for a container of MIDI track information	492

<a href="#">seq64::midi_control</a>	This class (formerly a struct) contains the control information for sequences that make up a live set . . . . .	501
<a href="#">seq64::midi_in_alsa</a>	This class implements the ALSA version of a MIDI input object . . . . .	506
<a href="#">seq64::midi_in_jack</a>	The class for handling JACK MIDI input . . . . .	508
<a href="#">seq64::midi_info</a>	The class for holding basic information on the MIDI input and output ports currently present in the system . . . . .	512
<a href="#">seq64::midi_jack</a>	This class implements with JACK version of the <a href="#">midi_alsa</a> object . . . . .	523
<a href="#">seq64::midi_jack_data</a>	Contains the JACK MIDI API data as a kind of scratchpad for this object . . . . .	536
<a href="#">seq64::midi_jack_info</a>	The class for handling JACK MIDI port enumeration . . . . .	539
<a href="#">seq64::midi_list</a>	This class is the std::list implementation of the <a href="#">midi_container</a> . . . . .	546
<a href="#">seq64::midi_measures</a>	Provides a data structure to hold the numeric equivalent of the measures string "measures↔:beats:divisions" ("m:b:d") . . . . .	550
<a href="#">seq64::midi_message</a>	Provides a handy capsule for a MIDI message, based on the std::vector<unsigned char> data type from the RtMidi project . . . . .	553
<a href="#">seq64::midi_out_alsa</a>	This class implements the ALSA version of a MIDI output object . . . . .	556
<a href="#">seq64::midi_out_jack</a>	The JACK MIDI output API class . . . . .	558
<a href="#">seq64::midi_port_info</a>	A class for holding port information . . . . .	561
<a href="#">seq64::midi_queue</a>	Provides a queue of <a href="#">midi_message</a> structures . . . . .	565
<a href="#">seq64::midi_splitter</a>	This class handles the parsing and writing of MIDI files . . . . .	568
<a href="#">seq64::midi_timing</a>	We anticipate the need to have a small structure holding the parameters needed to calculate MIDI times within an arbitrary song . . . . .	574
<a href="#">seq64::midi_vector</a>	This class is the std::vector implementation of the <a href="#">midi_container</a> . . . . .	578
<a href="#">seq64::midibase</a>	This class implements with ALSA version of the midibase object . . . . .	582
<a href="#">seq64::midibus</a>	This class implements with ALSA version of the midibus object . . . . .	604
<a href="#">seq64::midifile</a>	This class handles the parsing and writing of MIDI files . . . . .	616
<a href="#">seq64::mutex</a>	Simple wrapper for the pthread_mutex_t type used as a recursive mutex . . . . .	637
<a href="#">seq64::editable_event::name_value_t</a>	Provides a type that contains the pair of values needed for the various lookup maps that are needed to manage editable events . . . . .	638
<a href="#">seq64::options</a>	This class supports a full tabbed options dialog . . . . .	639
<a href="#">seq64::optionsfile</a>	Provides a file for reading and writing the application' main configuration file . . . . .	646
<a href="#">seq64::perfedit</a>	This class supports a Performance Editor that is used to arrange the patterns/sequences defined in the patterns panel . . . . .	650

<a href="#">seq64::perfnames</a>	This class implements the left-side keyboard in the patterns window . . . . .	668
<a href="#">seq64::perform</a>	This class supports the performance mode . . . . .	676
<a href="#">seq64::performcallback</a>	Provides for notification of events . . . . .	762
<a href="#">seq64::perftroll</a>	This class implements the performance roll user interface . . . . .	764
<a href="#">seq64::perftime</a>	This class implements drawing the piano time at the top of the "performance window" (the "song editor") . . . . .	784
<a href="#">seq64::midi_port_info::port_info_t</a>	Hold the information for a single port . . . . .	793
<a href="#">seq64::rc_settings</a>	This class contains the options formerly named "global_XXXXXX" . . . . .	795
<a href="#">seq64::rect</a>	A small helper class representing a rectangle . . . . .	817
<a href="#">seq64::gui_drawingarea_gtk2::rect</a>	A small helper structure representing a rectangle . . . . .	818
<a href="#">seq64::rterror</a>	Exception handling class for rtexmidi . . . . .	819
<a href="#">seq64::rtmidi</a>	The main class of the rtmidi API . . . . .	822
<a href="#">seq64::rtmidi_in</a>	A realtime MIDI input class . . . . .	830
<a href="#">seq64::rtmidi_in_data</a>	The <a href="#">rtmidi_in_data</a> structure is used to pass private class data to the MIDI input handling function or thread . . . . .	834
<a href="#">seq64::rtmidi_info</a>	A class for enumerating MIDI clients and ports . . . . .	839
<a href="#">seq64::rtmidi_out</a>	A realtime MIDI output class . . . . .	850
<a href="#">seq64::Seq24PerfInput</a>	Implements the default (Seq24) performance input characteristics of this application . . . . .	852
<a href="#">seq64::Seq24SeqEventInput</a>	This structure implement the normal interaction methods for Seq24 . . . . .	856
<a href="#">seq64::seqdata</a>	This class supports drawing piano-roll events on a window . . . . .	858
<a href="#">seq64::seqedit</a>	Implements the Pattern Editor, which has references to: . . . . .	871
<a href="#">seq64::seqevent</a>	Implements the piano event drawing area . . . . .	904
<a href="#">seq64::seqkeys</a>	This class implements the left side piano of the pattern/sequence editor . . . . .	918
<a href="#">seq64::seqmenu</a>	This class handles the right-click menu of the sequence slots in the pattern window . . . . .	930
<a href="#">seq64::seqroll</a>	Implements the piano roll section of the pattern editor . . . . .	943
<a href="#">seq64::seqtime</a>	This class implements the piano time, whatever that is . . . . .	976
<a href="#">seq64::sequence</a>	Firstly a receptable for a single track of MIDI data read from a MIDI file or edited into a pattern . . . . .	982
<a href="#">seq64::trigger</a>	This class hold a single trigger for a sequence object . . . . .	1053
<a href="#">seq64::triggers</a>	Receptable the triggers that can be used with a sequence object . . . . .	1057
<a href="#">seq64::user_instrument</a>	Provides data about the MIDI instruments, readable from the "user" configuration file . . . . .	1074

[seq64::user\\_instrument\\_t](#)

This structure corresponds to [user-instrument-N] definitions in the `~/ .seq24usr` or `~/ .config/sequencer64/sequencer64.usr` file . . . . . 1078

[seq64::user\\_midi\\_bus](#)

Provides data about the MIDI busses, readable from the "user" configuration file . . . . . 1079

[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) . . . . . 1084

[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 . . . . . 1084

[seq64::userfile](#)

Supports the user's `~/ .config/sequencer64/sequencer64.usr` and `~/ .seq24usr` configuration file . . . . . 1119





## Chapter 12

# Namespace Documentation

### 12.1 Gtk Namespace Reference

### 12.2 seq64 Namespace Reference

Define this macro to use the new seq24 v.

#### Data Structures

- class [AbstractPerfInput](#)  
*Provides an abstract base class to provide the minimal interface for the various "perf input" classes.*
- 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.*
- struct [FruitySeqEventInput](#)  
*This structure implements the interaction methods for the "fruity" mode of operation.*
- 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 ALSA 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 [perfedit](#)  
*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 [perroll](#)  
*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](#)  
*A small helper class representing a rectangle.*
- 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.*
- struct [Seq24SeqEventInput](#)  
*This structure implement 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 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 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 copped 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_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 codes for the mouse-handling used by the application.*

- 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` }

*Provides a set of methods for drawing certain items.*

- 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.*
- `bool 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.*
- `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, midibpm bpm, int ppqn)`  
*Converts a MIDI pulse/ticks/clock value into a string that represents "hours:minutes:seconds.fraction".*
- `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".*
- `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 &timestring, int bpm, int ppqn)`
- `midipulse string_to_pulses (const std::string &s, const midi_timing &mt)`  
*Converts a time string to pulses.*
- `midibyte string_to_midibyte (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.*
- `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.*
- `int zoom_power_of_2 (int ppqn)`  
*Calculates a suitable starting zoom value for the given PPQN value.*
- `double bpm_from_tempo_us (double tempous)`  
*This function calculates the effective beats-per-minute based on the value of a Tempo meta-event.*
- `double tempo_us_from_bpm (midibpm bpm)`  
*Converts tempo (e.g.*  
*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](#) (midibpm bpm, int ppqn, int bw, int measures=1)

*Calculates the length of an integral number of measures, in ticks.*
- 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".*
- 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 &errmessage, int argc, char \*argv [])

*Provides the command-line option support, as well as some setup support, extracted from the main routine of Sequencer64.*
- 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.*
- 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.*
- 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.*
- 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.*
- bool [ppqn\\_is\\_valid](#) (int ppqn)

*Common code for handling PPQN settings.*
- int [jack\\_sync\\_callback](#) (jack\_transport\_state\_t state, jack\_position\_t \*pos, void \*arg)

*Global functions for JACK support and JACK sessions.*
- void [jack\\_shutdown\\_callback](#) (void \*arg)

*This callback is to shut down JACK by clearing the [jack\\_assistant](#) :: m\_jack\_running flag.*
- 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)
- 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)
- void [jack\\_session\\_callback](#) (jack\_session\_event\_t \*ev, void \*arg)

*Set the m\_jsession\_ev (event) value of the perform object.*
- bool [invalid\\_key](#) (unsigned int key)
- std::string [keyval\\_name](#) (unsigned int key)

*Obtains the name of the key.*
- 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 &timestring, [midibpm](#) bpm, int ppqn)

*Converts a string that represents "hours:minutes:seconds.fraction" into a MIDI pulse/ticks/clock value.*
- int [jack\\_dummy\\_callback](#) (jack\_nframes\_t, 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_renderer ()`  
*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)`
- `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.*
- `static long clamp (long val, long low, long hi)`  
*An internal function used by the FruitySeqRollInput class.*
- `void silence_jack_errors (bool silent)`  
*This function silences JACK error output to the console.*
- `void silence_jack_info (bool silent)`  
*This function silences JACK info output to the console.*
- `std::string midi_api_name (int i)`  
*Function to get RtMidi API names in a reusable manner.*
- `int midi_probe ()`  
*Formerly the main program of the RtMidi test program midiprobe.*
- `bool midi_input_test (rtmidi_info &info, int portindex)`

- *Provides testing the MIDI input process for 10 seconds.*
- long `min` (long a, long b)  
*min()* for long values.
- int `jack_process_rtmidi_input` (jack\_nframes\_t nframes, void \*arg)  
*Defines the JACK input process callback.*
- int `jack_process_rtmidi_output` (jack\_nframes\_t nframes, void \*arg)  
*Defines the JACK process input callback.*
- int `jack_process_io` (jack\_nframes\_t nframes, void \*arg)  
*Provides a JACK callback function that uses the callbacks defined in the `midi_jack` module.*
- static void `jack_message_bit_bucket` (const char \*)  
*This function merely eats the string passed as a parameter.*
- static void `midi_input_callback` (midi\_message &message, void \*)  
*Provides the callback for `midi_input_test()`.*

## 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_MIDI_SYSEX`  
*System Messages.*
- 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_SENS`
- 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_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\\_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](#)

*Perfedt beats/measure.*

- const midilong [c\\_perf\\_bw](#)

*Perfedt beat-width.*

- 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\\_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\\_16](#)
- const int [c\\_midi\\_control\\_17](#)
- const int [c\\_midi\\_control\\_18](#)
- const int [c\\_midi\\_control\\_19](#)
- const int [c\\_midi\\_controls\\_extended](#)
- int [g\\_midi\\_control\\_limit](#)
- 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 more help text.*
- static const char \*const [s\\_help\\_4](#)  
*Still more help text.*
- static const std::string [s\\_build\\_alsamidi\\_support](#)  
*This section of variables provide static information about the options enabled or disabled during the build.*
- static const std::string [s\\_build\\_portmidi\\_support](#)
- static const std::string [s\\_build\\_rtmidi\\_support](#)
- static const std::string [s\\_build\\_highlight\\_empty](#)
- static const std::string [s\\_build\\_lash\\_support](#)
- static const std::string [s\\_build\\_jack\\_support](#)
- static const std::string [s\\_build\\_jack\\_session](#)
- static const std::string [s\\_event\\_editor](#)
- static const std::string [s\\_build\\_pause\\_support](#)
- static const std::string [s\\_build\\_use\\_event\\_map](#)
- static const std::string [s\\_build\\_chord\\_generator](#)
- static const std::string [s\\_build\\_edit\\_highlight](#)
- static const std::string [s\\_build\\_timesig\\_tempo](#)
- static const std::string [s\\_build\\_midi\\_vector](#)
- static const std::string [s\\_build\\_solid\\_grid](#)
- static const std::string [s\\_build\\_follow\\_progress](#)
- static const std::string [s\\_statistics\\_support](#)
- static const std::string [s\\_strip\\_empty\\_mutes](#)
- static const std::string [s\\_seq32\\_jack\\_support](#)
- static const std::string [s\\_seq32\\_transport](#)
- static const std::string [s\\_seq32\\_transpose](#)
- static const std::string [s\\_seq32\\_menu\\_buttons](#)
- static const std::string [s\\_seq32\\_lfo\\_support](#)
- static const std::string [s\\_debug\\_mode](#)
- [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 const int [c\\_status\\_replace](#)  
*Purely internal constants used with the functions that implement MIDI control for the application.*
- static const int [c\\_status\\_snapshot](#)  
*This value signals the "snapshot" functionality.*
- static const int [c\\_status\\_queue](#)  
*This value signals the "queue" functionality.*
- 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 long `s_handlesize`  
*An internal variable for handle size.*
- 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.*
- const int `c_mainwid_x`  
*The width of the main pattern/sequence grid, in pixels.*
- const int `c_mainwid_y`
- static `perfed` \* `gs_perfed_pointer_0`  
*Holds a pointer to the first instance of perfedit for the entire application, once it is created.*
- static `perfed` \* `gs_perfed_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.*

### 12.2.1 Detailed Description

Do not document a namespace; it breaks Doxygen.

*Obsolete* Now a permanent option.

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.

Provides a new option to save the Time Signature and Tempo data that may be present in a MIDI file (in the first track) in the sequence object, and write them back to the MIDI file when saved again, in Sequencer64 format. The SeqSpec events that Seq24 and Sequencer64 save for these "events" are not readable by other MIDI applications, such as QTractor. By enabling this macro, other sequencers can read the correct time-signature and tempo values.

```
#define SEQ64_HANDLE_TIMESIG_AND_TEMPO
```

### 12.2.2 Typedef Documentation

#### 12.2.2.1 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.

#### 12.2.2.2 bussbyte

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



## 12.2.2.3 midishort

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

## 12.2.2.4 midilong

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

## 12.2.2.5 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.

## 12.2.2.6 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.

## 12.2.2.7 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.

## 12.2.2.8 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.

### 12.2.3 Enumeration Type Documentation

#### 12.2.3.1 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.

#### 12.2.3.2 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_UVALENT()` 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	

### 12.2.3.3 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 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	

### 12.2.3.4 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_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_SCROLL_UP	
SEQ64_SCROLL_DOWN	
SEQ64_SCROLL_LEFT	
SEQ64_SCROLL_RIGHT	

### 12.2.3.5 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_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 synthesizer 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.

### 12.2.3.6 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.

### 12.2.3.7 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	
-------------	--

## Enumerator

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	

## 12.2.3.8 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.

## 12.2.3.9 mouse\_action\_e

```
enum seq64::mouse_action_e
```

## Enumerator

e_action_select	
e_action_draw	
e_action_grow	

## 12.2.3.10 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	
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	

## 12.2.3.11 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.

## 12.2.4 Function Documentation

## 12.2.4.1 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.

## 12.2.4.2 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.

## 12.2.4.3 extract\_timing\_numbers()

```
bool 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
  - "213:4:920"
  - "0:1:0"
- hours : minutes : seconds . fraction
  - "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.

**Parameters**

	<i>s</i>	Provides the input time string, in measures or time format, to be processed.
out	<i>part</i> <sub>↔</sub> <i>_1</i>	The destination reference for the first part of the time.
out	<i>part</i> <sub>↔</sub> <i>_2</i>	The destination reference for the second part of the time.
out	<i>part</i> <sub>↔</sub> <i>_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 true if a reasonable portion (3 numbers) was good for extraction. The fraction part will start with a period for easier conversion to fractional seconds.

**12.2.4.4 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.

**12.2.4.5 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.

**12.2.4.6 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.

**12.2.4.7 pulses\_to\_timestring()** [1/2]

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

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.

**Returns**

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

**12.2.4.8 pulses\_to\_timestring()** [2/2]

```
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.

**12.2.4.9 measurestring\_to\_pulses()**

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

**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.

## 12.2.4.10 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.

## 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.

## 12.2.4.11 timestring\_to\_pulses() [1/2]

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

## 12.2.4.12 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.

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).

**12.2.4.13 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.

**12.2.4.14 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.

**12.2.4.15 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

<code>s</code>	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.

#### 12.2.4.16 [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

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

#### Returns

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

#### 12.2.4.17 [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.

**12.2.4.18 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.

**12.2.4.19 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  $((t2 * 256) + t1) * 256 + t0$ .

As a test case, note that the default tempo is 120 beats/minute, which is equivalent to  $ttttt=500000$  (0x07A120).

**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.

**12.2.4.20 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.

#### 12.2.4.21 bpm\_from\_tempo\_us()

```
double 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.

#### 12.2.4.22 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.

#### 12.2.4.23 pulse\_length\_us()

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

The formula for the pulse-length in seconds is:

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

##### 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

#### 12.2.4.24 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.

**12.2.4.25 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.

**12.2.4.26 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

12.2.4.27 `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.

12.2.4.28 `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.

## 12.2.4.29 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.

## 12.2.4.30 measures\_to\_ticks()

```
midipulse seq64::measures_to_ticks (
    midibpm bpm,
    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>bpm</i>	The B value in the equation, beats/measure.
<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.

#### 12.2.4.31 wave\_func()

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

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

#### 12.2.4.32 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.

#### 12.2.4.33 `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.

#### 12.2.4.34 `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.

#### 12.2.4.35 `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.

**12.2.4.36 parse\_options\_files()**

```
bool seq64::parse_options_files (
    perform & p,
    std::string & errmessage,
    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()`. 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>errmessage</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.

**12.2.4.37 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.

**12.2.4.38 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.

**12.2.4.39 build\_details()**

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

**Returns**

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

**12.2.4.40 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 <b>func</b> macro.
<i>m2</i>	The second message.

**Returns**

Returns "m1: m2" as a standard C++ string.

**12.2.4.41 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.

**12.2.4.42 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.

**12.2.4.43 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.

**12.2.4.44 file\_access()**

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

**12.2.4.45 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.

#### 12.2.4.46 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.

#### 12.2.4.47 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.

#### 12.2.4.48 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.

#### 12.2.4.49 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.

#### 12.2.4.50 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.

#### 12.2.4.51 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.

##### 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.

**12.2.4.52 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).

**12.2.4.53 jack\_sync\_callback()**

```
int seq64::jack_sync_callback (
    jack_transport_state_t state,
    jack_position_t * pos,
    void * arg )
```

This JACK synchronization callback informs the specified perform object of the current state and parameters of JACK.

The transport state will be:

- JackTransportStopped when a new position is requested.
- JackTransportStarting when the transport is waiting to start.
- JackTransportRolling when the timeout has expired, and the position is now a moving target.

This is the slow-sync callback, which the stazed code replaces with [jack\\_transport\\_callback\(\)](#).

**Parameters**

<i>state</i>	The JACK Transport state.
<i>pos</i>	The JACK position value.
<i>arg</i>	The pointer to the <a href="#">jack_assistant</a> object. Currently not checked for nullity, nor dynamic-casted.

**Returns**

Returns 1 if the function works, and 0 if something was wrong.

**12.2.4.54 jack\_shutdown\_callback()**

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

**Parameters**

<i>arg</i>	Points to the <a href="#">jack_assistant</a> in charge of JACK support for the perform object.
------------	--

**12.2.4.55 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://esseyj.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 seconds 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-&gt;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 Sequencer64 at present, and 1 if one, say, presses "rewind" in <code>qjackctl</code> .
<i>arg</i>	Provides the <a href="#">jack_assistant</a> pointer, currently unchecked for nullity.

**12.2.4.56 jack\_transport\_callback()**

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

**12.2.4.57 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:

#### ifdef SEQ64\_JACK\_SESSION

#### else

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

#### 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.

#### 12.2.4.58 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.
-------------	---

#### 12.2.4.59 get\_current\_jack\_position()

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

#### Warning

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

#### 12.2.4.60 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 <a href="#">jack_assistant</a> object. Currently not checked for nullity.

12.2.4.61 `invalid_key()`

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

12.2.4.62 `keyval_name()`

```
std::string seq64::keyval_name (
    unsigned int key )
```

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.

12.2.4.63 `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.
----------	---

#### 12.2.4.64 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.

#### 12.2.4.65 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.

#### 12.2.4.66 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.

#### 12.2.4.67 millisleep()

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

#### 12.2.4.68 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.

#### 12.2.4.69 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.

#### 12.2.4.70 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.

#### 12.2.4.71 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.

12.2.4.72 `usr()`

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

## Returns

Returns the global object `g_user_settings`.

12.2.4.73 `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).

12.2.4.74 `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".
<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.

#### 12.2.4.75 jack\_dummy\_callback()

```
int seq64::jack_dummy_callback (
    jack_nframes_t ,
    void * arg )
```

##### Returns

Does nothing, always returns 0.

#### 12.2.4.76 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.

#### 12.2.4.77 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.

#### 12.2.4.78 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.

#### 12.2.4.79 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]".

## 12.2.4.80 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`.

## 12.2.4.81 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.

## 12.2.4.82 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.
-----------	-----------------------------------

**12.2.4.83 is\_shift\_key()** [1/3]

```
bool seq64::is_shift_key (  
    GdkEventKey * ev )
```

## Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

**12.2.4.84 is\_no\_modifier()**

```
bool seq64::is_no_modifier (  
    GdkEventScroll * ev )
```

## Parameters

<i>ev</i>	The scroll event to be tested.
-----------	--------------------------------

**12.2.4.85 is\_ctrl\_key()** [2/3]

```
bool seq64::is_ctrl_key (  
    GdkEventScroll * ev )
```

## Parameters

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

**12.2.4.86 is\_shift\_key()** [2/3]

```
bool seq64::is_shift_key (  
    GdkEventScroll * ev )
```

**Parameters**

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

**12.2.4.87 is\_ctrl\_key()** [3/3]

```
bool seq64::is_ctrl_key (  
    GdkEventButton * ev )
```

**Parameters**

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

**12.2.4.88 is\_shift\_key()** [3/3]

```
bool seq64::is_shift_key (  
    GdkEventButton * ev )
```

**Parameters**

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

**12.2.4.89 is\_ctrl\_shift\_key()**

```
bool seq64::is_ctrl_shift_key (  
    GdkEventButton * ev )
```

**Parameters**

<i>ev</i>	The keystroke event to be tested.
-----------	-----------------------------------

**12.2.4.90 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.
-----------	-----------------------------------

## 12.2.4.91 test\_widget\_click()

```
void seq64::test_widget_click (
    GtkWidget * w )
```

## 12.2.4.92 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).

## 12.2.4.93 update\_perfedit\_sequences()

```
void seq64::update_perfedit_sequences ( )
```

It is used by other objects (seqedit and eventedit) that can modify the currently-edited sequence shown in the perfedit (song window).

## 12.2.4.94 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.

## 12.2.4.95 clamp() [1/2]

```
static long seq64::clamp (
    long val,
```

```

    long low,
    long hi ) [inline], [static]

```

#### 12.2.4.96 clamp() [2/2]

```

static long seq64::clamp (
    long val,
    long low,
    long hi ) [inline], [static]

```

#### 12.2.4.97 silence\_jack\_errors()

```

void seq64::silence_jack_errors (
    bool silent )

```

Probably not good to silence this output, but let's provide the option, for the sake of symmetry, consistency, what have you.

#### 12.2.4.98 silence\_jack\_info()

```

void seq64::silence_jack_info (
    bool silent )

```

We were getting way too many informational message, to the point of obscuring the debug and error output.

#### 12.2.4.99 midi\_api\_name()

```

std::string seq64::midi_api_name (
    int i )

```

##### Parameters

<i>i</i>	The integer value code for the desired API. Must range from int(RTMIDI_API_UNSPECIFIED) to int(RTMIDI_API_DUMMY).
----------	---

##### Returns

Returns a human-readable name for the API.

#### 12.2.4.100 midi\_probe()

```

int seq64::midi_probe ( )

```

We will upgrade this function for some better testing eventually. It uses the functionality of the `midi_info`/`rtmidi_info` objects, plus its own version of some of that functionality.

#### Returns

Currently always returns 0.

#### 12.2.4.101 `midi_input_test()`

```
bool seq64::midi_input_test (
    rtmidi_info & info,
    int portindex )
```

#### 12.2.4.102 `min()`

```
long seq64::min (
    long a,
    long b ) [inline]
```

#### Parameters

<i>a</i>	First operand.
<i>b</i>	Second operand.

#### Returns

Returns the minimum value of a and b.

#### 12.2.4.103 `jack_process_rtmidi_input()`

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

It is the JACK process callback for a MIDI output port (e.g. "system:midi\_capture\_1", which gives us the output of the Korg nanoKEY2 MIDI controller), also known as a "Readable Client" by qjackctl. It does the following:

This callback receives data from JACK and gives it to our application's input port.

This function does the following:

```

-# Get the JACK port buffer and the MIDI event-count into this
  buffer.
-# For each MIDI event, get the event from JACK and push it into a
  local midi_message object.
-# Get the event time, converting it to a delta time if possible.
-# If it is not a SysEx continuation, then:
  -# If we're using a callback, pass the data to that callback. Do
    we need this callback to interface with the midibus-based
    code?
  -# Otherwise, add the midi_message container to the rtmidi input
    queue. One can then grab this data in a midibase ::
    poll_for_midi() call. We still ought to check the add
    success.

```

The ALSA code polls for events, and that model is also available here. We're still working exactly how it will work best.

This function used to be static, but now we make it available to [midi\\_jack\\_info](#). Also note the `s_null_detected` flag. It is used only to have the `apiprint()` debug messages appear only once, for better trouble-shooting.

#### Parameters

<i>nframes</i>	The frame number to be processed.
<i>arg</i>	A pointer to the <a href="#">midi_jack_data</a> structure to be processed.

#### Returns

Returns 0.

#### 12.2.4.104 jack\_process\_rtmidi\_output()

```

int seq64::jack_process_rtmidi_output (
    jack_nframes_t nframes,
    void * arg )

```

It is the JACK process callback for a MIDI input port (a [midi\\_in\\_jack](#) object associated with, for example, "system↵:midi\_playback\_1", representing, for example, a Korg nanoKEY2 to which we can send information), also known as a "Writable Client" by qjackctl. Here's how it works:

```

-# Get the JACK port buffer, for our local jack port. Clear it.
-# Loop while the number of bytes available for reading [via
  jack_ringbuffer_read_space()] is non-zero. Note that the second
  parameter is where the data is copied.
-# Get the size of each event, and allocate space for an event to be
  written to an event port buffer (the JACK "reserve" function).
-# Read the data from the ringbuffer into this port buffer. JACK
  should then send it to the remote port.

```

Since this is an output port, "buff" is the area to which we can write data, to send it to the "remote" (i.e. outside our application) port. The data is written to the ringbuffer in `api_init_out()`, and here we read the ring buffer and pass it to the output buffer.

We were wondering if, like the JACK midiseq example program, we need to wrap our process in a for-loop over the number of frames. In our tests, we are getting 1024 frames, and the code seems to work without that loop.

## Parameters

<i>nframes</i>	The frame number to be processed.
<i>arg</i>	A pointer to the JackMIDIData structure to be processed.

## Returns

Returns 0.

## 12.2.4.105 jack\_process\_io()

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

## 12.2.4.106 jack\_message\_bit\_bucket()

```
static void seq64::jack_message_bit_bucket (
    const char * ) [static]
```

## 12.2.4.107 midi\_input\_callback()

```
static void seq64::midi_input_callback (
    midi_message & message,
    void * ) [static]
```

## 12.2.5 Variable Documentation

## 12.2.5.1 c\_controller\_names

```
std::string seq64::c_controller_names
```

This array is used only by the seqedit class.

## 12.2.5.2 EVENT\_STATUS\_BIT

```
const midibyte seq64::EVENT_STATUS_BIT
```

### 12.2.5.3 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.

### 12.2.5.4 EVENT\_NOTE\_OFF

```
const midibyte seq64::EVENT_NOTE_OFF
```

### 12.2.5.5 EVENT\_NOTE\_ON

```
const midibyte seq64::EVENT_NOTE_ON
```

### 12.2.5.6 EVENT\_AFTERTOUCH

```
const midibyte seq64::EVENT_AFTERTOUCH
```

### 12.2.5.7 EVENT\_CONTROL\_CHANGE

```
const midibyte seq64::EVENT_CONTROL_CHANGE
```

### 12.2.5.8 EVENT\_PROGRAM\_CHANGE

```
const midibyte seq64::EVENT_PROGRAM_CHANGE
```

### 12.2.5.9 EVENT\_CHANNEL\_PRESSURE

```
const midibyte seq64::EVENT_CHANNEL_PRESSURE
```

#### 12.2.5.10 EVENT\_PITCH\_WHEEL

```
const midibyte seq64::EVENT_PITCH_WHEEL
```

#### 12.2.5.11 EVENT\_MIDI\_SYSEX

```
const midibyte seq64::EVENT_MIDI_SYSEX
```

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.

#### 12.2.5.12 EVENT\_MIDI\_QUARTER\_FRAME

```
const midibyte seq64::EVENT_MIDI_QUARTER_FRAME
```

#### 12.2.5.13 EVENT\_MIDI\_SONG\_POS

```
const midibyte seq64::EVENT_MIDI_SONG_POS
```

#### 12.2.5.14 EVENT\_MIDI\_SONG\_SELECT

```
const midibyte seq64::EVENT_MIDI_SONG_SELECT
```

#### 12.2.5.15 EVENT\_MIDI\_SONG\_F4

```
const midibyte seq64::EVENT_MIDI_SONG_F4
```

**12.2.5.16 EVENT\_MIDI\_SONG\_F5**

```
const midibyte seq64::EVENT_MIDI_SONG_F5
```

**12.2.5.17 EVENT\_MIDI\_TUNE\_SELECT**

```
const midibyte seq64::EVENT_MIDI_TUNE_SELECT
```

**12.2.5.18 EVENT\_MIDI\_SYSEX\_END**

```
const midibyte seq64::EVENT_MIDI_SYSEX_END
```

**12.2.5.19 EVENT\_MIDI\_SYSEX\_CONTINUE**

```
const midibyte seq64::EVENT_MIDI_SYSEX_CONTINUE
```

**12.2.5.20 EVENT\_MIDI\_CLOCK**

```
const midibyte seq64::EVENT_MIDI_CLOCK
```

**12.2.5.21 EVENT\_MIDI\_SONG\_F9**

```
const midibyte seq64::EVENT_MIDI_SONG_F9
```

**12.2.5.22 EVENT\_MIDI\_START**

```
const midibyte seq64::EVENT_MIDI_START
```

**12.2.5.23 EVENT\_MIDI\_CONTINUE**

```
const midibyte seq64::EVENT_MIDI_CONTINUE
```



#### 12.2.5.24 EVENT\_MIDI\_STOP

```
const midibyte seq64::EVENT_MIDI_STOP
```

#### 12.2.5.25 EVENT\_MIDI\_SONG\_FD

```
const midibyte seq64::EVENT_MIDI_SONG_FD
```

#### 12.2.5.26 EVENT\_MIDI\_ACTIVE\_SENS

```
const midibyte seq64::EVENT_MIDI_ACTIVE_SENS
```

#### 12.2.5.27 EVENT\_MIDI\_RESET

```
const midibyte seq64::EVENT_MIDI_RESET
```

#### 12.2.5.28 EVENT\_MIDI\_META

```
const midibyte seq64::EVENT_MIDI_META
```

#### 12.2.5.29 EVENT\_NULL\_CHANNEL

```
const midibyte seq64::EVENT_NULL_CHANNEL
```

However, it also means that the channel 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.

#### 12.2.5.30 EVENT\_GET\_CHAN\_MASK

```
const midibyte seq64::EVENT_GET_CHAN_MASK
```

#### 12.2.5.31 EVENT\_CLEAR\_CHAN\_MASK

```
const midibyte seq64::EVENT_CLEAR_CHAN_MASK
```

#### 12.2.5.32 EVENTS\_ALL

```
const int seq64::EVENTS_ALL
```

We reversed the parts of each token for consistency with the macros defined above.

#### 12.2.5.33 EVENTS\_UNSELECTED

```
const int seq64::EVENTS_UNSELECTED
```

#### 12.2.5.34 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.

#### 12.2.5.35 c\_midibus\_input\_size

```
const int seq64::c_midibus_input_size
```

Not sure if the value needs to be so large.

#### 12.2.5.36 c\_midibus\_sysex\_chunk

```
const int seq64::c_midibus_sysex_chunk
```

### 12.2.5.37 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
```

Footer ("proprietary") 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)
```

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 new 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. Track buss number.

**12.2.5.38 c\_midich**

```
const midilong seq64::c_midich
```

**12.2.5.39 c\_midiclocks**

```
const midilong seq64::c_midiclocks
```

**12.2.5.40 c\_triggers**

```
const midilong seq64::c_triggers
```

**12.2.5.41 c\_notes**

```
const midilong seq64::c_notes
```

**12.2.5.42 c\_timesig**

```
const midilong seq64::c_timesig
```

**12.2.5.43 c\_bpmtag**

```
const midilong seq64::c_bpmtag
```

**12.2.5.44 c\_triggers\_new**

```
const midilong seq64::c_triggers_new
```

**12.2.5.45 c\_mutegroups**

```
const midilong seq64::c_mutegroups
```

#### 12.2.5.46 c\_midictrl

```
const midilong seq64::c_midictrl
```

#### 12.2.5.47 c\_musickey

```
const midilong seq64::c_musickey
```

#### 12.2.5.48 c\_musicscale

```
const midilong seq64::c_musicscale
```

#### 12.2.5.49 c\_backsequence

```
const midilong seq64::c_backsequence
```

#### 12.2.5.50 c\_transpose

```
const midilong seq64::c_transpose
```

#### 12.2.5.51 c\_perf\_bp\_mes

```
const midilong seq64::c_perf_bp_mes
```

#### 12.2.5.52 c\_perf\_bw

```
const midilong seq64::c_perf_bw
```

#### 12.2.5.53 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.

#### 12.2.5.54 c\_midi\_control\_bpm\_up

```
const int seq64::c_midi_control_bpm_up
```

#### 12.2.5.55 c\_midi\_control\_bpm\_dn

```
const int seq64::c_midi_control_bpm_dn
```

#### 12.2.5.56 c\_midi\_control\_ss\_up

```
const int seq64::c_midi_control_ss_up
```

#### 12.2.5.57 c\_midi\_control\_ss\_dn

```
const int seq64::c_midi_control_ss_dn
```

#### 12.2.5.58 c\_midi\_control\_mod\_replace

```
const int seq64::c_midi_control_mod_replace
```

**12.2.5.59 c\_midi\_control\_mod\_snapshot**

```
const int seq64::c_midi_control_mod_snapshot
```

**12.2.5.60 c\_midi\_control\_mod\_queue**

```
const int seq64::c_midi_control_mod_queue
```

**12.2.5.61 c\_midi\_control\_mod\_gmute**

```
const int seq64::c_midi_control_mod_gmute
```

**12.2.5.62 c\_midi\_control\_mod\_glearn**

```
const int seq64::c_midi_control_mod_glearn
```

**12.2.5.63 c\_midi\_control\_play\_ss**

```
const int seq64::c_midi_control_play_ss
```

**12.2.5.64 c\_midi\_controls**

```
const int seq64::c_midi_controls
```

**12.2.5.65 c\_midi\_control\_playback**

```
const int seq64::c_midi_control_playback
```

**12.2.5.66 c\_midi\_control\_record**

```
const int seq64::c_midi_control_record
```

**12.2.5.67 c\_midi\_control\_solo**

```
const int seq64::c_midi_control_solo
```

**12.2.5.68 c\_midi\_control\_thru**

```
const int seq64::c_midi_control_thru
```

**12.2.5.69 c\_midi\_control\_bpm\_page\_up**

```
const int seq64::c_midi_control_bpm_page_up
```

**12.2.5.70 c\_midi\_control\_bpm\_page\_dn**

```
const int seq64::c_midi_control_bpm_page_dn
```

**12.2.5.71 c\_midi\_control\_16**

```
const int seq64::c_midi_control_16
```

**12.2.5.72 c\_midi\_control\_17**

```
const int seq64::c_midi_control_17
```

**12.2.5.73 c\_midi\_control\_18**

```
const int seq64::c_midi_control_18
```

**12.2.5.74 c\_midi\_control\_19**

```
const int seq64::c_midi_control_19
```



## 12.2.5.75 c\_midi\_controls\_extended

```
const int seq64::c_midi_controls_extended
```

## 12.2.5.76 g\_midi\_control\_limit

```
int seq64::g_midi_control_limit
```

## 12.2.5.77 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.wikipedia.org/wiki/Heptatonic_scale)
- [https://en.wikibooks.org/wiki/Music\\_Theory/Scales\\_and\\_Intervals](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

## 12.2.5.78 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 .

### 12.2.5.79 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 .

#### 12.2.5.80 c\_scales\_text

```
const char seq64::c_scales_text[c_scale_size][20]
```

#### 12.2.5.81 c\_key\_text

```
const char seq64::c_key_text[SEQ64_OCTAVE_SIZE][4]
```

#### 12.2.5.82 c\_interval\_text

```
const char seq64::c_interval_text[16][4]
```

#### 12.2.5.83 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.

#### 12.2.5.84 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.

#### 12.2.5.85 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. :-)

#### 12.2.5.86 c\_chord\_size

```
const int seq64::c_chord_size
```

### 12.2.5.87 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.

### 12.2.5.88 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.

### 12.2.5.89 c\_max\_busses

```
const int seq64::c_max_busses
```

### 12.2.5.90 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";
```

### 12.2.5.91 long\_options

```
struct option seq64::long_options[] [static]
```

Note the terminating null structure..

### 12.2.5.92 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.

```
0123456789 @AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz
ooooooooo oxxxxxx x  xx  xx xxx xxxxxxxx  xx xxxxx xxxxa  x
```

Previous arg-list, items missing! "ChVH:lRrb;q:Lni:jJmaAM:pPusSU:x:"

**12.2.5.93 s\_help\_1a**

```
const char* const seq64::s_help_1a [static]
```

**12.2.5.94 s\_help\_1b**

```
const char* const seq64::s_help_1b [static]
```

**12.2.5.95 s\_help\_2**

```
const char* const seq64::s_help_2 [static]
```

**12.2.5.96 s\_help\_3**

```
const char* const seq64::s_help_3 [static]
```

**12.2.5.97 s\_help\_4**

```
const char* const seq64::s_help_4 [static]
```

**12.2.5.98 s\_build\_alsamidi\_support**

```
const std::string seq64::s_build_alsamidi_support [static]
```

**12.2.5.99 s\_build\_portmidi\_support**

```
const std::string seq64::s_build_portmidi_support [static]
```

**12.2.5.100 s\_build\_rtmidi\_support**

```
const std::string seq64::s_build_rtmidi_support [static]
```

**12.2.5.101 s\_build\_highlight\_empty**

```
const std::string seq64::s_build_highlight_empty [static]
```

**12.2.5.102 s\_build\_lash\_support**

```
const std::string seq64::s_build_lash_support [static]
```

**12.2.5.103 s\_build\_jack\_support**

```
const std::string seq64::s_build_jack_support [static]
```

**12.2.5.104 s\_build\_jack\_session**

```
const std::string seq64::s_build_jack_session [static]
```

**12.2.5.105 s\_event\_editor**

```
const std::string seq64::s_event_editor [static]
```

**12.2.5.106 s\_build\_pause\_support**

```
const std::string seq64::s_build_pause_support [static]
```

**12.2.5.107 s\_build\_use\_event\_map**

```
const std::string seq64::s_build_use_event_map [static]
```

**12.2.5.108 s\_build\_chord\_generator**

```
const std::string seq64::s_build_chord_generator [static]
```

**12.2.5.109 s\_build\_edit\_highlight**

```
const std::string seq64::s_build_edit_highlight [static]
```

**12.2.5.110 s\_build\_timesig\_tempo**

```
const std::string seq64::s_build_timesig_tempo [static]
```

**12.2.5.111 s\_build\_midi\_vector**

```
const std::string seq64::s_build_midi_vector [static]
```

**12.2.5.112 s\_build\_solid\_grid**

```
const std::string seq64::s_build_solid_grid [static]
```

**12.2.5.113 s\_build\_follow\_progress**

```
const std::string seq64::s_build_follow_progress [static]
```

**12.2.5.114 s\_statistics\_support**

```
const std::string seq64::s_statistics_support [static]
```

**12.2.5.115 s\_strip\_empty\_mutes**

```
const std::string seq64::s_strip_empty_mutes [static]
```

**12.2.5.116 s\_seq32\_jack\_support**

```
const std::string seq64::s_seq32_jack_support [static]
```

**12.2.5.117 s\_seq32\_transport**

```
const std::string seq64::s_seq32_transport [static]
```

**12.2.5.118 s\_seq32\_transpose**

```
const std::string seq64::s_seq32_transpose [static]
```

**12.2.5.119 s\_seq32\_menu\_buttons**

```
const std::string seq64::s_seq32_menu_buttons [static]
```

**12.2.5.120 s\_seq32\_lfo\_support**

```
const std::string seq64::s_seq32_lfo_support [static]
```

**12.2.5.121 s\_debug\_mode**

```
const std::string seq64::s_debug_mode [static]
```

**12.2.5.122 s\_status\_pairs**

```
jack_status_pair_t seq64::s_status_pairs[ ]
```

Terminated by a 0 value and an empty string.

**12.2.5.123 s\_character\_mapping**

```
struct charpair_t seq64::s_character_mapping[ ]
```



12.2.5.124 `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.

12.2.5.125 `c_status_replace`

```
const int seq64::c_status_replace [static]
```

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".

12.2.5.126 `c_status_snapshot`

```
const int seq64::c_status_snapshot [static]
```

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().
```

#### 12.2.5.127 c\_status\_queue

```
const int seq64::c_status_queue [static]
```

If this bit is set, then [perform::sequence\\_playing\\_toggle\(\)](#) calls [sequence::toggle\\_queued\(\)](#) on the given sequence number.

#### 12.2.5.128 g\_rc\_settings

```
rc_settings seq64::g_rc_settings [static]
```

#### 12.2.5.129 g\_user\_settings

```
user_settings seq64::g_user_settings [static]
```

#### 12.2.5.130 s\_handlesize [1/2]

```
const long seq64::s_handlesize [static]
```

#### 12.2.5.131 s\_jitter\_amount

```
const int seq64::s_jitter_amount [static]
```

#### 12.2.5.132 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.

#### 12.2.5.133 c\_mainwid\_x

```
const int seq64::c_mainwid_x
```

Affected by the `c_mainwid_border` and `c_mainwid_spacing` values.

**12.2.5.134 c\_mainwid\_y**

```
const int seq64::c_mainwid_y
```

**12.2.5.135 gs\_perfedit\_pointer\_0**

```
perfedit* seq64::gs_perfedit_pointer_0 [static]
```

**12.2.5.136 gs\_perfedit\_pointer\_1**

```
perfedit* seq64::gs_perfedit_pointer_1 [static]
```

**12.2.5.137 s\_handlesize** [2/2]

```
const long seq64::s_handlesize [static]
```



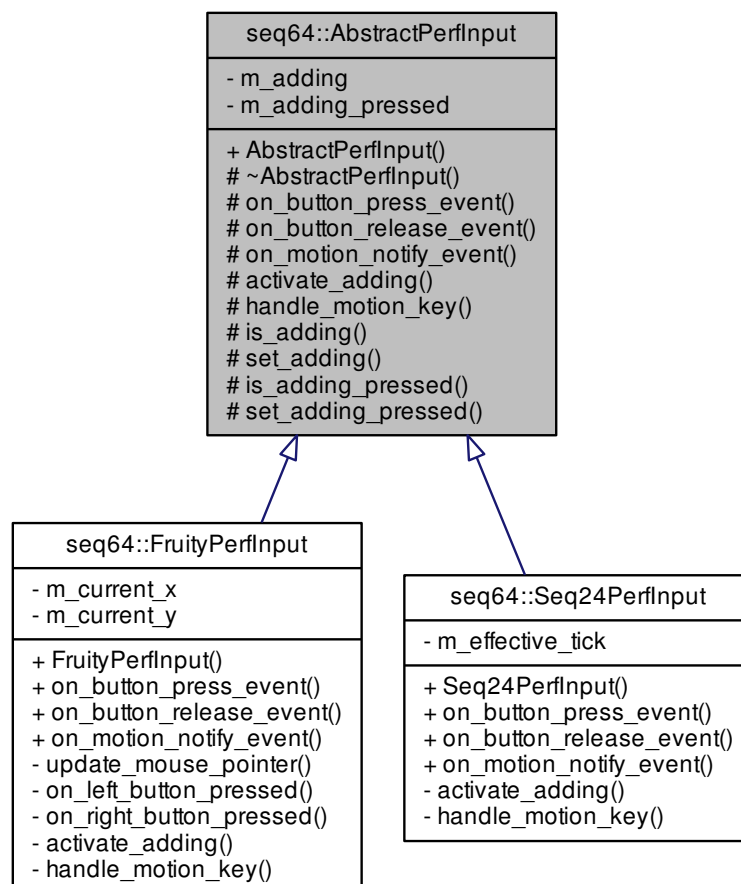
## Chapter 13

# Data Structure Documentation

### 13.1 seq64::AbstractPerInput Class Reference

Provides an abstract base class to provide the minimal interface for the various "perf input" classes.

Inheritance diagram for seq64::AbstractPerInput:



## Public Member Functions

- [AbstractPerfInput](#) ()  
*Default constructor.*

## Protected Member Functions

- virtual [~AbstractPerfInput](#) ()  
*Destructor, does nothing.*
- virtual bool [on\\_button\\_press\\_event](#) (GdkEventButton \*a\_ev, [perfroll](#) &roll)=0
- virtual bool [on\\_button\\_release\\_event](#) (GdkEventButton \*a\_ev, [perfroll](#) &roll)=0
- virtual bool [on\\_motion\\_notify\\_event](#) (GdkEventMotion \*a\_ev, [perfroll](#) &roll)=0
- virtual void [activate\\_adding](#) (bool adding, [perfroll](#) &roll)=0
- virtual bool [handle\\_motion\\_key](#) (bool is\_left, [perfroll](#) &roll)=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*

## Private Attributes

- 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.*

## Friends

- class [perfroll](#)

## 13.1.1 Constructor & Destructor Documentation

### 13.1.1.1 AbstractPerfInput()

```
seq64::AbstractPerfInput::AbstractPerfInput ( ) [inline]
```

### 13.1.1.2 ~AbstractPerfInput()

```
virtual seq64::AbstractPerfInput::~~AbstractPerfInput ( ) [inline], [protected], [virtual]
```

## 13.1.2 Member Function Documentation

### 13.1.2.1 on\_button\_press\_event()

```
virtual bool seq64::AbstractPerfInput::on_button_press_event (  
    GdkEventButton * a_ev,  
    perfroll & roll ) [protected], [pure virtual]
```

Implemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

### 13.1.2.2 on\_button\_release\_event()

```
virtual bool seq64::AbstractPerfInput::on_button_release_event (  
    GdkEventButton * a_ev,  
    perfroll & roll ) [protected], [pure virtual]
```

Implemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

### 13.1.2.3 on\_motion\_notify\_event()

```
virtual bool seq64::AbstractPerfInput::on_motion_notify_event (  
    GdkEventMotion * a_ev,  
    perfroll & roll ) [protected], [pure virtual]
```

Implemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

### 13.1.2.4 activate\_adding()

```
virtual void seq64::AbstractPerfInput::activate_adding (  
    bool adding,  
    perfroll & roll ) [protected], [pure virtual]
```

Implemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

### 13.1.2.5 handle\_motion\_key()

```
virtual bool seq64::AbstractPerfInput::handle_motion_key (  
    bool is_left,  
    perfroll & roll ) [protected], [pure virtual]
```

Implemented in [seq64::Seq24PerfInput](#), and [seq64::FruityPerfInput](#).

#### 13.1.2.6 is\_adding()

```
bool seq64::AbstractPerfInput::is_adding ( ) const [inline], [protected]
```

#### 13.1.2.7 set\_adding()

```
void seq64::AbstractPerfInput::set_adding (
    bool flag ) [inline], [protected]
```

#### 13.1.2.8 is\_adding\_pressed()

```
bool seq64::AbstractPerfInput::is_adding_pressed ( ) const [inline], [protected]
```

#### 13.1.2.9 set\_adding\_pressed()

```
void seq64::AbstractPerfInput::set_adding_pressed (
    bool flag ) [inline], [protected]
```

### 13.1.3 Friends And Related Function Documentation

#### 13.1.3.1 perfroll

```
friend class perfroll [friend]
```

### 13.1.4 Field Documentation

#### 13.1.4.1 m\_adding

```
bool seq64::AbstractPerfInput::m_adding [private]
```



## 13.1.4.2 m\_adding\_pressed

```
bool seq64::AbstractPerfInput::m_adding_pressed [private]
```

## 13.2 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.*

### 13.2.1 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.

### 13.2.2 Constructor & Destructor Documentation

#### 13.2.2.1 automutex() [1/3]

```
seq64::automutex::automutex ( ) [private]
```

#### 13.2.2.2 automutex() [2/3]

```
seq64::automutex::automutex (
    const automutex & ) [private]
```

#### 13.2.2.3 automutex() [3/3]

```
seq64::automutex::automutex (
    mutex & my\_mutex ) [inline]
```

## Parameters

<code>my_mutex</code>	The caller's mutex to be used for locking.
-----------------------	--

13.2.2.4 `~automutex()`

```
seq64::automutex::~~automutex ( ) [inline]
```

## 13.2.3 Member Function Documentation

13.2.3.1 `operator=()`

```
automutex& seq64::automutex::operator= (
    const automutex & ) [private]
```

## 13.2.4 Field Documentation

13.2.4.1 `m_safety_mutex`

```
mutex& seq64::automutex::m_safety_mutex [private]
```

13.3 `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 ()`  
*Print some information about the available MIDI 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.*
- `bool 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.*

## 13.3.1 Constructor & Destructor Documentation

### 13.3.1.1 busarray()

```
seq64::busarray::busarray ( )
```

### 13.3.1.2 ~busarray()

```
seq64::busarray::~~busarray ( )
```

However, now that we swap containers, we cannot call this functionality, because it deletes the bus's midibus pointer and nullifies it.

DISABLED, BUT WE NEED A WAY TO CLEAN UP AT EXIT TIME!!!

## 13.3.2 Member Function Documentation

### 13.3.2.1 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 delay 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.

**13.3.2.2 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.

**13.3.2.3 initialize()**

```
bool seq64::busarray::initialize ( )
```

Not sure we need this function.

**Returns**

Returns true if all busses initialized successfully.

**13.3.2.4 count()**

```
int seq64::busarray::count ( ) const [inline]
```

### 13.3.2.5 bus()

```
midibus* seq64::busarray::bus (
    bussbyte b ) [inline]
```

### 13.3.2.6 start()

```
void seq64::busarray::start ( )
```

### 13.3.2.7 stop()

```
void seq64::busarray::stop ( )
```

### 13.3.2.8 continue\_from()

```
void seq64::busarray::continue_from (
    midipulse tick )
```

#### Parameters

<i>tick</i>	Provides the tick value for all busses to continue from.
-------------	--

### 13.3.2.9 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.
-------------	---

### 13.3.2.10 clock()

```
void seq64::busarray::clock (
    midipulse tick )
```

## Parameters

<i>tick</i>	Provides the tick value for all busses use as the clock tick.
-------------	---

## 13.3.2.11 sysex()

```
void seq64::busarray::sysex (
    event * ev )
```

## Parameters

<i>ev</i>	Provides the SysEx event to handle.
-----------	-------------------------------------

## 13.3.2.12 play()

```
void seq64::busarray::play (
    bussbyte bus,
    event * e24,
    midibyte channel )
```

## Parameters

<i>bus</i>	The MIDI buss on which to play the event.
<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.

## 13.3.2.13 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.

13.3.2.14 `set_all_clocks()`

```
void seq64::busarray::set_all_clocks ( )
```

Note that the settings to apply are added when the `add()` call is made.

13.3.2.15 `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, then `e_clock_off` is returned.

13.3.2.16 `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.



## 13.3.2.17 print()

```
void seq64::busarray::print ( )
```

## 13.3.2.18 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.

## 13.3.2.19 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.

**13.3.2.20 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.

**13.3.2.21 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.

**13.3.2.22 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.

#### 13.3.2.23 poll\_for\_midi()

```
bool 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.

##### Returns

Returns true if a MIDI event was detected on one of the busses. Note that this is a boolean value, while the [midibase::poll\\_for\\_midi\(\)](#) function returns an integer.

#### 13.3.2.24 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.

#### 13.3.2.25 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.

### 13.3.3 Field Documentation

#### 13.3.3.1 m\_container

```
std::vector<businfo> seq64::busarray::m_container [private]
```

## 13.4 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](#) ()
- const [midibus](#) \* [bus](#) () const
- [midibus](#) \* [bus](#) ()
- bool [active](#) () const
- bool [initialize](#) ()  
*This function is called when the businfo object is added to the busarray.*
- bool [initialized](#) () const
- [clock\\_e](#) [init\\_clock](#) () const
- bool [init\\_input](#) () const
- void [bus](#) ([midibus](#) \*b)
- void [activate](#) ()
- void [deactivate](#) ()
- void [init\\_clock](#) ([clock\\_e](#) clocktype)
- void [init\\_input](#) (bool flag)

### 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)

## 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`

### 13.4.1 Detailed Description

There will be in input instance and an output instance of this object contained by mastermidibus.

### 13.4.2 Constructor & Destructor Documentation

#### 13.4.2.1 `businfo()` [1/3]

```
seq64::businfo::businfo ( )
```

There will be in input instance and an output instance of this object contained by mastermidibus.

#### 13.4.2.2 `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.
------------	--

13.4.2.3 **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.
------------	---------------------------------

13.4.2.4 **~businfo()**

```
seq64::businfo::~~businfo ( ) [inline]
```

## 13.4.3 Member Function Documentation

13.4.3.1 **remove()**

```
void seq64::businfo::remove ( ) [inline]
```

13.4.3.2 **bus()** [1/3]

```
const midibus* seq64::businfo::bus ( ) const [inline]
```

13.4.3.3 **bus()** [2/3]

```
midibus* seq64::businfo::bus ( ) [inline]
```

#### 13.4.3.4 active()

```
bool seq64::businfo::active ( ) const [inline]
```

#### 13.4.3.5 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.

#### 13.4.3.6 initialized()

```
bool seq64::businfo::initialized ( ) const [inline]
```

#### 13.4.3.7 init\_clock() [1/3]

```
clock_e seq64::businfo::init_clock ( ) const [inline]
```

#### 13.4.3.8 init\_input() [1/2]

```
bool seq64::businfo::init_input ( ) const [inline]
```

#### 13.4.3.9 bus() [3/3]

```
void seq64::businfo::bus (
    midibus * b ) [inline]
```

#### 13.4.3.10 activate()

```
void seq64::businfo::activate ( ) [inline]
```

#### 13.4.3.11 deactivate()

```
void seq64::businfo::deactivate ( ) [inline]
```

#### 13.4.3.12 init\_clock() [2/3]

```
void seq64::businfo::init_clock (
    clock_e clocktype ) [inline]
```

#### 13.4.3.13 init\_input() [2/2]

```
void seq64::businfo::init_input (
    bool flag ) [inline]
```

#### 13.4.3.14 start()

```
void seq64::businfo::start ( ) [inline], [private]
```



#### 13.4.3.15 stop()

```
void seq64::businfo::stop ( ) [inline], [private]
```

#### 13.4.3.16 continue\_from()

```
void seq64::businfo::continue_from (
    midipulse tick ) [inline], [private]
```

#### 13.4.3.17 init\_clock() [3/3]

```
void seq64::businfo::init_clock (
    midipulse tick ) [inline], [private]
```

#### 13.4.3.18 clock()

```
void seq64::businfo::clock (
    midipulse tick ) [inline], [private]
```

#### 13.4.3.19 sysex()

```
void seq64::businfo::sysex (
    event * ev ) [inline], [private]
```

### 13.4.4 Friends And Related Function Documentation

#### 13.4.4.1 busarray

```
friend class busarray [friend]
```

### 13.4.5 Field Documentation

#### 13.4.5.1 m\_bus

```
midibus* seq64::businfo::m_bus [private]
```

#### 13.4.5.2 m\_active

```
bool seq64::businfo::m_active [private]
```

#### 13.4.5.3 m\_initialized

```
bool seq64::businfo::m_initialized [private]
```

#### 13.4.5.4 m\_init\_clock

```
clock_e seq64::businfo::m_init_clock [private]
```

#### 13.4.5.5 m\_init\_input

```
bool seq64::businfo::m_init_input [private]
```

### 13.5 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.*

### 13.5.1 Detailed Description

Useful in passing more generic events to non-GUI classes.

## 13.5.2 Constructor & Destructor Documentation

### 13.5.2.1 `click()` [1/3]

```
seq64::click::click ( )
```

Sets all members to false, zero, or the lowest good value.

### 13.5.2.2 `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.

### 13.5.2.3 `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.
------------	---

## 13.5.3 Member Function Documentation

### 13.5.3.1 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.

### 13.5.3.2 is\_press()

```
bool seq64::click::is_press ( ) const [inline]
```

### 13.5.3.3 is\_left()

```
bool seq64::click::is_left ( ) const [inline]
```

### 13.5.3.4 is\_middle()

```
bool seq64::click::is_middle ( ) const [inline]
```

### 13.5.3.5 is\_right()

```
bool seq64::click::is_right ( ) const [inline]
```

### 13.5.3.6 x()

```
int seq64::click::x ( ) const [inline]
```

#### 13.5.3.7 y()

```
int seq64::click::y ( ) const [inline]
```

#### 13.5.3.8 button()

```
int seq64::click::button ( ) const [inline]
```

#### 13.5.3.9 modifier()

```
seq_modifier_t seq64::click::modifier ( ) const [inline]
```

#### 13.5.3.10 mod\_control()

```
bool seq64::click::mod_control ( ) const [inline]
```

#### 13.5.3.11 mod\_control\_shift()

```
bool seq64::click::mod_control_shift ( ) const [inline]
```

#### 13.5.3.12 mod\_super()

```
bool seq64::click::mod_super ( ) const [inline]
```

### 13.5.4 Field Documentation

#### 13.5.4.1 m\_is\_press

```
bool seq64::click::m_is_press [private]
```

## 13.5.4.2 m\_x

```
int seq64::click::m_x [private]
```

0 is the left-most coordinate.

## 13.5.4.3 m\_y

```
int seq64::click::m_y [private]
```

0 is the top-most coordinate.

## 13.5.4.4 m\_button

```
int seq64::click::m_button [private]
```

Left is 1, mmiddle is 2, and right is 3. These numbers are defined via macros, and are Linux-specific and Gtk-specific.

## 13.5.4.5 m\_modifier

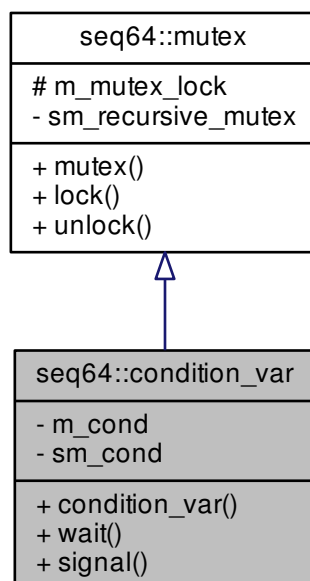
```
seq_modifier_t seq64::click::m_modifier [private]
```

Note that SEQ64\_NO\_MASK is our word for 0, meaning "no modifier".

## 13.6 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

### 13.6.1 Detailed Description

Therefore this class derives from the mutex class. A "has-a" relationship might be more logical than this "is-a" relationship.

### 13.6.2 Constructor & Destructor Documentation

#### 13.6.2.1 `condition_var()`

```
seq64::condition_var::condition_var ( )
```

### 13.6.3 Member Function Documentation

#### 13.6.3.1 `wait()`

```
void seq64::condition_var::wait ( )
```



### 13.6.3.2 signal()

```
void seq64::condition_var::signal ( )
```

## 13.6.4 Field Documentation

### 13.6.4.1 sm\_cond

```
const pthread_cond_t seq64::condition_var::sm_cond [static], [private]
```

Define the static condition variable used by all mutex locks.

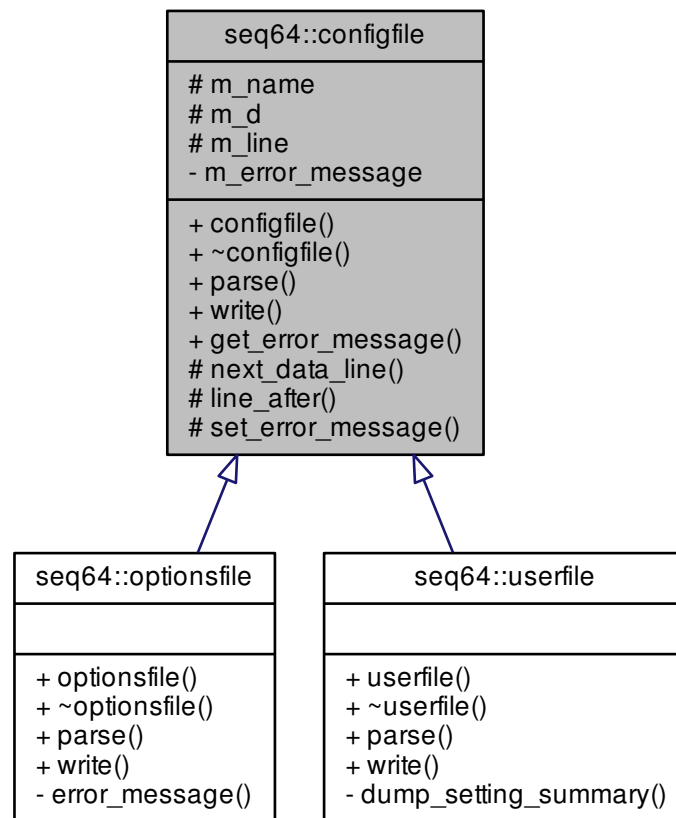
### 13.6.4.2 m\_cond

```
pthread_cond_t seq64::condition_var::m_cond [private]
```

## 13.7 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.*
- 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.*

## 13.7.1 Constructor & Destructor Documentation

### 13.7.1.1 configfile()

```
seq64::configfile::configfile (
    const std::string & name )
```

#### Parameters

<i>name</i>	The name of the configuration file.
-------------	-------------------------------------

### 13.7.1.2 ~configfile()

```
virtual seq64::configfile::~~configfile ( ) [inline], [virtual]
```

## 13.7.2 Member Function Documentation

### 13.7.2.1 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.

**13.7.2.2 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.

**13.7.2.3 parse()**

```
virtual bool seq64::configfile::parse (
    perform & perf ) [pure virtual]
```

Implemented in [seq64::userfile](#), and [seq64::optionsfile](#).

**13.7.2.4 write()**

```
virtual bool seq64::configfile::write (
    const perform & perf ) [pure virtual]
```

Implemented in [seq64::userfile](#), and [seq64::optionsfile](#).

#### 13.7.2.5 get\_error\_message()

```
const std::string& seq64::configfile::get_error_message ( ) const [inline]
```

#### 13.7.2.6 set\_error\_message()

```
void seq64::configfile::set_error_message (
    const std::string & msg ) [inline], [protected]
```

### 13.7.3 Field Documentation

#### 13.7.3.1 m\_error\_message

```
std::string seq64::configfile::m_error_message [private]
```

Not a 100% foolproof yet.

#### 13.7.3.2 m\_name

```
std::string seq64::configfile::m_name [protected]
```

#### 13.7.3.3 m\_d

```
char* seq64::configfile::m_d [protected]
```

#### 13.7.3.4 m\_line

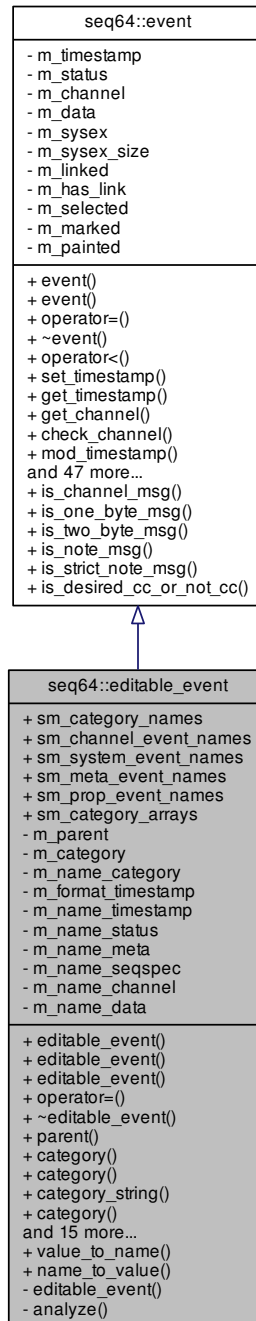
```
char seq64::configfile::m_line[SEQ64_LINE_MAX] [protected]
```

This member receives an input line, and so needs to be a character buffer.

## 13.8 seq64::editable\_event Class Reference

Provides for the management of MIDI editable events.

Inheritance diagram for seq64::editable\_event:



### Data Structures

- 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)  
*This constructor simply initializes all of the class members.*
- `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].*
- 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 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 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.*

## 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\\_seqspec](#)  
*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.*

### 13.8.1 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.

### 13.8.2 Member Enumeration Documentation

#### 13.8.2.1 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[].
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[].
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.

## 13.8.2.2 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.

## 13.8.3 Constructor &amp; Destructor Documentation

## 13.8.3.1 editable\_event() [1/4]

```
seq64::editable_event::editable_event ( ) [private]
```

13.8.3.2 `editable_event()` [2/4]

```
seq64::editable_event::editable_event (
    const editable_events & parent )
```

`editable_event::editable_event ()` : `event ()`, `m_category (category_name)`, `m_name_category ()`, `m_format ↵`  
`timestamp (timestamp_measures)`, `m_name_timestamp ()`, `m_name_status ()`, `m_name_meta ()`, `m_name ↵`  
`seqspec ()`, `m_name_channel ()`, `m_name_data ()` { // Empty body } Principal constructor.

## Parameters

<i>parent</i>	Provides the overall editable-events object that manages the whole set of editable-event.
---------------	---

13.8.3.3 `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.

13.8.3.4 `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.
------------	---

13.8.3.5 `~editable_event()`

```
virtual seq64::editable_event::~~editable_event ( ) [inline], [virtual]
```

### 13.8.4 Member Function Documentation

#### 13.8.4.1 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.

#### 13.8.4.2 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.

#### 13.8.4.3 operator=()

```
editable_event & seq64::editable_event::operator= (
    const editable_event & rhs )
```

**13.8.4.4 parent()**

```
const editable_events& seq64::editable_event::parent ( ) const [inline]
```

**13.8.4.5 category()** [1/3]

```
category_t seq64::editable_event::category ( ) const [inline]
```

**13.8.4.6 category()** [2/3]

```
void seq64::editable_event::category (
    category_t c )
```

Note that a bad value is translated to the value of category\_name.

**Parameters**

<i>c</i>	Provides the category value to set.
----------	-------------------------------------

**13.8.4.7 category\_string()**

```
const std::string& seq64::editable_event::category_string ( ) const [inline]
```

**13.8.4.8 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.
-------------	---

**13.8.4.9 timestamp\_string()**

```
const std::string& seq64::editable_event::timestamp_string ( ) const [inline]
```

**13.8.4.10 timestamp()** [1/3]

```
midipulse seq64::editable_event::timestamp ( ) const [inline]
```

**13.8.4.11 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.
-----------	---

**13.8.4.12 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.
------------------	---

**13.8.4.13 time\_as\_pulses()**

```
std::string seq64::editable_event::time_as_pulses ( ) [inline]
```

**13.8.4.14 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.

**13.8.4.15 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.

**13.8.4.16 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.
<i>sd1</i>	Provides the string defining the second data byte of the event, if applicable to the event.

**13.8.4.17 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.

**13.8.4.18 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.

#### 13.8.4.19 status\_string()

```
std::string seq64::editable_event::status_string ( ) const [inline]
```

#### 13.8.4.20 meta\_string()

```
std::string seq64::editable_event::meta_string ( ) const [inline]
```

#### 13.8.4.21 seqspec\_string()

```
std::string seq64::editable_event::seqspec_string ( ) const [inline]
```

#### 13.8.4.22 channel\_string()

```
std::string seq64::editable_event::channel_string ( ) const [inline]
```

#### 13.8.4.23 data\_string()

```
std::string seq64::editable_event::data_string ( ) const [inline]
```

#### 13.8.4.24 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.



### 13.8.5 Field Documentation

#### 13.8.5.1 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.

#### 13.8.5.2 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.

#### 13.8.5.3 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.

#### 13.8.5.4 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.

#### 13.8.5.5 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.

#### 13.8.5.6 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.

#### 13.8.5.7 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.

#### 13.8.5.8 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.

#### 13.8.5.9 m\_name\_category

```
std::string seq64::editable_event::m_name_category [private]
```

#### 13.8.5.10 m\_format\_timestamp

```
timestamp_format_t seq64::editable_event::m_format_timestamp [private]
```

The default is to display in timestamp\_measures format.

#### 13.8.5.11 m\_name\_timestamp

```
std::string seq64::editable_event::m_name_timestamp [private]
```

#### 13.8.5.12 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.

## 13.8.5.13 m\_name\_meta

```
std::string seq64::editable_event::m_name_meta [private]
```

If not applicable, this name will be empty.

## 13.8.5.14 m\_name\_seqspect

```
std::string seq64::editable_event::m_name_seqspect [private]
```

## 13.8.5.15 m\_name\_channel

```
std::string seq64::editable_event::m_name_channel [private]
```

## 13.8.5.16 m\_name\_data

```
std::string seq64::editable_event::m_name_data [private]
```

## 13.9 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.*
- `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()`.*
- `iterator current_event () const`
  - 'Getter' function for member m\_current\_event The caller must make sure the iterator is not `Events::end()`.*

## 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 std::multimap< Key, editable_event >::iterator iterator`
- `typedef std::multimap< Key, editable_event >::const_iterator const_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](#)

## 13.9.1 Member Typedef Documentation

### 13.9.1.1 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.

### 13.9.1.2 EventsPair

```
typedef std::pair<Key, editable\_event> seq64::editable\_events::EventsPair [private]
```

### 13.9.1.3 Events

```
typedef std::multimap<Key, editable\_event> seq64::editable\_events::Events [private]
```

### 13.9.1.4 iterator

```
typedef std::multimap<Key, editable\_event>::iterator seq64::editable\_events::iterator [private]
```

### 13.9.1.5 const\_iterator

```
typedef std::multimap<Key, editable_event>::const_iterator seq64::editable_events::const_↵
iterator [private]
```

## 13.9.2 Constructor & Destructor Documentation

### 13.9.2.1 editable\_events() [1/3]

```
seq64::editable_events::editable_events ( ) [private]
```

### 13.9.2.2 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.

### 13.9.2.3 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 <code>editable_events</code> object to be copied.
------------	--

### 13.9.2.4 ~editable\_events()

```
virtual seq64::editable_events::~~editable_events ( ) [inline], [virtual]
```

### 13.9.3 Member Function Documentation

#### 13.9.3.1 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 <a href="#">editable_events</a> object to be assigned.
------------	---

##### Returns

Returns a reference to "this" object, to support the serial assignment of [editable\\_eventss](#).

#### 13.9.3.2 timing()

```
const midi_timing& seq64::editable_events::timing ( ) const [inline]
```

#### 13.9.3.3 string\_to\_pulses()

```
midipulse seq64::editable_events::string_to_pulses (
    const std::string & ts_string ) const [inline]
```

#### 13.9.3.4 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.

### 13.9.3.5 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.

### 13.9.3.6 events()

```
Events& seq64::editable_events::events ( ) [inline]
```

### 13.9.3.7 begin() [1/2]

```
iterator seq64::editable_events::begin ( ) [inline]
```

### 13.9.3.8 begin() [2/2]

```
const_iterator seq64::editable_events::begin ( ) const [inline]
```

### 13.9.3.9 end() [1/2]

```
iterator seq64::editable_events::end ( ) [inline]
```

### 13.9.3.10 end() [2/2]

```
const_iterator seq64::editable_events::end ( ) const [inline]
```

### 13.9.3.11 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.
-----------	---

## 13.9.3.12 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.
-----------	---

## 13.9.3.13 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.

## 13.9.3.14 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.

## 13.9.3.15 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.

13.9.3.16 `replace()`

```
bool seq64::editable_events::replace (
    iterator ie,
    const editable_event & e ) [inline]
```

13.9.3.17 `remove()`

```
void seq64::editable_events::remove (
    iterator ie ) [inline]
```

13.9.3.18 `clear()`

```
void seq64::editable_events::clear ( ) [inline]
```

13.9.3.19 `current_event()` [1/2]

```
iterator seq64::editable_events::current_event ( ) const [inline]
```

13.9.3.20 `current_event()` [2/2]

```
void seq64::editable_events::current_event (
    iterator cei ) [inline], [private]
```

## Parameters

<i>cei</i>	Provide an iterator to the event to set as the current event.
------------	---

## 13.9.4 Friends And Related Function Documentation

### 13.9.4.1 eventslots

```
friend class eventslots [friend]
```

## 13.9.5 Field Documentation

### 13.9.5.1 m\_events

```
Events seq64::editable_events::m_events [private]
```

### 13.9.5.2 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.

### 13.9.5.3 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.

### 13.9.5.4 m\_midi\_parameters

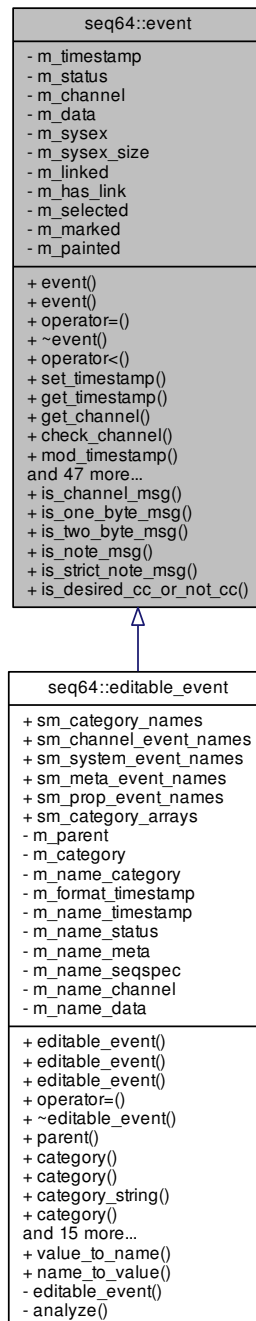
```
midi_timing seq64::editable_events::m_midi_parameters [private]
```

It holds the beats/minute, beats/measure, beat-width, and PPQN values needed to properly convert MIDI pulse timestamps to time and measure values.

## 13.10 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\\_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*
- 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, 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 obtaining SYSEX data byte-by-byte.*
- void `restart_sysex` ()  
*Deletes and clears out the SYSEX buffer.*
- `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` and `m_sysex_size` 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 print () const`  
*Prints out the timestamp, data size, the current status byte, any SYSEX 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_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.*

- int `m_sysex_size`

*Gives the size of the SYSEX message.*

- 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.".*

### 13.10.1 Detailed Description

A MIDI event consists of 3 bytes:

```
-# Status byte, lsssnnn, 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.

### 13.10.2 Member Typedef Documentation

#### 13.10.2.1 SysexContainer

```
typedef std::vector<midibyte> seq64::event::SysexContainer
```

### 13.10.3 Constructor & Destructor Documentation

#### 13.10.3.1 event() [1/2]

```
seq64::event::event ( )
```



## 13.10.3.2 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.
------------	---

## 13.10.3.3 ~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.

## 13.10.4 Member Function Documentation

## 13.10.4.1 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.

**13.10.4.2 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.

**13.10.4.3 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.
-------------	---

## 13.10.4.4 get\_timestamp()

```
midipulse seq64::event::get_timestamp ( ) const [inline]
```

## 13.10.4.5 get\_channel()

```
midibyte seq64::event::get_channel ( ) const [inline]
```

## 13.10.4.6 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.

## 13.10.4.7 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.

#### 13.10.4.8 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.

#### 13.10.4.9 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.

## 13.10.4.10 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.

## 13.10.4.11 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.

## 13.10.4.12 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
```

```

\param m
    The channel status or message byte to be tested, with the channel
    bits masked off.

\param cc
    The desired cc value, which the datum must match, if the message is
    a control-change message.

\param datum
    The current datum, to be compared to cc, if the message is a
    control-change message.

\return
    Returns true if the message is not a control-change, or if it is
    and the cc and datum parameters match.

```

#### 13.10.4.13 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.

#### 13.10.4.14 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 `EVE<←` NT\_CLEAR\_CHAN\_MASK.

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.

## 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.
---------------	--

13.10.4.15 `set_status()` [2/2]

```
void seq64::event::set_status (
    midibyte eventcode,
    midibyte channel )
```

## 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 assume to have already had its high nybble cleared by masking against <code>EVENT_GET_CHAN_MASK</code> .

13.10.4.16 `set_status_keep_channel()`

```
void seq64::event::set_status_keep_channel (
    midibyte eventcode ) [inline]
```

It replaces stazed's `set_status()` with the optional "record" parameter.

## Parameters

<i>eventcode</i>	The status byte, generally read from the MIDI buss.
------------------	---

13.10.4.17 `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.
----------------	-----------------------------

#### 13.10.4.18 get\_status()

```
midibyte seq64::event::get_status ( ) const [inline]
```

#### 13.10.4.19 non\_cc\_match()

```
bool seq64::event::non_cc_match (
    midibyte status ) [inline]
```

##### Parameters

<i>status</i>	The status to be checked.
---------------	---------------------------

#### 13.10.4.20 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.

#### 13.10.4.21 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.

##### Parameters

<i>d1</i>	The byte value to set as the first data byte.
-----------	---



**13.10.4.22 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.

**13.10.4.23 get\_data()**

```
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.

**13.10.4.24 increment\_data1()**

```
void seq64::event::increment_data1 ( ) [inline]
```

**13.10.4.25 decrement\_data1()**

```
void seq64::event::decrement_data1 ( ) [inline]
```

**13.10.4.26 increment\_data2()**

```
void seq64::event::increment_data2 ( ) [inline]
```

**13.10.4.27 decrement\_data2()**

```
void seq64::event::decrement_data2 ( ) [inline]
```

**13.10.4.28 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 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.

**13.10.4.29 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.
-------------	--

**13.10.4.30 restart\_sysex()**

```
void seq64::event::restart_sysex ( )
```

(The m\_sysex member used to be a pointer.)

**13.10.4.31 get\_sysex()** [1/2]

```
SysexContainer& seq64::event::get_sysex ( ) [inline]
```

**13.10.4.32** get\_sysex() [2/2]

```
const SysexContainer& seq64::event::get_sysex ( ) const [inline]
```

**13.10.4.33** set\_sysex\_size()

```
void seq64::event::set_sysex_size (
    int len ) [inline]
```

**13.10.4.34** get\_sysex\_size()

```
int seq64::event::get_sysex_size ( ) const [inline]
```

**13.10.4.35** 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 <a href="#">is_linked()</a> is correct.
----	---

**13.10.4.36** get\_linked()

```
event* seq64::event::get_linked ( ) const [inline]
```

**13.10.4.37** is\_linked()

```
bool seq64::event::is_linked ( ) const [inline]
```

**13.10.4.38** clear\_link()

```
void seq64::event::clear_link ( ) [inline]
```

**13.10.4.39 paint()**

```
void seq64::event::paint ( ) [inline]
```

**13.10.4.40 unpaint()**

```
void seq64::event::unpaint ( ) [inline]
```

**13.10.4.41 is\_painted()**

```
bool seq64::event::is_painted ( ) const [inline]
```

**13.10.4.42 mark()**

```
void seq64::event::mark ( ) [inline]
```

**13.10.4.43 unmark()**

```
void seq64::event::unmark ( ) [inline]
```

**13.10.4.44 is\_marked()**

```
bool seq64::event::is_marked ( ) const [inline]
```

**13.10.4.45 select()**

```
void seq64::event::select ( ) [inline]
```

**13.10.4.46 unselect()**

```
void seq64::event::unselect ( ) [inline]
```

**13.10.4.47 is\_selected()**

```
bool seq64::event::is_selected ( ) const [inline]
```

**13.10.4.48 make\_clock()**

```
void seq64::event::make_clock ( ) [inline]
```

**13.10.4.49 data()**

```
midibyte seq64::event::data (
    int index ) const [inline]
```

**13.10.4.50 get\_note()**

```
midibyte seq64::event::get_note ( ) const [inline]
```

**13.10.4.51 set\_note()**

```
void seq64::event::set_note (
    midibyte note ) [inline]
```

**Parameters**

<i>note</i>	Provides the note value to set.
-------------	---------------------------------

**13.10.4.52 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.
-----------	---

#### 13.10.4.53 get\_note\_velocity()

```
midibyte seq64::event::get_note_velocity ( ) const [inline]
```

#### 13.10.4.54 set\_note\_velocity()

```
void seq64::event::set_note_velocity (
    int vel ) [inline]
```

##### Parameters

<i>vel</i>	Provides the velocity value to set.
------------	-------------------------------------

#### 13.10.4.55 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.

#### 13.10.4.56 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.

#### 13.10.4.57 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.

#### 13.10.4.58 is\_note\_off\_recorded()

```
bool seq64::event::is_note_off_recorded ( ) const [inline]
```

The channel nybble is masked off before the test.

##### Returns

Returns true if the event is a Note On event with velocity of 0.

#### 13.10.4.59 print()

```
void seq64::event::print ( ) const
```

#### 13.10.4.60 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.

### 13.10.5 Field Documentation

#### 13.10.5.1 m\_timestamp

```
midipulse seq64::event::m_timestamp [private]
```

#### 13.10.5.2 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 nibble = type of event; The low nibble = channel. Bit 7 is present in all status bytes.

#### 13.10.5.3 m\_channel

```
midibyte seq64::event::m_channel [private]
```

This member adds another 4 bytes to the event object, most likely.

#### 13.10.5.4 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.

#### 13.10.5.5 m\_sysex

```
SysexContainer seq64::event::m_sysex [private]
```

Adapted from Stazed's Seq32 project on GitHub.

#### 13.10.5.6 m\_sysex\_size

```
int seq64::event::m_sysex_size [private]
```

Perhaps redundant.

#### 13.10.5.7 m\_linked

```
event* seq64::event::m_linked [private]
```



#### 13.10.5.8 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.

#### 13.10.5.9 m\_selected

```
bool seq64::event::m_selected [private]
```

#### 13.10.5.10 m\_marked

```
bool seq64::event::m_marked [private]
```

#### 13.10.5.11 m\_painted

```
bool seq64::event::m_painted [private]
```

## 13.11 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.*

#### 13.11.1 Detailed Description

Its types match the `m_timestamp` and `get_rank()` function of this event class.

## 13.11.2 Constructor & Destructor Documentation

### 13.11.2.1 `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 <a href="#">event::get_rank()</a> function for more information.

### 13.11.2.2 `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.
------------	--------------------------------------

## 13.11.3 Member Function Documentation

### 13.11.3.1 `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.

### 13.11.4 Field Documentation

#### 13.11.4.1 m\_timestamp

`midipulse seq64::event_list::event_key::m_timestamp [private]`

#### 13.11.4.2 m\_rank

`int seq64::event_list::event_key::m_rank [private]`

## 13.12 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 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.*
- `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` &e)
  - The multimap version of this function does nothing.
- bool `is_modified` () const
  - 'Getter' function for member `m_is_modified`
- 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` ()
  - TEMPORARILY HERE for gdb.

## 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 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 std::multimap< `event_key`, `event` >::iterator `iterator`
- typedef std::multimap< `event_key`, `event` >::const\_iterator `const_iterator`

## 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.
- 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 new flag to indicate if an event was added or removed.*

### Friends

- class [editable\\_events](#)
- class [midifile](#)
- class [midi\\_container](#)
- class [midi\\_splitter](#)
- class [sequence](#)

## 13.12.1 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.

## 13.12.2 Member Typedef Documentation

### 13.12.2.1 Events

```
typedef std::multimap<event_key, event> seq64::event_list::Events [private]
```

### 13.12.2.2 EventsPair

```
typedef std::pair<event_key, event> seq64::event_list::EventsPair [private]
```

### 13.12.2.3 iterator

```
typedef std::multimap<event_key, event>::iterator seq64::event_list::iterator [private]
```

### 13.12.2.4 const\_iterator

```
typedef std::multimap<event_key, event>::const_iterator seq64::event_list::const_iterator  
[private]
```

## 13.12.3 Constructor & Destructor Documentation

### 13.12.3.1 event\_list() [1/2]

```
seq64::event_list::event_list ( )
```

### 13.12.3.2 event\_list() [2/2]

```
seq64::event_list::event_list (
    const event_list & rhs )
```

#### Parameters

<i>rhs</i>	Provides the event list to be copied.
------------	---------------------------------------

### 13.12.3.3 ~event\_list()

```
seq64::event_list::~~event_list ( )
```

## 13.12.4 Member Function Documentation

## 13.12.4.1 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.
------------	---

## 13.12.4.2 begin() [1/2]

```
iterator seq64::event_list::begin ( ) [inline]
```

## 13.12.4.3 begin() [2/2]

```
const_iterator seq64::event_list::begin ( ) const [inline]
```

## 13.12.4.4 end() [1/2]

```
iterator seq64::event_list::end ( ) [inline]
```

## 13.12.4.5 end() [2/2]

```
const_iterator seq64::event_list::end ( ) const [inline]
```

## 13.12.4.6 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.

## 13.12.4.7 empty()

```
bool seq64::event_list::empty ( ) const [inline]
```

```
return m_events.size() == 0;
```

## 13.12.4.8 add()

```
bool seq64::event_list::add (
    const event & e ) [inline]
```

## Parameters

<i>e</i>	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.

13.12.4.9 `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 sorted after the addition. This is a time-consuming operation.

## 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

<i>e</i>	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.

13.12.4.10 `push_back()`

```
void seq64::event_list::push_back (
    const event & ) [inline]
```



#### 13.12.4.11 is\_modified()

```
bool seq64::event_list::is_modified ( ) const [inline]
```

#### 13.12.4.12 unmodify()

```
void seq64::event_list::unmodify ( ) [inline]
```

But use it with great caution.

#### 13.12.4.13 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.
-----------	---

#### 13.12.4.14 clear()

```
void seq64::event_list::clear ( ) [inline]
```

Sets the modified-flag.

#### 13.12.4.15 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 multi-map. 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.

#### 13.12.4.16 `sort()`

```
void seq64::event_list::sort ( )
```

#### 13.12.4.17 `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.
-----------	---

#### 13.12.4.18 `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.
-----------	---

#### 13.12.4.19 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.

#### 13.12.4.20 clear\_links()

```
void seq64::event_list::clear_links ( ) [private]
```

#### 13.12.4.21 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.
----------------	---

#### 13.12.4.22 mark\_selected()

```
bool seq64::event_list::mark_selected ( ) [private]
```

##### Returns

Returns true if there was even one event selected and marked.

#### 13.12.4.23 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.
----------------	---

**13.12.4.24 mark\_all()**

```
void seq64::event_list::mark_all ( ) [private]
```

Not yet used, but might come in handy with the event editor dialog.

**13.12.4.25 unmark\_all()**

```
void seq64::event_list::unmark_all ( ) [private]
```

**13.12.4.26 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.

**13.12.4.27 unpaint\_all()**

```
void seq64::event_list::unpaint_all ( ) [private]
```

**13.12.4.28 count\_selected\_notes()**

```
int seq64::event_list::count_selected_notes ( ) const [private]
```

#### 13.12.4.29 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. We could add a flag to [count\\_selected\\_notes\(\)](#) to break, I suppose.

##### Returns

Returns true if at least one note is selected.

#### 13.12.4.30 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.

##### 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.

#### 13.12.4.31 select\_all()

```
void seq64::event_list::select_all ( ) [private]
```

#### 13.12.4.32 unselect\_all()

```
void seq64::event_list::unselect_all ( ) [private]
```

#### 13.12.4.33 print()

```
void seq64::event_list::print ( ) const [private]
```

#### 13.12.4.34 events()

```
const Events& seq64::event_list::events ( ) const [inline], [private]
```

### 13.12.5 Friends And Related Function Documentation

#### 13.12.5.1 editable\_events

```
friend class editable_events [friend]
```

#### 13.12.5.2 midifile

```
friend class midifile [friend]
```

#### 13.12.5.3 midi\_container

```
friend class midi_container [friend]
```

#### 13.12.5.4 midi\_splitter

```
friend class midi_splitter [friend]
```

#### 13.12.5.5 sequence

```
friend class sequence [friend]
```

### 13.12.6 Field Documentation

#### 13.12.6.1 m\_events

```
Events seq64::event_list::m_events [private]
```

## 13.12.6.2 m\_is\_modified

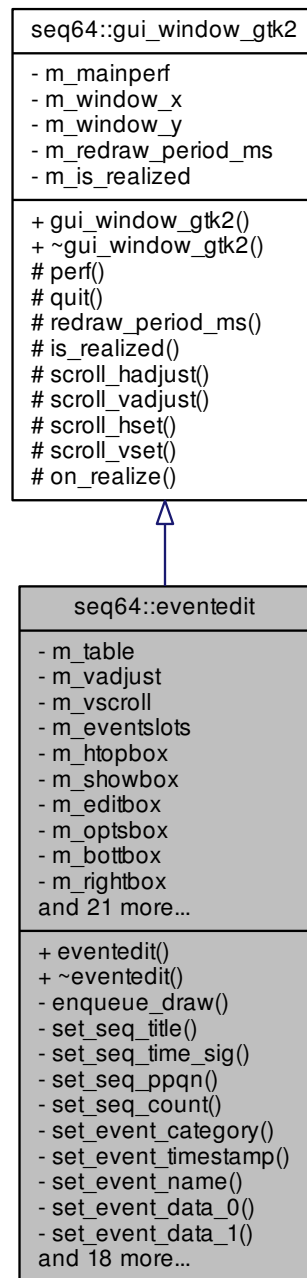
```
bool seq64::event_list::m_is_modified [private]
```

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.

## 13.13 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\\_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\\_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

### 13.13.1 Constructor & Destructor Documentation

#### 13.13.1.1 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
-----
htop | (OLD LAYOUT)                :  :                      | 0
-----
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
| - - - - - - - - - - - - - - - | c | ----- 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?   ] | |
| ...      ...      ...      | | ----- optsbox ----- | | 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 experimental 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.

#### 13.13.1.2 ~eventedit()

```
seq64::eventedit::~~eventedit ( ) [virtual]
```

We're going to have to run the application through `valgrind` to make sure that nothing is left behind.

## 13.13.2 Member Function Documentation

### 13.13.2.1 enqueue\_draw()

```
void seq64::eventedit::enqueue_draw ( ) [private]
```

### 13.13.2.2 set\_seq\_title()

```
void seq64::eventedit::set_seq_title (
    const std::string & title ) [private]
```

#### Parameters

<i>title</i>	The name of the sequence.
--------------	---------------------------

### 13.13.2.3 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.
------------	-------------------------------------

### 13.13.2.4 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.
----------	---

### 13.13.2.5 set\_seq\_count()

```
void seq64::eventedit::set_seq_count ( ) [private]
```

#### 13.13.2.6 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.
----------	--

#### 13.13.2.7 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.
-----------	--

#### 13.13.2.8 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.
----------	---

#### 13.13.2.9 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.
----------	---

#### 13.13.2.10 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.
----------	--

#### 13.13.2.11 perf\_modify()

```
void seq64::eventedit::perf_modify ( ) [private]
```

#### 13.13.2.12 set\_dirty()

```
void seq64::eventedit::set_dirty (
    bool flag = true ) [private]
```

This includes changing a label and enabling/disabling the Save button.

##### Parameters

<i>flag</i>	If true, the modified status is indicated, otherwise it is cleared.
-------------	---

#### 13.13.2.13 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.
--------------	--

#### 13.13.2.14 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.

#### 13.13.2.15 change\_focus()

```
void seq64::eventedit::change_focus (
    bool set_it = true ) [private]
```

Similar to the same function in seqedit.

#### Parameters

<i>set_it</i>	If true (the default value), indicates we want focus, otherwise we want to give up focus.
---------------	---

#### 13.13.2.16 close\_out()

```
void seq64::eventedit::close_out ( ) [private]
```

#### 13.13.2.17 handle\_close()

```
void seq64::eventedit::handle_close ( ) [private]
```

Simply calls [close\\_out\(\)](#).

#### 13.13.2.18 handle\_delete()

```
void seq64::eventedit::handle_delete ( ) [private]
```



#### 13.13.2.19 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\(\)](#).

#### 13.13.2.20 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\(\)](#).

#### 13.13.2.21 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.

#### 13.13.2.22 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.

#### 13.13.2.23 on\_realize()

```
void seq64::eventedit::on_realize ( ) [private]
```

Then it sets the vertical adjustment to account for the number of events in the eventslot.

#### 13.13.2.24 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 <a href="#">gui_window_gtk2</a> 's version of this function.
--------------	---

13.13.2.25 `on_focus_in_event()`

```
bool seq64::eventedit::on_focus_in_event (
    GdkEventFocus * ) [private]
```

It sets the focus flag and calls [change\\_focus\(\)](#).

13.13.2.26 `on_focus_out_event()`

```
bool seq64::eventedit::on_focus_out_event (
    GdkEventFocus * ) [private]
```

It resets the focus flag and calls [change\\_focus\(\)](#).

13.13.2.27 `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.

13.13.2.28 `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.

### 13.13.3 Friends And Related Function Documentation

#### 13.13.3.1 eventslots

```
friend class eventslots [friend]
```

### 13.13.4 Field Documentation

#### 13.13.4.1 m\_table

```
Gtk::Table* seq64::eventedit::m_table [private]
```

Provides the layout table for UI.

#### 13.13.4.2 m\_vadjust

```
Gtk::Adjustment* seq64::eventedit::m_vadjust [private]
```

#### 13.13.4.3 m\_vscroll

```
Gtk::VScrollbar* seq64::eventedit::m_vscroll [private]
```

#### 13.13.4.4 m\_eventslots

```
eventslots* seq64::eventedit::m_eventslots [private]
```

#### 13.13.4.5 m\_htopbox

Gtk::HBox\* seq64::eventedit::m\_htopbox [private]

#### 13.13.4.6 m\_showbox

Gtk::VBox\* seq64::eventedit::m\_showbox [private]

#### 13.13.4.7 m\_editbox

Gtk::VBox\* seq64::eventedit::m\_editbox [private]

#### 13.13.4.8 m\_optsbox

Gtk::VBox\* seq64::eventedit::m\_optsbox [private]

#### 13.13.4.9 m\_bottbox

Gtk::HBox\* seq64::eventedit::m\_bottbox [private]

#### 13.13.4.10 m\_rightbox

Gtk::VBox\* seq64::eventedit::m\_rightbox [private]

#### 13.13.4.11 m\_button\_del

Gtk::Button\* seq64::eventedit::m\_button\_del [private]

#### 13.13.4.12 m\_button\_ins

Gtk::Button\* seq64::eventedit::m\_button\_ins [private]

**13.13.4.13 m\_button\_modify**

Gtk::Button\* seq64::eventedit::m\_button\_modify [private]

**13.13.4.14 m\_button\_save**

Gtk::Button\* seq64::eventedit::m\_button\_save [private]

**13.13.4.15 m\_button\_cancel**

Gtk::Button\* seq64::eventedit::m\_button\_cancel [private]

**13.13.4.16 m\_label\_seq\_name**

Gtk::Label\* seq64::eventedit::m\_label\_seq\_name [private]

Shows the name of the pattern.

**13.13.4.17 m\_label\_time\_sig**

Gtk::Label\* seq64::eventedit::m\_label\_time\_sig [private]

**13.13.4.18 m\_label\_ppqn**

Gtk::Label\* seq64::eventedit::m\_label\_ppqn [private]

**13.13.4.19 m\_label\_channel**

Gtk::Label\* seq64::eventedit::m\_label\_channel [private]

**13.13.4.20 m\_label\_ev\_count**

Gtk::Label\* seq64::eventedit::m\_label\_ev\_count [private]

**13.13.4.21 m\_label\_spacer**

```
Gtk::Label* seq64::eventedit::m_label_spacer [private]
```

**13.13.4.22 m\_label\_modified**

```
Gtk::Label* seq64::eventedit::m_label_modified [private]
```

**13.13.4.23 m\_label\_category**

```
Gtk::Label* seq64::eventedit::m_label_category [private]
```

Shows the type of MIDI event.

**13.13.4.24 m\_entry\_ev\_timestamp**

```
Gtk::Entry* seq64::eventedit::m_entry_ev_timestamp [private]
```

**13.13.4.25 m\_entry\_ev\_name**

```
Gtk::Entry* seq64::eventedit::m_entry_ev_name [private]
```

**13.13.4.26 m\_entry\_ev\_data\_0**

```
Gtk::Entry* seq64::eventedit::m_entry_ev_data_0 [private]
```

**13.13.4.27 m\_entry\_ev\_data\_1**

```
Gtk::Entry* seq64::eventedit::m_entry_ev_data_1 [private]
```

**13.13.4.28 m\_label\_time\_fmt**

```
Gtk::Label* seq64::eventedit::m_label_time_fmt [private]
```

#### 13.13.4.29 m\_label\_right

```
Gtk::Label* seq64::eventedit::m_label_right [private]
```

#### 13.13.4.30 m\_seq

```
sequence& seq64::eventedit::m_seq [private]
```

#### 13.13.4.31 m\_have\_focus

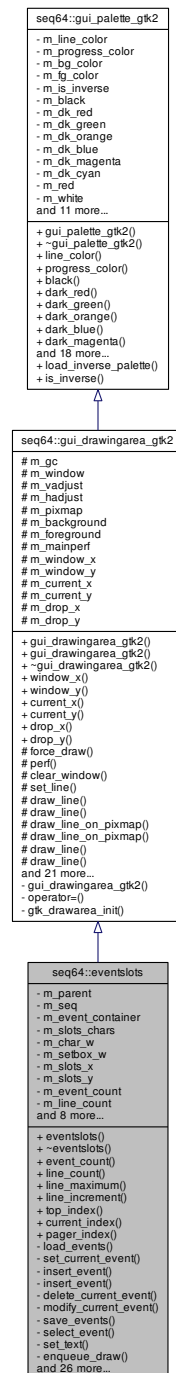
```
bool seq64::eventedit::m_have_focus [private]
```

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.

## 13.14 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.*

- 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 `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

- 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.*
- 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.*
- [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

### 13.14.1 Constructor & Destructor Documentation

#### 13.14.1.1 eventslots()

```
seq64::eventslots::eventslots (
    perform & p,
    eventedit & parent,
    sequence & seq,
    Gtk::Adjustment & vadjust )
```

#### 13.14.1.2 ~eventslots()

```
virtual seq64::eventslots::~~eventslots ( ) [inline], [virtual]
```

### 13.14.2 Member Function Documentation

#### 13.14.2.1 event\_count()

```
int seq64::eventslots::event_count ( ) const [inline]
```

#### 13.14.2.2 line\_count()

```
int seq64::eventslots::line_count ( ) const [inline]
```

#### 13.14.2.3 line\_maximum()

```
int seq64::eventslots::line_maximum ( ) const [inline]
```

#### 13.14.2.4 line\_increment()

```
int seq64::eventslots::line_increment ( ) const [inline]
```

#### 13.14.2.5 top\_index()

```
int seq64::eventslots::top_index ( ) const [inline]
```

**13.14.2.6** current\_index()

```
int seq64::eventslots::current_index ( ) const [inline]
```

**13.14.2.7** pager\_index()

```
int seq64::eventslots::pager_index ( ) const [inline]
```

**13.14.2.8** 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.

**13.14.2.9** set\_current\_event()

```
void seq64::eventslots::set_current_event (
    const editable_events::iterator ei,
    int index,
    bool full_redraw = true ) [private]
```

Note in the sprintf() 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.

**13.14.2.10** 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.

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.

#### 13.14.2.11 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.

### 13.14.2.12 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.

### 13.14.2.13 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.

#### 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.

### 13.14.2.14 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.

### 13.14.2.15 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 SEQ64_NULL_EVENT 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 frame under fast movement.

## 13.14.2.16 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.

## 13.14.2.17 enqueue\_draw()

```
void seq64::eventslots::enqueue_draw ( ) [private]
```

## 13.14.2.18 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.

**13.14.2.19 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.

**13.14.2.20 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 = ???
```

13.14.2.21 `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.

13.14.2.22 `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.
------------------	---

13.14.2.23 `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.
-------------------	--

#### 13.14.2.24 decrement\_top()

```
int seq64::eventslots::decrement_top ( ) [private]
```

##### Returns

Returns 0, or SEQ64\_NULL\_EVENT\_INDEX if the iterator could not be decremented.

#### 13.14.2.25 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.

#### 13.14.2.26 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.

#### 13.14.2.27 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.

#### 13.14.2.28 decrement\_bottom()

```
int seq64::eventslots::decrement_bottom ( ) [private]
```

##### Returns

Returns 0, or SEQ64\_NULL\_EVENT\_INDEX if the iterator could not be decremented.

#### 13.14.2.29 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.

#### 13.14.2.30 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.

#### 13.14.2.31 on\_expose\_event()

```
bool seq64::eventslots::on_expose_event (
    GdkEventExpose * ev ) [private]
```

It draws all of the sequences.

#### 13.14.2.32 on\_button\_press\_event()

```
bool seq64::eventslots::on_button_press_event (
    GdkEventButton * ev ) [private]
```

#### 13.14.2.33 on\_button\_release\_event()

```
bool seq64::eventslots::on_button_release_event (
    GdkEventButton * ev ) [private]
```

#### 13.14.2.34 on\_focus\_in\_event()

```
bool seq64::eventslots::on_focus_in_event (
    GdkEventFocus * ev ) [private]
```

See the same function in the perfrill module.

#### 13.14.2.35 on\_focus\_out\_event()

```
bool seq64::eventslots::on_focus_out_event (
    GdkEventFocus * ev ) [private]
```

#### 13.14.2.36 on\_scroll\_event()

```
bool seq64::eventslots::on_scroll_event (
    GdkEventScroll * ev ) [private]
```

#### 13.14.2.37 on\_size\_allocate()

```
void seq64::eventslots::on_size_allocate (
    Gtk::Allocation & a ) [private]
```

It first calls the base-class version of this function.

#### 13.14.2.38 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\(\)](#).

#### 13.14.2.39 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\(\)](#).

#### 13.14.2.40 on\_frame\_up()

```
void seq64::eventslots::on_frame_up ( ) [private]
```

#### 13.14.2.41 on\_frame\_down()

```
void seq64::eventslots::on_frame_down ( ) [private]
```

#### 13.14.2.42 on\_frame\_home()

```
void seq64::eventslots::on_frame_home ( ) [private]
```

#### 13.14.2.43 on\_frame\_end()

```
void seq64::eventslots::on_frame_end ( ) [private]
```

### 13.14.3 Friends And Related Function Documentation

#### 13.14.3.1 eventedit

```
friend class eventedit [friend]
```

### 13.14.4 Field Documentation

#### 13.14.4.1 m\_parent

```
eventedit& seq64::eventslots::m_parent [private]
```

#### 13.14.4.2 m\_seq

```
sequence& seq64::eventslots::m_seq [private]
```

#### 13.14.4.3 m\_event\_container

```
editable_events seq64::eventslots::m_event_container [private]
```

#### 13.14.4.4 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).

#### 13.14.4.5 m\_char\_w

```
int seq64::eventslots::m_char_w [private]
```

This value is obtained from a font-renderer accessor function.

#### 13.14.4.6 m\_setbox\_w

```
int seq64::eventslots::m_setbox_w [private]
```

This used to be hardwired to  $6 * 2$  (character-width times two).

#### 13.14.4.7 m\_slots\_x

```
int seq64::eventslots::m_slots_x [private]
```

#### 13.14.4.8 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.

#### 13.14.4.9 m\_event\_count

```
int seq64::eventslots::m_event_count [private]
```

#### 13.14.4.10 m\_line\_count

```
int seq64::eventslots::m_line_count [private]
```



#### 13.14.4.11 m\_line\_maximum

```
int seq64::eventslots::m_line_maximum [private]
```

#### 13.14.4.12 m\_line\_overlap

```
int seq64::eventslots::m_line_overlap [private]
```

#### 13.14.4.13 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.

#### 13.14.4.14 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.

#### 13.14.4.15 m\_top\_iterator

```
editable_events::iterator seq64::eventslots::m_top_iterator [private]
```

#### 13.14.4.16 m\_bottom\_iterator

```
editable_events::iterator seq64::eventslots::m_bottom_iterator [private]
```

#### 13.14.4.17 m\_current\_iterator

```
editable_events::iterator seq64::eventslots::m_current_iterator [private]
```

## 13.14.4.18 m\_pager\_index

```
int seq64::eventslots::m_pager_index [private]
```

## 13.15 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](#) ()  
*Rotate 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.*

## 13.15.1 Member Enumeration Documentation

### 13.15.1.1 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.

## 13.15.2 Constructor & Destructor Documentation

### 13.15.2.1 font()

```
seq64::font::font ( )
```

## 13.15.3 Member Function Documentation

### 13.15.3.1 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.

#### Parameters

<i>wp</i>	Provides the windows pointer for the window that holds the color map.
-----------	---

### 13.15.3.2 render\_string\_on\_drawable()

```
void seq64::font::render_string_on_drawable (
    Glib::RefPtr< Gdk::GC > gc,
    int x,
    int y,
    Glib::RefPtr< Gdk::Drawable > a_draw,
    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

<i>gc</i>	Provides the graphics context for drawing the text using GTK+.
<i>x</i>	The horizontal location of the text.
<i>y</i>	The vertical location of the text.
<i>a_draw</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 <a href="#">font::BLACK</a> , <a href="#">font::WHITE</a> , <a href="#">font::BLACK_ON_YELLOW</a> , <a href="#">font::YELLOW_ON_BLACK</a> . The actual correct colors are provided by selecting one of four font pixmaps, as described in the <a href="#">init()</a> function.
<i>invert</i>	If true, apply color inversion, if specified.

### 13.15.3.3 char\_width()

```
int seq64::font::char_width ( ) const [inline]
```

### 13.15.3.4 char\_height()

```
int seq64::font::char_height ( ) const [inline]
```

### 13.15.3.5 padded\_height()

```
int seq64::font::padded_height ( ) const [inline]
```

## 13.15.4 Field Documentation

### 13.15.4.1 m\_use\_new\_font

```
bool seq64::font::m_use_new_font [private]
```

### 13.15.4.2 m\_cell\_w

```
int seq64::font::m_cell_w [private]
```

### 13.15.4.3 m\_cell\_h

```
int seq64::font::m_cell_h [private]
```

### 13.15.4.4 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.

**13.15.4.5 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.

**13.15.4.6 m\_offset**

```
int seq64::font::m_offset [private]
```

**13.15.4.7 m\_padded\_h**

```
int seq64::font::m_padded_h [private]
```

**13.15.4.8 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).

**13.15.4.9 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.

**13.15.4.10 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.

**13.15.4.11 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.

#### 13.15.4.12 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.

#### 13.15.4.13 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.

#### 13.15.4.14 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.

#### 13.15.4.15 m\_clip\_mask

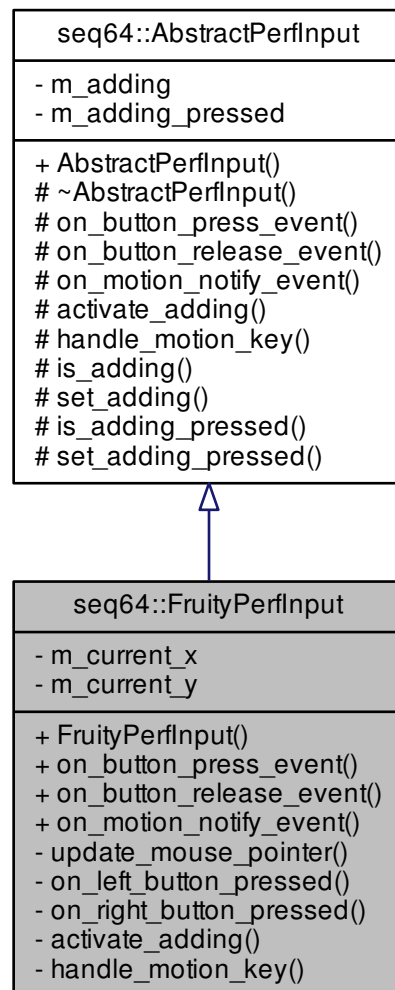
```
Glib::RefPtr<Gdk::Bitmap> seq64::font::m_clip_mask [private]
```

All we know is it seems to be a requirement for creating a pixmap object from an XMP file.

## 13.16 seq64::FruityPerfInput Class Reference

Implements the performance input of that certain fruity sequencer that people seem to like.

Inheritance diagram for seq64::FruityPerInput:



## Public Member Functions

- [FruityPerInput \(\)](#)  
*Default constructor.*
- bool [on\\_button\\_press\\_event](#) (GdkEventButton \*ev, [perffroll](#) &roll)  
*Handles a button-press event in the Fruity manner.*
- bool [on\\_button\\_release\\_event](#) (GdkEventButton \*ev, [perffroll](#) &roll)  
*Handles a button-release event.*
- bool [on\\_motion\\_notify\\_event](#) (GdkEventMotion \*ev, [perffroll](#) &roll)  
*Handles a Fruity motion-notify event.*



## Private Member Functions

- void [update\\_mouse\\_pointer](#) ([perffroll](#) &roll)  
*Updates the mouse pointer, implementing a context-sensitive mouse.*
- bool [on\\_left\\_button\\_pressed](#) (GdkEventButton \*ev, [perffroll](#) &roll)  
*Handles the left button of the mouse.*
- bool [on\\_right\\_button\\_pressed](#) (GdkEventButton \*ev, [perffroll](#) &roll)  
*Handles the right button of the mouse.*
- virtual void [activate\\_adding](#) (bool, [perffroll](#) &)
- virtual bool [handle\\_motion\\_key](#) (bool, [perffroll](#) &)

## Private Attributes

- long [m\\_current\\_x](#)  
*The current x value of the mouse.*
- long [m\\_current\\_y](#)  
*The current y value of the mouse.*

## Friends

- class [perffroll](#)

## Additional Inherited Members

### 13.16.1 Constructor & Destructor Documentation

#### 13.16.1.1 FruityPerfInput()

```
seq64::FruityPerfInput::FruityPerfInput ( ) [inline]
```

### 13.16.2 Member Function Documentation

#### 13.16.2.1 on\_button\_press\_event()

```
bool seq64::FruityPerfInput::on_button_press_event (
    GdkEventButton * ev,
    perffroll & roll ) [virtual]
```

#### Parameters

<i>ev</i>	The button-press event to process.
<i>roll</i>	The song editor piano roll that is the "parent" of this class.

**Returns**

Returns true if a modification occurred.

Implements [seq64::AbstractPerfInput](#).

**13.16.2.2 on\_button\_release\_event()**

```
bool seq64::FruityPerfInput::on_button_release_event (
    GdkEventButton * ev,
    perfroll & roll ) [virtual]
```

Why is m\_adding\_pressed modified conditionally when the same modification is then made unconditionally?

**Parameters**

<i>ev</i>	The button-release event to process.
<i>roll</i>	The song editor piano roll that is the "parent" of this class.

**Returns**

Returns true if a modification occurred.

Implements [seq64::AbstractPerfInput](#).

**13.16.2.3 on\_motion\_notify\_event()**

```
bool seq64::FruityPerfInput::on_motion_notify_event (
    GdkEventMotion * ev,
    perfroll & roll ) [virtual]
```

**Parameters**

<i>ev</i>	The motion-notify event to process.
<i>roll</i>	The song editor piano roll that is the "parent" of this class.

**Returns**

Returns true if a modification occurred, and sets the perform modified flag based on that result.

Implements [seq64::AbstractPerfInput](#).

#### 13.16.2.4 update\_mouse\_pointer()

```
void seq64::FruityPerfInput::update_mouse_pointer (
    perffroll & roll ) [private]
```

Note that `perform::convert_xy()` returns its values via side-effects on the last two parameters.

##### Parameters

<i>roll</i>	The song editor piano roll that is the "parent" of this class.
-------------	--

#### 13.16.2.5 on\_left\_button\_pressed()

```
bool seq64::FruityPerfInput::on_left_button_pressed (
    GdkEventButton * ev,
    perffroll & roll ) [private]
```

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.
<i>roll</i>	The song editor piano roll that is the "parent" of this class.

##### Returns

Now returns true if a modification occurred.

#### 13.16.2.6 on\_right\_button\_pressed()

```
bool seq64::FruityPerfInput::on_right_button_pressed (
    GdkEventButton * ev,
    perffroll & roll ) [private]
```

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.
<i>roll</i>	The song editor piano roll that is the "parent" of this class.

**Returns**

Returns true if a modification occurred.

**13.16.2.7 activate\_adding()**

```
virtual void seq64::FruityPerfInput::activate_adding (
    bool ,
    perfroll & ) [inline], [private], [virtual]
```

Implements [seq64::AbstractPerfInput](#).

**13.16.2.8 handle\_motion\_key()**

```
virtual bool seq64::FruityPerfInput::handle_motion_key (
    bool ,
    perfroll & ) [inline], [private], [virtual]
```

Implements [seq64::AbstractPerfInput](#).

**13.16.3 Friends And Related Function Documentation****13.16.3.1 perfroll**

```
friend class perfroll [friend]
```

**13.16.4 Field Documentation****13.16.4.1 m\_current\_x**

```
long seq64::FruityPerfInput::m_current_x [private]
```

## 13.16.4.2 m\_current\_y

```
long seq64::FruityPerfInput::m_current_y [private]
```

## 13.17 seq64::FruitySeqEventInput Struct Reference

This structure implements the interaction methods for the "fruity" mode of operation.

### Public Member Functions

- [FruitySeqEventInput \(\)](#)  
*Default constructor.*
- void [update\\_mouse\\_pointer](#) (segevent &ths)  
*Provides support for a context-sensitive mouse.*
- bool [on\\_button\\_press\\_event](#) (GdkEventButton \*ev, [segevent](#) &ths)  
*Implements the on-button-press event callback.*
- bool [on\\_button\\_release\\_event](#) (GdkEventButton \*ev, [segevent](#) &ths)  
*Implements the on-button-release callback.*
- bool [on\\_motion\\_notify\\_event](#) (GdkEventMotion \*ev, [segevent](#) &ths)  
*Implements the on-motion-notify callback.*

### Data Fields

- 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.*

## 13.17.1 Constructor & Destructor Documentation

### 13.17.1.1 FruitySeqEventInput()

```
seq64::FruitySeqEventInput::FruitySeqEventInput ( ) [inline]
```

## 13.17.2 Member Function Documentation

### 13.17.2.1 update\_mouse\_pointer()

```
void seq64::FruitySeqEventInput::update_mouse_pointer (
    segevent & segev )
```

## Parameters

<i>segev</i>	Provides the sequevent pane (actually a strip on the seqedit window) to update to show the proper mouse cursor (left pointer, center pointer, and pencil).
--------------	--

## 13.17.2.2 on\_button\_press\_event()

```
bool seq64::FruitySeqEventInput::on_button_press_event (
    GdkEventButton * ev,
    sequevent & segev )
```

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.
<i>segev</i>	Provides the sequevent strip to be affected by this button event.

## Returns

Returns true if a modification was made. It used to return true all the time.

### 13.17.2.3 on\_button\_release\_event()

```
bool seq64::FruitySeqEventInput::on_button_release_event (
    GdkEventButton * ev,
    seqevent & segev )
```

#### Parameters

<i>ev</i>	The button event for the press of a mouse button.
<i>segev</i>	Provides the seqevent strip to be affected by this button event.

#### Returns

Returns true if a modification was made. It used to return true all the time.

### 13.17.2.4 on\_motion\_notify\_event()

```
bool seq64::FruitySeqEventInput::on_motion_notify_event (
    GdkEventMotion * ev,
    seqevent & segev )
```

#### Parameters

<i>ev</i>	The button event for the press of a mouse button.
<i>segev</i>	Provides the seqevent strip to be affected by this button event.

#### Returns

Returns true if a modification occurred, and sets the perform modified flag based on that result.

## 13.17.3 Field Documentation

### 13.17.3.1 m\_justselected\_one

```
bool seq64::FruitySeqEventInput::m_justselected_one
```

### 13.17.3.2 m\_is\_drag\_pasting\_start

```
bool seq64::FruitySeqEventInput::m_is_drag_pasting_start
```

### 13.17.3.3 m\_is\_drag\_pasting

```
bool seq64::FruitySeqEventInput::m_is_drag_pasting
```

## 13.18 seq64::FruitySeqRollInput Class Reference

Implements the fruity mouse interaction paradigm for the seqroll.

### Public Member Functions

- [FruitySeqRollInput \(\)](#)  
*Default constructor.*
- void [update\\_mouse\\_pointer](#) (seqroll &ths)  
*Updates the mouse pointer, implementing a context-sensitive mouse.*
- bool [on\\_button\\_press\\_event](#) (GdkEventButton \*ev, [seqroll](#) &ths)  
*Implements the fruity on-button-press callback.*
- bool [on\\_button\\_release\\_event](#) (GdkEventButton \*ev, [seqroll](#) &ths)  
*Implements the fruity handling for the on-button-release event.*
- bool [on\\_motion\\_notify\\_event](#) (GdkEventMotion \*ev, [seqroll](#) &ths)  
*Implements the fruity handling for the on-motion-notify event.*

### Private Attributes

- 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.*

## 13.18.1 Constructor & Destructor Documentation

### 13.18.1.1 FruitySeqRollInput()

```
seq64::FruitySeqRollInput::FruitySeqRollInput ( ) [inline]
```

## 13.18.2 Member Function Documentation

### 13.18.2.1 update\_mouse\_pointer()

```
void seq64::FruitySeqRollInput::update_mouse_pointer (
    seqroll & scroll )
```



## Parameters

<i>scroll</i>	Provides the "parent" of this interaction class.
---------------	--

## 13.18.2.2 on\_button\_press\_event()

```
bool seq64::FruitySeqRollInput::on_button_press_event (
    GdkEventButton * ev,
    seqroll & scroll )
```

This function now uses the needs\_update flag to determine if the perform object should modify().

## Parameters

<i>ev</i>	The button event.
<i>scroll</i>	The parent of this "fruity" interaction class.

## Returns

Returns the value of needs\_update. It used to return only true.

## 13.18.2.3 on\_button\_release\_event()

```
bool seq64::FruitySeqRollInput::on_button_release_event (
    GdkEventButton * ev,
    seqroll & scroll )
```

## Parameters

<i>ev</i>	The button event.
<i>scroll</i>	The parent of this "fruity" interaction class.

## 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.

## 13.18.2.4 on\_motion\_notify\_event()

```
bool seq64::FruitySeqRollInput::on_motion_notify_event (
    GdkEventMotion * ev,
    seqroll & scroll )
```

**Parameters**

<i>ev</i>	The motion event.
<i>sroll</i>	The parent of this "fruity" interaction class. (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 interation, in the Options menu, on the fly.)

**Returns**

Returns the value of needs\_update.

In "fruity" interatction 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.

**13.18.3 Field Documentation****13.18.3.1 m\_erase\_painting**

```
bool seq64::FruitySeqRollInput::m_erase_painting [private]
```

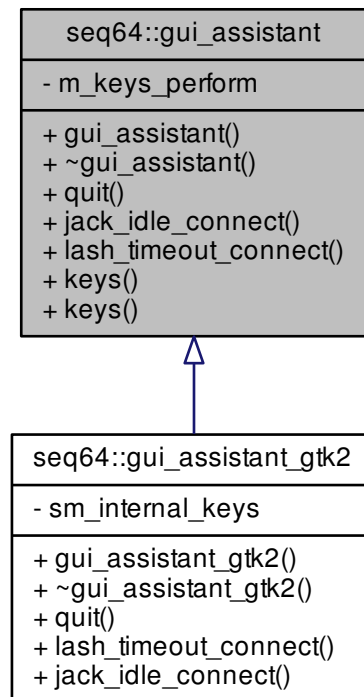
**13.18.3.2 m\_drag\_paste\_start\_pos**

```
int seq64::FruitySeqRollInput::m_drag_paste_start_pos[2] [private]
```

**13.19 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 ()=0`
- `virtual void jack_idle_connect (jack_assistant &jack)=0`
- `virtual void lash_timeout_connect (lash *lashobject)=0`
- `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.*

### 13.19.1 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).

### 13.19.2 Constructor & Destructor Documentation

#### 13.19.2.1 `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.
-----------	---

#### 13.19.2.2 `~gui_assistant()`

```
virtual seq64::gui_assistant::~~gui_assistant ( ) [inline], [virtual]
```

### 13.19.3 Member Function Documentation

#### 13.19.3.1 `quit()`

```
virtual void seq64::gui_assistant::quit ( ) [pure virtual]
```

Implemented in [seq64::gui\\_assistant\\_gtk2](#).

#### 13.19.3.2 `jack_idle_connect()`

```
virtual void seq64::gui_assistant::jack_idle_connect (
    jack\_assistant & jack ) [pure virtual]
```

Implemented in [seq64::gui\\_assistant\\_gtk2](#).

### 13.19.3.3 lash\_timeout\_connect()

```
virtual void seq64::gui_assistant::lash_timeout_connect (
    lash * lashobject ) [pure virtual]
```

Implemented in [seq64::gui\\_assistant\\_gtk2](#).

### 13.19.3.4 keys() [1/2]

```
const keys_perform& seq64::gui_assistant::keys ( ) const [inline]
```

### 13.19.3.5 keys() [2/2]

```
keys_perform& seq64::gui_assistant::keys ( ) [inline]
```

## 13.19.4 Field Documentation

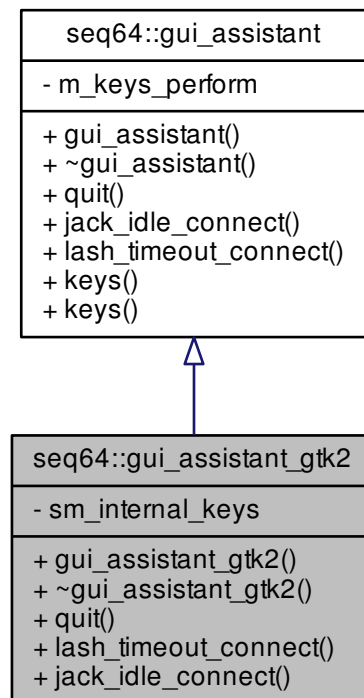
### 13.19.4.1 m\_keys\_perform

```
keys_perform& seq64::gui_assistant::m_keys_perform [private]
```

## 13.20 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.*

### 13.20.1 Constructor & Destructor Documentation

### 13.20.1.1 gui\_assistant\_gtk2()

```
seq64::gui_assistant_gtk2::gui_assistant_gtk2 ( )
```

### 13.20.1.2 ~gui\_assistant\_gtk2()

```
virtual seq64::gui_assistant_gtk2::~~gui_assistant_gtk2 ( ) [inline], [virtual]
```

## 13.20.2 Member Function Documentation

### 13.20.2.1 quit()

```
void seq64::gui_assistant_gtk2::quit ( ) [virtual]
```

Implements [seq64::gui\\_assistant](#).

### 13.20.2.2 lash\_timeout\_connect()

```
void seq64::gui_assistant_gtk2::lash_timeout_connect (
    lash * lashobject ) [virtual]
```

The time-out value is set to 250 ms.

Implements [seq64::gui\\_assistant](#).

### 13.20.2.3 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.

Implements [seq64::gui\\_assistant](#).

## 13.20.3 Field Documentation

### 13.20.3.1 sm\_internal\_keys

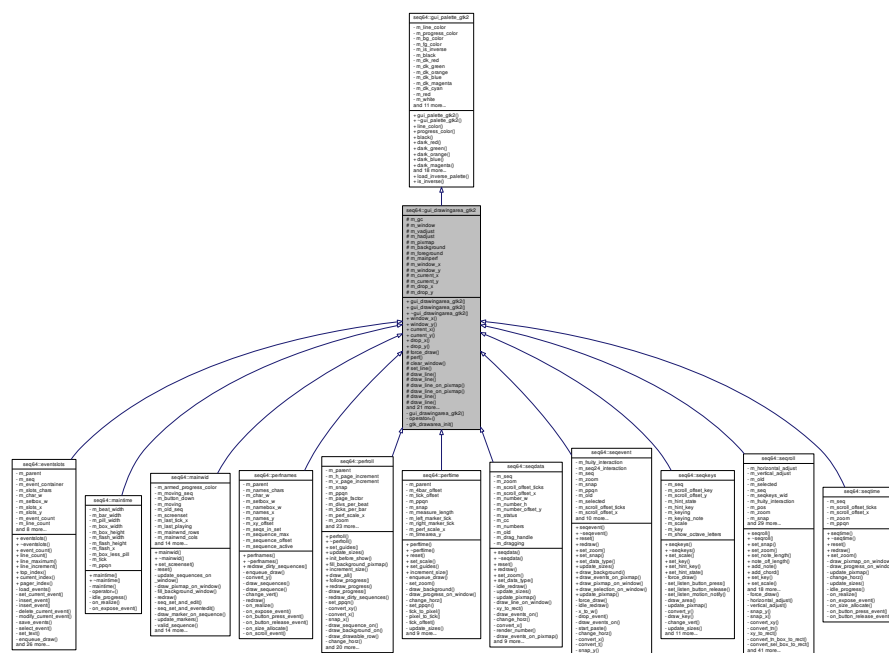
`keys_perform_gtk2` seq64::gui\_assistant\_gtk2::sm\_internal\_keys [static], [private]

This object is set into the reference provided in the `gui_assistant` base class.

## 13.21 seq64::gui\_drawingarea\_gtk2 Class Reference

Implements the basic drawing areas of the application.

Inheritance diagram for `seq64::gui_drawingarea_gtk2`:



## Data Structures

- struct `rect`

*A small helper structure representing a rectangle.*

## 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 [window\\_y](#) () const
- *'Getter' function for member m\_window\_y*
- 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*

## Protected Member Functions

- virtual void [force\\_draw](#) ()
- *Provides a common function for redrawing.*
- [perform](#) & [perf](#) ()
- *'Getter' function for member m\_mainperf*
- 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

### 13.21.1 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.

### 13.21.2 Constructor & Destructor Documentation

#### 13.21.2.1 `gui_drawingarea_gtk2()` [1/3]

```
seq64::gui_drawingarea_gtk2::gui_drawingarea_gtk2 (
    const gui\_drawingarea\_gtk2 & ) [private]
```

#### 13.21.2.2 `gui_drawingarea_gtk2()` [2/3]

```
seq64::gui_drawingarea_gtk2::gui_drawingarea_gtk2 (
    perform & p,
    int window_x = 0,
    int window_y = 0 )
```

### 13.21.2.3 `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 )
```

### 13.21.2.4 `~gui_drawingarea_gtk2()`

```
seq64::gui_drawingarea_gtk2::~~gui_drawingarea_gtk2 ( ) [virtual]
```

## 13.21.3 Member Function Documentation

### 13.21.3.1 `operator=()`

```
gui_drawingarea_gtk2& seq64::gui_drawingarea_gtk2::operator= (
    const gui_drawingarea_gtk2 & ) [private]
```

### 13.21.3.2 `window_x()`

```
int seq64::gui_drawingarea_gtk2::window_x ( ) const [inline]
```

### 13.21.3.3 `window_y()`

```
int seq64::gui_drawingarea_gtk2::window_y ( ) const [inline]
```

### 13.21.3.4 `current_x()`

```
int seq64::gui_drawingarea_gtk2::current_x ( ) const [inline]
```

#### 13.21.3.5 current\_y()

```
int seq64::gui_drawingarea_gtk2::current_y ( ) const [inline]
```

#### 13.21.3.6 drop\_x()

```
int seq64::gui_drawingarea_gtk2::drop_x ( ) const [inline]
```

#### 13.21.3.7 drop\_y()

```
int seq64::gui_drawingarea_gtk2::drop_y ( ) const [inline]
```

#### 13.21.3.8 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](#).

#### 13.21.3.9 perf()

```
perform& seq64::gui_drawingarea_gtk2::perf ( ) [inline], [protected]
```

#### 13.21.3.10 clear\_window()

```
void seq64::gui_drawingarea_gtk2::clear_window ( ) [inline], [protected]
```

One less need to access `m_window` directly.

#### 13.21.3.11 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.

**13.21.3.12 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.

**13.21.3.13 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.

**13.21.3.14 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.
<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.

**13.21.3.15 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.

**13.21.3.16 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.

**13.21.3.17 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.

**13.21.3.18 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.

**13.21.3.19 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.

**13.21.3.20 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.

**13.21.3.21 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.

**13.21.3.22 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-&gt;set_foreground(color)</code> . Defaults to true.

**13.21.3.23 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-&gt;set_foreground(color)</code> . Defaults to true.

**13.21.3.24 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-&gt;set_foreground(color)</code> . Defaults to true.

**13.21.3.25 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-&gt;set_foreground(color)</code> . Defaults to true.

**13.21.3.26 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-&gt;set_foreground(color)</code> . Defaults to true.

**13.21.3.27 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-&gt;set_foreground(color)</code> . Defaults to true.

**13.21.3.28 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-&gt;set_foreground(color)</code> . Defaults to true.

**13.21.3.29 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-&gt;set_foreground(color)</code> . Defaults to true.

**13.21.3.30 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-&gt;set_foreground(color)</code> . Defaults to true.

13.21.3.31 `draw_drawable()`

```
void seq64::gui_drawingarea_gtk2::draw_drawable (
    int xsrc,
    int ysrc,
    int xdest,
    int ydest,
    int width,
    int height ) [inline], [protected]
```

13.21.3.32 `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 <a href="#">gui_window_gtk2</a> 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-&gt;get_step_increment()</code> , or provided especially to keep up with the progress bar.

13.21.3.33 `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-&gt;get_step_increment()</code> .

## 13.21.3.34 scroll\_hset()

```
void seq64::gui_drawingarea_gtk2::scroll_hset (  
    Gtk::Adjustment & hadjust,  
    double value ) [protected]
```

## 13.21.3.35 scroll\_vset()

```
void seq64::gui_drawingarea_gtk2::scroll_vset (  
    Gtk::Adjustment & vadjust,  
    double value ) [protected]
```

## 13.21.3.36 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.
----------	------------------------

## 13.21.3.37 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.
----------	------------------------

### 13.21.3.38 gtk\_drawarea\_init()

```
void seq64::gui_drawingarea_gtk2::gtk_drawarea_init ( ) [private]
```

### 13.21.3.39 on\_realize()

```
void seq64::gui_drawingarea_gtk2::on_realize ( ) [protected]
```

It allocates any additional resources that weren't initialized in the constructor.

## 13.21.4 Field Documentation

### 13.21.4.1 m\_gc

```
Glib::RefPtr<Gdk::GC> seq64::gui_drawingarea_gtk2::m_gc [protected]
```

### 13.21.4.2 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.

### 13.21.4.3 m\_vadjust

```
Gtk::Adjustment& seq64::gui_drawingarea_gtk2::m_vadjust [protected]
```

### 13.21.4.4 m\_hadjust

```
Gtk::Adjustment& seq64::gui_drawingarea_gtk2::m_hadjust [protected]
```

### 13.21.4.5 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.



#### 13.21.4.6 m\_background

Glib::RefPtr<Gdk::Pixmap> seq64::gui\_drawingarea\_gtk2::m\_background [protected]

Our wrappers still leave this member exposed (giggle).

#### 13.21.4.7 m\_foreground

Glib::RefPtr<Gdk::Pixmap> seq64::gui\_drawingarea\_gtk2::m\_foreground [protected]

Our wrappers still leave this member exposed.

#### 13.21.4.8 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.

#### 13.21.4.9 m\_window\_x

int seq64::gui\_drawingarea\_gtk2::m\_window\_x [protected]

Could make this constant, but some windows are resizable. Window width value.

#### 13.21.4.10 m\_window\_y

int seq64::gui\_drawingarea\_gtk2::m\_window\_y [protected]

#### 13.21.4.11 m\_current\_x

int seq64::gui\_drawingarea\_gtk2::m\_current\_x [protected]

Current mouse x value.

#### 13.21.4.12 m\_current\_y

int seq64::gui\_drawingarea\_gtk2::m\_current\_y [protected]



'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](#) () const

'Getter' function for member `m_grey`

- const [Color](#) & [dark\\_grey](#) () const

'Getter' function for member `m_dk_grey`

- const [Color](#) & [light\\_grey](#) () const

'Getter' function for member `m_lt_grey`

- 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](#) & [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](#) () const

'Getter' function for member `m_blk_key`

- const [Color](#) & [white\\_key](#) () const

'Getter' function for member `m_wht_key`

- 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.*

## 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\\_dk\\_red](#)  
*Provides a blood-red color.*
- static const [Color m\\_dk\\_green](#)  
*Provides a dark green color.*
- static const [Color m\\_dk\\_orange](#)  
*Provides a dark orange color.*
- static const [Color m\\_dk\\_blue](#)  
*Provides the dark blue color.*
- static const [Color m\\_dk\\_magenta](#)  
*Provides the dark cyan color.*
- static const [Color m\\_dk\\_cyan](#)  
*Provides the dark cyan color.*
- static const [Color m\\_red](#)  
*Provides the red color.*
- static const [Color m\\_white](#)  
*Provides the white color.*
- static const [Color m\\_orange](#)  
*Provides the orange color.*
- static const [Color m\\_yellow](#)  
*Provides the yellow color.*

- static const [Color m\\_green](#)  
*Provides the green color.*
- static const [Color m\\_blue](#)  
*Provides the blue color.*
- static [Color m\\_grey](#)  
*Provides the grey color.*
- static [Color m\\_dk\\_grey](#)  
*Provides the dark grey color.*
- static [Color m\\_lt\\_grey](#)  
*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](#)  
*Provides the color of a black key.*
- static [Color m\\_wht\\_key](#)  
*Provides the color of a white key.*

### 13.22.1 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.

### 13.22.2 Member Typedef Documentation

#### 13.22.2.1 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.

### 13.22.3 Constructor & Destructor Documentation

#### 13.22.3.1 `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`.

#### 13.22.3.2 `~gui_palette_gtk2()`

```
seq64::gui_palette_gtk2::~~gui_palette_gtk2 ( )
```

### 13.22.4 Member Function Documentation

#### 13.22.4.1 `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

<code>inverse</code>	If true, load the alternate palette. Otherwise, load the default palette.
----------------------	---

#### 13.22.4.2 is\_inverse()

```
static bool seq64::gui_palette_gtk2::is_inverse ( ) [inline], [static]
```

#### 13.22.4.3 line\_color()

```
const Color& seq64::gui_palette_gtk2::line_color ( ) const [inline]
```

Might eventually be selectable from the "user" configuration file

#### 13.22.4.4 progress\_color()

```
const Color& seq64::gui_palette_gtk2::progress_color ( ) const [inline]
```

Now selectable from the "user" configuration file.

#### 13.22.4.5 black()

```
const Color& seq64::gui_palette_gtk2::black ( ) const [inline]
```

#### 13.22.4.6 dark\_red()

```
const Color& seq64::gui_palette_gtk2::dark_red ( ) const [inline]
```

#### 13.22.4.7 dark\_green()

```
const Color& seq64::gui_palette_gtk2::dark_green ( ) const [inline]
```

#### 13.22.4.8 dark\_orange()

```
const Color& seq64::gui_palette_gtk2::dark_orange ( ) const [inline]
```

**13.22.4.9 dark\_blue()**

```
const Color& seq64::gui_palette_gtk2::dark_blue ( ) const [inline]
```

**13.22.4.10 dark\_magenta()**

```
const Color& seq64::gui_palette_gtk2::dark_magenta ( ) const [inline]
```

**13.22.4.11 dark\_cyan()**

```
const Color& seq64::gui_palette_gtk2::dark_cyan ( ) const [inline]
```

**13.22.4.12 white()**

```
const Color& seq64::gui_palette_gtk2::white ( ) const [inline]
```

**13.22.4.13 grey()**

```
const Color& seq64::gui_palette_gtk2::grey ( ) const [inline]
```

**13.22.4.14 dark\_grey()**

```
const Color& seq64::gui_palette_gtk2::dark_grey ( ) const [inline]
```

**13.22.4.15 light\_grey()**

```
const Color& seq64::gui_palette_gtk2::light_grey ( ) const [inline]
```

**13.22.4.16 red()**

```
const Color& seq64::gui_palette_gtk2::red ( ) const [inline]
```



**13.22.4.17 orange()**

```
const Color& seq64::gui_palette_gtk2::orange ( ) const [inline]
```

**13.22.4.18 yellow()**

```
const Color& seq64::gui_palette_gtk2::yellow ( ) const [inline]
```

**13.22.4.19 green()**

```
const Color& seq64::gui_palette_gtk2::green ( ) const [inline]
```

**13.22.4.20 blue()**

```
const Color& seq64::gui_palette_gtk2::blue ( ) const [inline]
```

**13.22.4.21 black\_paint()**

```
const Color& seq64::gui_palette_gtk2::black_paint ( ) const [inline]
```

**13.22.4.22 white\_paint()**

```
const Color& seq64::gui_palette_gtk2::white_paint ( ) const [inline]
```

**13.22.4.23 black\_key()**

```
const Color& seq64::gui_palette_gtk2::black_key ( ) const [inline]
```

**13.22.4.24 white\_key()**

```
const Color& seq64::gui_palette_gtk2::white_key ( ) const [inline]
```

#### 13.22.4.25 `bg_color()` [1/2]

```
const Color& seq64::gui_palette_gtk2::bg_color ( ) const [inline]
```

#### 13.22.4.26 `bg_color()` [2/2]

```
void seq64::gui_palette_gtk2::bg_color (
    const Color & c ) [inline]
```

#### 13.22.4.27 `fg_color()` [1/2]

```
const Color& seq64::gui_palette_gtk2::fg_color ( ) const [inline]
```

#### 13.22.4.28 `fg_color()` [2/2]

```
void seq64::gui_palette_gtk2::fg_color (
    const Color & c ) [inline]
```

### 13.22.5 Field Documentation

#### 13.22.5.1 `m_is_inverse`

```
bool seq64::gui_palette_gtk2::m_is_inverse [static], [private]
```

By default, the inverse color palette is not loaded.

#### 13.22.5.2 `m_black`

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_black [static], [private]
```

#### 13.22.5.3 `m_dk_red`

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_red [static], [private]
```

#### 13.22.5.4 m\_dk\_green

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_green [static], [private]
```

#### 13.22.5.5 m\_dk\_orange

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_orange [static], [private]
```

#### 13.22.5.6 m\_dk\_blue

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_blue [static], [private]
```

#### 13.22.5.7 m\_dk\_magenta

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_magenta [static], [private]
```

#### 13.22.5.8 m\_dk\_cyan

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_dk_cyan [static], [private]
```

#### 13.22.5.9 m\_red

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_red [static], [private]
```

#### 13.22.5.10 m\_white

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_white [static], [private]
```

#### 13.22.5.11 m\_orange

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_orange [static], [private]
```

**13.22.5.12 m\_yellow**

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_yellow [static], [private]
```

**13.22.5.13 m\_green**

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_green [static], [private]
```

**13.22.5.14 m\_blue**

```
const STATIC_COLOR seq64::gui_palette_gtk2::m_blue [static], [private]
```

**13.22.5.15 m\_grey**

```
STATIC_COLOR seq64::gui_palette_gtk2::m_grey [static], [private]
```

**13.22.5.16 m\_dk\_grey**

```
STATIC_COLOR seq64::gui_palette_gtk2::m_dk_grey [static], [private]
```

**13.22.5.17 m\_lt\_grey**

```
STATIC_COLOR seq64::gui_palette_gtk2::m_lt_grey [static], [private]
```

**13.22.5.18 m\_blk\_paint**

```
STATIC_COLOR seq64::gui_palette_gtk2::m_blk_paint [static], [private]
```

**13.22.5.19 m\_wht\_paint**

```
STATIC_COLOR seq64::gui_palette_gtk2::m_wht_paint [static], [private]
```

#### 13.22.5.20 m\_blk\_key

STATIC\_COLOR seq64::gui\_palette\_gtk2::m\_blk\_key [static], [private]

#### 13.22.5.21 m\_wht\_key

STATIC\_COLOR seq64::gui\_palette\_gtk2::m\_wht\_key [static], [private]

#### 13.22.5.22 m\_line\_color

Color seq64::gui\_palette\_gtk2::m\_line\_color [private]

#### 13.22.5.23 m\_progress\_color

Color seq64::gui\_palette\_gtk2::m\_progress\_color [private]

#### 13.22.5.24 m\_bg\_color

Color seq64::gui\_palette\_gtk2::m\_bg\_color [private]

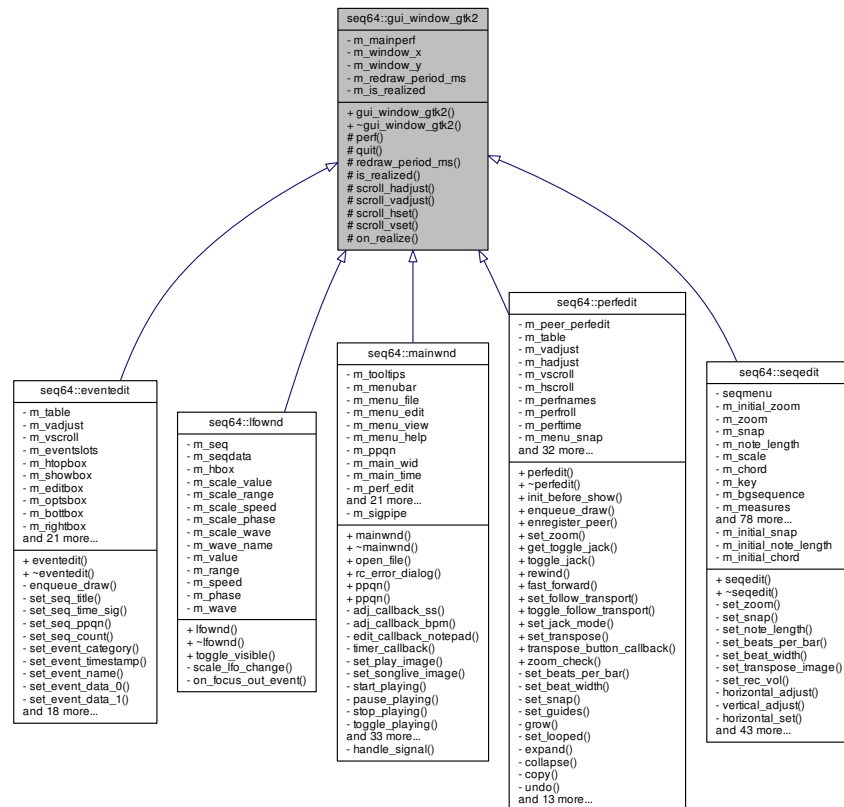
#### 13.22.5.25 m\_fg\_color

Color seq64::gui\_palette\_gtk2::m\_fg\_color [private]

## 13.23 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*
- 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.*

## 13.23.1 Constructor & Destructor Documentation

### 13.23.1.1 `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

<code>p</code>	Refers to the main performance object.
<code>window_x</code>	The width of the window.
<code>window_y</code>	The height of the window.

### 13.23.1.2 ~gui\_window\_gtk2()

```
seq64::gui_window_gtk2::~~gui_window_gtk2 ( ) [virtual]
```

## 13.23.2 Member Function Documentation

### 13.23.2.1 perf()

```
perform& seq64::gui_window_gtk2::perf ( ) [inline], [protected]
```

### 13.23.2.2 quit()

```
virtual void seq64::gui_window_gtk2::quit ( ) [inline], [protected], [virtual]
```

### 13.23.2.3 redraw\_period\_ms()

```
int seq64::gui_window_gtk2::redraw_period_ms ( ) const [inline], [protected]
```

### 13.23.2.4 is\_realized()

```
bool seq64::gui_window_gtk2::is_realized ( ) const [inline], [protected]
```

### 13.23.2.5 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-&gt;get_step_increment()</code> , or provided especially to keep up with the progress bar.



### 13.23.2.6 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.

### 13.23.2.7 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.

### 13.23.2.8 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.

### 13.23.2.9 on\_realize()

```
void seq64::gui_window_gtk2::on_realize ( ) [protected]
```

### 13.23.3 Field Documentation

#### 13.23.3.1 m\_mainperf

```
perform& seq64::gui_window_gtk2::m_mainperf [private]
```

And a whole lot more than that.

#### 13.23.3.2 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.

#### 13.23.3.3 m\_window\_y

```
int seq64::gui_window_gtk2::m_window_y [private]
```

#### 13.23.3.4 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.

#### 13.23.3.5 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.

## 13.24 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\\_perfeddit](#) (bool start)  
*'Setter' function for member [parent\(\).start\\_from\\_perfeddit\(\)](#)*
- 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*

## 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.*
- void [show\\_position](#) (const jack\_position\_t &pos) const  
*Shows a one-line summary of a JACK position structure.*
- int [sync](#) (jack\_transport\_state\_t state=(jack\_transport\_state\_t)(-1))  
*A helper function for syncing up with JACK parameters.*
- void [set\\_position](#) ([midipulse](#) currenttick)  
*Provides the code that was effectively commented out in the [perform::position\\_jack\(\)](#) function.*

## 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)
- void [jack\\_shutdown\\_callback](#) (void \*arg)  
*This callback is to shut down JACK by clearing the [jack\\_assistant](#) :: m\_jack\_running flag.*
- int [jack\\_sync\\_callback](#) (jack\_transport\_state\_t state, jack\_position\_t \*pos, 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)
- void [jack\\_session\\_callback](#) (jack\_session\_event\_t \*ev, void \*arg)  
*Set the m\_jsession\_ev (event) value of the perform object.*

## 13.24.1 Constructor & Destructor Documentation

### 13.24.1.1 jack\_assistant()

```
seq64::jack_assistant::jack_assistant (
    perform & parent,
    midi_bpm bpmminute = SEQ64_DEFAULT_BPM,
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    int bpmmeasure = 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>bpmmeasure</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.

### 13.24.1.2 ~jack\_assistant()

```
seq64::jack_assistant::~~jack_assistant ( )
```

The perform object currently calls [jack\\_assistant::deinit\(\)](#), but that call could be made here instead.

## 13.24.2 Member Function Documentation

### 13.24.2.1 parent() [1/2]

```
perform& seq64::jack_assistant::parent ( ) [inline]
```

### 13.24.2.2 parent() [2/2]

```
const perform& seq64::jack_assistant::parent ( ) const [inline]
```

### 13.24.2.3 is\_running()

```
bool seq64::jack_assistant::is_running ( ) const [inline]
```

### 13.24.2.4 is\_master()

```
bool seq64::jack_assistant::is_master ( ) const [inline]
```

### 13.24.2.5 get\_ppqn()

```
int seq64::jack_assistant::get_ppqn ( ) const [inline]
```

### 13.24.2.6 get\_beat\_width()

```
int seq64::jack_assistant::get_beat_width ( ) const [inline]
```

### 13.24.2.7 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.
-----------	---

13.24.2.8 `get_beats_per_measure()`

```
int seq64::jack_assistant::get_beats_per_measure ( ) const [inline]
```

13.24.2.9 `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.
------------	--

13.24.2.10 `get_beats_per_minute()`

```
midibpm seq64::jack_assistant::get_beats_per_minute ( ) const [inline]
```

13.24.2.11 `set_beats_per_minute()`

```
void seq64::jack_assistant::set_beats_per_minute (
    midibpm bpmminute ) [inline]
```

We should consider adding validation. However, `perform::set_beats_per_minute()` does validate already.

## Parameters

<i>bpmminute</i>	Provides the beats/minute value to set.
------------------	---

13.24.2.12 `transport_state()`

```
jack_transport_state_t seq64::jack_assistant::transport_state ( ) const [inline]
```



## 13.24.2.13 transport\_not\_starting()

```
bool seq64::jack_assistant::transport_not_starting ( ) const [inline]
```

## 13.24.2.14 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.)

### 13.24.2.15 deinit()

```
bool seq64::jack_assistant::deinit ( )
```

#### Returns

Returns the value of `m_jack_running`, which should be false.

### 13.24.2.16 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.

### 13.24.2.17 activate()

```
bool seq64::jack_assistant::activate ( )
```

#### 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.

### 13.24.2.18 start()

```
void seq64::jack_assistant::start ( )
```

This function assumes that `m_jack_client` is not null, if `m_jack_running` is true.

Found this note in the Hydrogen code:

When `jack_transport_start()` is called, it takes effect from the next processing cycle. The location info from the `timebase_master`, if there is one, will not be available until the `_next_` next cycle. The code must therefore wait one cycle before syncing up with `timebase_master`.

## 13.24.2.19 stop()

```
void seq64::jack_assistant::stop ( )
```

This function assumes that `m_jack_client` is not null, if `m_jack_running` is true.

## 13.24.2.20 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.
-------------	--

13.24.2.21 `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 perfrroll grid will be incorrect. Perfrroll 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 <code>jack_assistant</code> object.
------------	---

## Returns

Returns true if JACK is running.

13.24.2.22 `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.
-------------	---------------------------------

**13.24.2.23 get\_jack\_tick()**

```
double seq64::jack_assistant::get_jack_tick ( ) const [inline]
```

**13.24.2.24 get\_jack\_pos()**

```
const jack_position_t& seq64::jack_assistant::get_jack_pos ( ) const [inline]
```

**13.24.2.25 toggle\_jack\_mode()**

```
void seq64::jack_assistant::toggle_jack_mode ( ) [inline]
```

**13.24.2.26 set\_jack\_mode()**

```
void seq64::jack_assistant::set_jack_mode (
    bool mode ) [inline]
```

**13.24.2.27 get\_jack\_mode()**

```
bool seq64::jack_assistant::get_jack_mode ( ) const [inline]
```

**13.24.2.28 get\_jack\_stop\_tick()**

```
midipulse seq64::jack_assistant::get_jack_stop_tick ( ) const [inline]
```

#### 13.24.2.29 set\_jack\_stop\_tick()

```
void seq64::jack_assistant::set_jack_stop_tick (
    long tick ) [inline]
```

#### 13.24.2.30 jack\_frame\_rate()

```
jack_nframes_t seq64::jack_assistant::jack_frame_rate ( ) const [inline]
```

#### 13.24.2.31 get\_follow\_transport()

```
bool seq64::jack_assistant::get_follow_transport ( ) const [inline]
```

#### 13.24.2.32 set\_follow\_transport()

```
void seq64::jack_assistant::set_follow_transport (
    bool aset ) [inline]
```

#### 13.24.2.33 toggle\_follow\_transport()

```
void seq64::jack_assistant::toggle_follow_transport ( ) [inline]
```

#### 13.24.2.34 toggle\_song\_start\_mode()

```
bool seq64::jack_assistant::toggle_song_start_mode ( )
```

#### 13.24.2.35 song\_start\_mode()

```
bool seq64::jack_assistant::song_start_mode ( ) const
```

**13.24.2.36 set\_start\_from\_perfeddit()**

```
void seq64::jack_assistant::set_start_from_perfeddit (
    bool start )
```

**13.24.2.37 client()**

```
jack_client_t * seq64::jack_assistant::client ( ) const
```

**13.24.2.38 client\_name()**

```
const std::string& seq64::jack_assistant::client_name ( ) const [inline]
```

**13.24.2.39 client\_uuid()**

```
const std::string& seq64::jack_assistant::client_uuid ( ) const [inline]
```

**13.24.2.40 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.
-------------	---------------------------------------

**13.24.2.41 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.

**13.24.2.42 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 <a href="#">create_jack_client()</a> . 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.

**13.24.2.43 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.

**13.24.2.44 show\_position()**

```
void seq64::jack_assistant::show_position (
    const jack_position_t & pos ) const [private]
```

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:





**Parameters**

<i>state</i>	The JACK transport state to be set.
--------------	-------------------------------------

**13.24.2.46 set\_position()**

```
void seq64::jack_assistant::set_position (
    midipulse currenttick ) [private]
```

We might be able to use it in other functions.

Computing the BBT information from the frame number is relatively simple here, but would become complex if we supported tempo or time signature changes at specific locations in the transport timeline.

```
ticks * 10 = jack ticks;
jack ticks / ticks per beat = num beats;
num beats / beats per minute = num minutes
num minutes * 60 = num seconds
num seconds * frame_rate = frame
```

**Parameters**

<i>currenttick</i>	Provides the current position to be set.
--------------------	--

**13.24.2.47 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.

**13.24.2.48 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.

**13.24.3 Friends And Related Function Documentation****13.24.3.1 jack\_transport\_callback**

```
int jack_transport_callback (
    jack_nframes_t nframes,
    void * arg ) [friend]
```

**13.24.3.2 jack\_shutdown\_callback**

```
void jack_shutdown_callback (
    void * arg ) [friend]
```

**Parameters**

<i>arg</i>	Points to the <a href="#">jack_assistant</a> in charge of JACK support for the perform object.
------------	--

**13.24.3.3 jack\_sync\_callback**

```
int jack_sync_callback (
    jack_transport_state_t state,
    jack_position_t * pos,
    void * arg ) [friend]
```

This JACK synchronization callback informs the specified perform object of the current state and parameters of JACK.

The transport state will be:

- JackTransportStopped when a new position is requested.
- JackTransportStarting when the transport is waiting to start.
- JackTransportRolling when the timeout has expired, and the position is now a moving target.

This is the slow-sync callback, which the stazed code replaces with [jack\\_transport\\_callback\(\)](#).

## Parameters

<i>state</i>	The JACK Transport state.
<i>pos</i>	The JACK position value.
<i>arg</i>	The pointer to the <a href="#">jack_assistant</a> object. Currently not checked for nullity, nor dynamic-casted.

## Returns

Returns 1 if the function works, and 0 if something was wrong.

## 13.24.3.4 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 seconds 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-&gt;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 <code>pos</code> , 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 <a href="#">jack_assistant</a> pointer, currently unchecked for nullity.

#### 13.24.3.5 get\_current\_jack\_position

```
long get_current_jack_position (
    void * arg ) [friend]
```

#### Warning

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

#### 13.24.3.6 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 <a href="#">jack_assistant</a> object. Currently not checked for nullity.

## 13.24.4 Field Documentation

## 13.24.4.1 sm\_status\_pairs

```
jack_status_pair_t seq64::jack_assistant::sm_status_pairs[] [static], [private]
```

## 13.24.4.2 m\_jack\_parent

```
perform& seq64::jack_assistant::m_jack_parent [private]
```

## 13.24.4.3 m\_jack\_client

```
jack_client_t* seq64::jack_assistant::m_jack_client [mutable], [private]
```

## 13.24.4.4 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.

## 13.24.4.5 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.

## 13.24.4.6 m\_jack\_frame\_current

```
jack_nframes_t seq64::jack_assistant::m_jack_frame_current [private]
```

#### 13.24.4.7 m\_jack\_frame\_last

```
jack_nframes_t seq64::jack_assistant::m_jack_frame_last [private]
```

Also used in incrementing m\_jack\_tick.

#### 13.24.4.8 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.

#### 13.24.4.9 m\_jack\_transport\_state

```
jack_transport_state_t seq64::jack_assistant::m_jack_transport_state [private]
```

Common values are `JackTransportStopped`, `JackTransportRolling`, and `JackTransportLooping`.

#### 13.24.4.10 m\_jack\_transport\_state\_last

```
jack_transport_state_t seq64::jack_assistant::m_jack_transport_state_last [private]
```

#### 13.24.4.11 m\_jack\_tick

```
double seq64::jack_assistant::m_jack_tick [private]
```

#### 13.24.4.12 m\_jsession\_ev

```
jack_session_event_t* seq64::jack_assistant::m_jsession_ev [private]
```

Used in the [session\\_event\(\)](#) function.

#### 13.24.4.13 m\_jack\_running

```
bool seq64::jack_assistant::m_jack_running [private]
```



#### 13.24.4.14 m\_jack\_master

```
bool seq64::jack_assistant::m_jack_master [private]
```

#### 13.24.4.15 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.

#### 13.24.4.16 m\_toggle\_jack

```
bool seq64::jack_assistant::m_toggle_jack [private]
```

#### 13.24.4.17 m\_jack\_stop\_tick

```
midipulse seq64::jack_assistant::m_jack_stop_tick [private]
```

Repositions the transport marker.

#### 13.24.4.18 m\_follow\_transport

```
bool seq64::jack_assistant::m_follow_transport [private]
```

#### 13.24.4.19 m\_ppqn

```
int seq64::jack_assistant::m_ppqn [private]
```

It is used for calculating ticks/beat (pulses/beat) and for setting the tick position.

#### 13.24.4.20 m\_beats\_per\_measure

```
int seq64::jack_assistant::m_beats_per_measure [private]
```

#### 13.24.4.21 m\_beat\_width

```
int seq64::jack_assistant::m_beat_width [private]
```

#### 13.24.4.22 m\_beats\_per\_minute

```
midibpm seq64::jack_assistant::m_beats_per_minute [private]
```

### 13.25 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.*

#### 13.25.1 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.

## 13.25.2 Field Documentation

### 13.25.2.1 js\_current\_tick

double seq64::jack\_scratchpad::js\_current\_tick

### 13.25.2.2 js\_total\_tick

double seq64::jack\_scratchpad::js\_total\_tick

### 13.25.2.3 js\_clock\_tick

double seq64::jack\_scratchpad::js\_clock\_tick

### 13.25.2.4 js\_jack\_stopped

bool seq64::jack\_scratchpad::js\_jack\_stopped

### 13.25.2.5 js\_dumping

bool seq64::jack\_scratchpad::js\_dumping

### 13.25.2.6 js\_init\_clock

bool seq64::jack\_scratchpad::js\_init\_clock

### 13.25.2.7 js\_looping

bool seq64::jack\_scratchpad::js\_looping

### 13.25.2.8 js\_playback\_mode

```
bool seq64::jack_scratchpad::js_playback_mode
```

### 13.25.2.9 js\_ticks\_converted

```
double seq64::jack_scratchpad::js_ticks_converted
```

### 13.25.2.10 js\_ticks\_delta

```
double seq64::jack_scratchpad::js_ticks_delta
```

### 13.25.2.11 js\_ticks\_converted\_last

```
double seq64::jack_scratchpad::js_ticks_converted_last
```

### 13.25.2.12 js\_delta\_tick\_frac

```
long seq64::jack_scratchpad::js_delta_tick_frac
```

## 13.26 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.*

### 13.26.1 Field Documentation

## 13.26.1.1 jf\_bit

```
unsigned seq64::jack_status_pair_t::jf_bit
```

## 13.26.1.2 jf\_meaning

```
std::string seq64::jack_status_pair_t::jf_meaning
```

## 13.27 seq64::keybindentry Class Reference

Class for management of application key-bindings.

Inherits Entry.

### Public Member Functions

- [keybindentry](#) ([type](#) t, unsigned int \*location\_to\_write=NULLPTR, [perform](#) \*p=NULLPTR, long s=0)  
*This constructor initializes the member with values dependent on the value type provided in the first parameter.*
- void [set](#) (unsigned int 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 int \* [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.*
- long [m\\_slot](#)  
*Provides an index into a set of group-keys or event-keys.*

## Friends

- class [options](#)

## 13.27.1 Member Enumeration Documentation

### 13.27.1.1 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.

## 13.27.2 Constructor & Destructor Documentation

### 13.27.2.1 keybindentry()

```
seq64::keybindentry::keybindentry (
    type t,
    unsigned int * location_to_write = nullptr,
    perform * p = nullptr,
    long 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. The default value of this parameter is the null pointer.
<i>p</i>	Points to the performance object used with this key-binding. The default value of this parameter is the null pointer.
<i>s</i>	Provides the slot value for this key-binding. The default value of this parameter is zero.

### 13.27.3 Member Function Documentation

#### 13.27.3.1 set()

```
void seq64::keybindentry::set (
    unsigned int val )
```

Then we call `set_text(buf)`. The `set_width_char()` function is then called.

#### 13.27.3.2 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()`.

### 13.27.4 Friends And Related Function Documentation

#### 13.27.4.1 options

```
friend class options [friend]
```

### 13.27.5 Field Documentation

#### 13.27.5.1 m\_key

```
unsigned int* 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.

### 13.27.5.2 m\_type

```
type seq64::keybindentry::m_type [private]
```

### 13.27.5.3 m\_perf

```
perform* seq64::keybindentry::m_perf [private]
```

### 13.27.5.4 m\_slot

```
long seq64::keybindentry::m_slot [private]
```

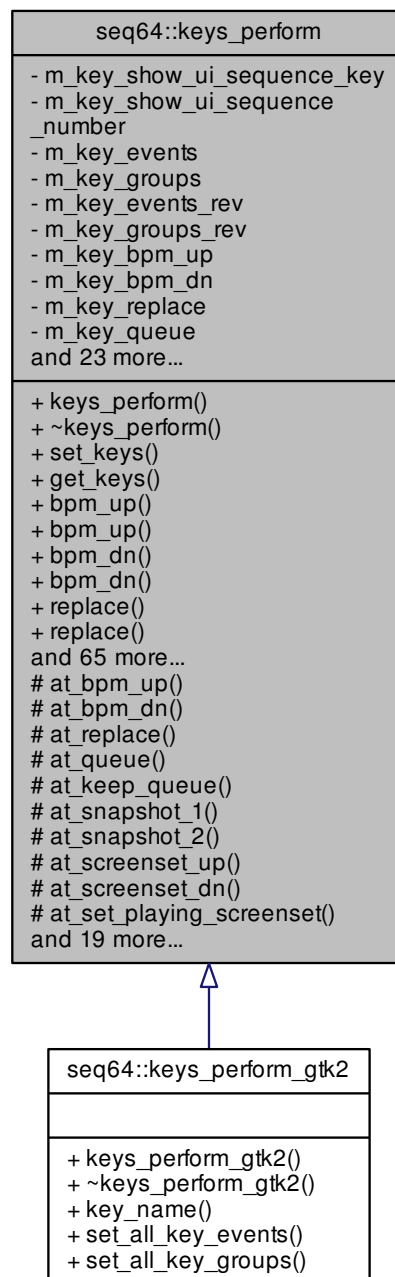
(This item should be changed to unsigned int, though.)

## 13.28 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 sets some running flags to false, signals this condition, then joins the input and output threads if the were launched.*

- 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 int [bpm\\_up](#) () const  
*'Getter' function for member m\_key\_bpm\_up*
- void [bpm\\_up](#) (unsigned int x)  
*'Setter' function for member m\_key\_bpm\_up*
- unsigned int [bpm\\_dn](#) () const  
*'Getter' function for member m\_key\_bpm\_dn*
- void [bpm\\_dn](#) (unsigned int x)  
*'Setter' function for member m\_key\_bpm\_dn*
- unsigned int [replace](#) () const  
*'Getter' function for member m\_key\_replace*
- void [replace](#) (unsigned int x)  
*'Setter' function for member m\_key\_replace*
- unsigned int [queue](#) () const  
*'Getter' function for member m\_key\_queue*
- void [queue](#) (unsigned int x)  
*'Setter' function for member m\_key\_queue*
- unsigned int [keep\\_queue](#) () const  
*'Getter' function for member m\_key\_keep\_queue*
- void [keep\\_queue](#) (unsigned int x)  
*'Setter' function for member m\_key\_keep\_queue*
- unsigned int [snapshot\\_1](#) () const  
*'Getter' function for member m\_key\_snapshot\_1*
- void [snapshot\\_1](#) (unsigned int x)  
*'Setter' function for member m\_key\_snapshot\_1*
- unsigned int [snapshot\\_2](#) () const  
*'Getter' function for member m\_key\_snapshot\_2*
- void [snapshot\\_2](#) (unsigned int x)  
*'Setter' function for member m\_key\_snapshot\_2*
- unsigned int [screenshot\\_up](#) () const  
*'Getter' function for member m\_key\_screenshot\_up*
- void [screenshot\\_up](#) (unsigned int x)  
*'Setter' function for member m\_key\_screenshot\_up*
- unsigned int [screenshot\\_dn](#) () const  
*'Getter' function for member m\_key\_screenshot\_dn*
- void [screenshot\\_dn](#) (unsigned int x)  
*'Setter' function for member m\_key\_screenshot\_dn*
- unsigned int [set\\_playing\\_screenshot](#) () const  
*'Getter' function for member m\_key\_playing\_screenshot*
- void [set\\_playing\\_screenshot](#) (unsigned int x)  
*'Setter' function for member m\_key\_playing\_screenshot*
- unsigned int [group\\_on](#) () const  
*'Getter' function for member m\_key\_group\_on*
- void [group\\_on](#) (unsigned int x)  
*'Setter' function for member m\_key\_group\_on*
- unsigned int [group\\_off](#) () const  
*'Getter' function for member m\_key\_group\_off*
- void [group\\_off](#) (unsigned int x)

- 'Setter' function for member m\_key\_group\_off*
- unsigned int [group\\_learn](#) () const
- 'Getter' function for member m\_key\_group\_learn*
- void [group\\_learn](#) (unsigned int x)
- 'Setter' function for member m\_key\_group\_learn*
- unsigned int [start](#) () const
- 'Getter' function for member m\_key\_start*
- void [start](#) (unsigned int x)
- 'Setter' function for member m\_key\_start*
- unsigned int [pause](#) () const
- 'Getter' function for member m\_key\_pause*
- void [pause](#) (unsigned int x)
- 'Setter' function for member m\_key\_pause*
- unsigned int [pattern\\_edit](#) () const
- 'Getter' function for member m\_key\_pattern\_edit*
- void [pattern\\_edit](#) (unsigned int x)
- 'Setter' function for member m\_key\_pattern\_edit*
- unsigned int [event\\_edit](#) () const
- 'Getter' function for member m\_key\_event\_edit*
- void [event\\_edit](#) (unsigned int x)
- 'Setter' function for member m\_key\_event\_edit*
- unsigned int [stop](#) () const
- 'Getter' function for member m\_key\_stop*
- void [stop](#) (unsigned int x)
- 'Setter' function for member m\_key\_stop*
- unsigned int [song\\_mode](#) () const
- void [song\\_mode](#) (unsigned int key)
- unsigned int [menu\\_mode](#) () const
- void [menu\\_mode](#) (unsigned int key)
- unsigned int [follow\\_transport](#) () const
- void [follow\\_transport](#) (unsigned int key)
- unsigned int [fast\\_forward](#) () const
- void [fast\\_forward](#) (unsigned int key)
- unsigned int [rewind](#) () const
- void [rewind](#) (unsigned int key)
- unsigned int [pointer\\_position](#) () const
- void [pointer\\_position](#) (unsigned int key)
- unsigned int [toggle\\_mutes](#) () const
- void [toggle\\_mutes](#) (unsigned int key)
- unsigned int [toggle\\_jack](#) () const
- void [toggle\\_jack](#) (unsigned int key)
- unsigned int [tap\\_bpm](#) () const
- void [tap\\_bpm](#) (unsigned int 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*
- [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 int [lookup\\_keyevent\\_key](#) (long seqnum)
- *'Getter' function for member m\_key\_events\_rev[seqnum];*
- long [lookup\\_keyevent\\_seq](#) (unsigned int keycode)
- *'Getter' function for member m\_key\_events\_rev[keycode];*
- unsigned int [lookup\\_keygroup\\_key](#) (long groupnum)
- *'Getter' function for member m\_key\_events\_rev[groupnum];*
- long [lookup\\_keygroup\\_group](#) (unsigned int keycode)
- *'Getter' function for member m\_key\_events\_rev[keycode];*
- virtual std::string [key\\_name](#) (unsigned int 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 int keycode, long sequence\_slot)
- *At construction time, this function sets up one keycode and one event slot.*
- void [set\\_key\\_group](#) (unsigned int keycode, long group\_slot)
- *At construction time, this function sets up one keycode and one group slot.*

## Protected Types

- typedef std::map< unsigned int, long > [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< long, unsigned int > [RevSlotMap](#)
- *This typedef is like SlotMap, but used for lookup in the other direction.*

## Protected Member Functions

- unsigned int \* [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 int \* [at\\_bpm\\_dn](#) ()
- *'Getter' function for member m\_key\_bpm\_dn*
- unsigned int \* [at\\_replace](#) ()
- *'Getter' function for member m\_key\_replace*
- unsigned int \* [at\\_queue](#) ()
- *'Getter' function for member m\_key\_queue*
- unsigned int \* [at\\_keep\\_queue](#) ()
- *'Getter' function for member m\_key\_keep\_queue*
- unsigned int \* [at\\_snapshot\\_1](#) ()
- *'Getter' function for member m\_key\_snapshot\_1*
- unsigned int \* [at\\_snapshot\\_2](#) ()

- 'Getter' function for member m\_key\_snapshot\_2*
- unsigned int \* [at\\_screensset\\_up](#) ()
- 'Getter' function for member m\_key\_screensset\_up*
- unsigned int \* [at\\_screensset\\_dn](#) ()
- 'Getter' function for member m\_key\_screensset\_dn*
- unsigned int \* [at\\_set\\_playing\\_screensset](#) ()
- 'Getter' function for member m\_key\_playing\_screensset*
- unsigned int \* [at\\_group\\_on](#) ()
- 'Getter' function for member m\_key\_group\_on*
- unsigned int \* [at\\_group\\_off](#) ()
- 'Getter' function for member m\_key\_group\_off*
- unsigned int \* [at\\_group\\_learn](#) ()
- 'Getter' function for member m\_key\_group\_learn*
- unsigned int \* [at\\_start](#) ()
- 'Getter' function for member m\_key\_start*
- unsigned int \* [at\\_pause](#) ()
- 'Getter' function for member m\_key\_pause*
- unsigned int \* [at\\_song\\_mode](#) ()
- 'Getter' function for member m\_key\_song\_mode*
- unsigned int \* [at\\_toggle\\_jack](#) ()
- 'Getter' function for member m\_key\_toggle\_jack*
- unsigned int \* [at\\_menu\\_mode](#) ()
- 'Getter' function for member m\_key\_menu\_mode*
- unsigned int \* [at\\_follow\\_transport](#) ()
- 'Getter' function for member m\_key\_follow\_transport*
- unsigned int \* [at\\_fast\\_forward](#) ()
- 'Getter' function for member m\_key\_fast\_forward*
- unsigned int \* [at\\_rewind](#) ()
- 'Getter' function for member m\_key\_rewind*
- unsigned int \* [at\\_pointer\\_position](#) ()
- 'Getter' function for member m\_key\_pointer\_position*
- unsigned int \* [at\\_toggle\\_mutes](#) ()
- 'Getter' function for member m\_key\_toggle\_mutes*
- unsigned int \* [at\\_tap\\_bpm](#) ()
- 'Getter' function for member m\_key\_tap\_bpm*
- unsigned int \* [at\\_pattern\\_edit](#) ()
- 'Getter' function for member m\_key\_pattern\_edit*
- unsigned int \* [at\\_event\\_edit](#) ()
- 'Getter' function for member m\_key\_event\_edit*
- unsigned int \* [at\\_stop](#) ()
- 'Getter' function for member m\_key\_stop*
- bool \* [at\\_show\\_ui\\_sequence\\_key](#) ()
- 'Getter' function for member m\_key\_show\_ui\_sequence\_key*
- bool \* [at\\_show\\_ui\\_sequence\\_number](#) ()
- 'Getter' function for member m\_key\_show\_ui\_sequence\_number*

## 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.*
- unsigned int [m\\_key\\_bpm\\_up](#)  
*Provides key assignments for some key sequencer features.*
- unsigned int [m\\_key\\_bpm\\_dn](#)  
*BPM down, semicolon.*
- unsigned int [m\\_key\\_replace](#)  
*Replace, Ctrl-L.*
- unsigned int [m\\_key\\_queue](#)  
*Queue, Ctrl-R.*
- unsigned int [m\\_key\\_keep\\_queue](#)  
*Keep queue, backslash.*
- unsigned int [m\\_key\\_snapshot\\_1](#)  
*Snapshot 1, Alt-L.*
- unsigned int [m\\_key\\_snapshot\\_2](#)  
*Snapshot 1, Alt-R.*
- unsigned int [m\\_key\\_screenset\\_up](#)  
*Set up, Right-].*
- unsigned int [m\\_key\\_screenset\\_dn](#)  
*Set down, Left-[.*
- unsigned int [m\\_key\\_set\\_playing\\_screenset](#)  
*Set set, Home key.*
- unsigned int [m\\_key\\_group\\_on](#)  
*Group on, igrave key.*
- unsigned int [m\\_key\\_group\\_off](#)  
*Group off, apostrophe!*
- unsigned int [m\\_key\\_group\\_learn](#)  
*Group learn, Insert.*
- unsigned int [m\\_key\\_start](#)  
*Start play, Space key.*
- unsigned int [m\\_key\\_pause](#)  
*Pause play, Period.*
- unsigned int [m\\_key\\_song\\_mode](#)  
*Song versus Live mode.*
- unsigned int [m\\_key\\_toggle\\_jack](#)  
*Toggle JACK connect.*
- unsigned int [m\\_key\\_menu\\_mode](#)  
*Menu enabled/disabled.*
- unsigned int [m\\_key\\_follow\\_transport](#)

*Toggle following JACK.*

- unsigned int [m\\_key\\_rewind](#)

*Start rewind.*

- unsigned int [m\\_key\\_fast\\_forward](#)

*Start fast-forward.*

- unsigned int [m\\_key\\_pointer\\_position](#)

*Set progress to mouse.*

- unsigned int [m\\_key\\_toggle\\_mutes](#)

*Toggle all patterns.*

- unsigned int [m\\_key\\_tap\\_bpm](#)

*To tap out the BPM.*

- unsigned int [m\\_key\\_pattern\\_edit](#)

*Show pattern editor.*

- unsigned int [m\\_key\\_event\\_edit](#)

*Show event editor.*

- unsigned int [m\\_key\\_stop](#)

*Stop play, Escape.*

## Friends

- class [options](#)
- class [perform](#)
- class [optionsfile](#)

## 13.28.1 Detailed Description

It provides a way a mapping keystrokes to sequencer actions and song settings.

## 13.28.2 Member Typedef Documentation

### 13.28.2.1 SlotMap

```
typedef std::map<unsigned int, long> seq64::keys_perform::SlotMap [protected]
```

### 13.28.2.2 RevSlotMap

```
typedef std::map<long, unsigned int> seq64::keys_perform::RevSlotMap [protected]
```

## 13.28.3 Constructor & Destructor Documentation

**13.28.3.1 keys\_perform()**

```
seq64::keys_perform::keys_perform ( )
```

**13.28.3.2 ~keys\_perform()**

```
seq64::keys_perform::~keys_perform ( ) [virtual]
```

Finally, any active patterns/sequences are deleted.

**13.28.4 Member Function Documentation****13.28.4.1 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.
------------	--

**13.28.4.2 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.
------------	--

**13.28.4.3 bpm\_up()** [1/2]

```
unsigned int seq64::keys_perform::bpm_up ( ) const [inline]
```



#### 13.28.4.4 bpm\_up() [2/2]

```
void seq64::keys_perform::bpm_up (
    unsigned int x ) [inline]
```

##### Parameters

x	The key value to assign to the operation.
---	---

#### 13.28.4.5 bpm\_dn() [1/2]

```
unsigned int seq64::keys_perform::bpm_dn ( ) const [inline]
```

#### 13.28.4.6 bpm\_dn() [2/2]

```
void seq64::keys_perform::bpm_dn (
    unsigned int x ) [inline]
```

##### Parameters

x	The key value to assign to the operation.
---	---

#### 13.28.4.7 replace() [1/2]

```
unsigned int seq64::keys_perform::replace ( ) const [inline]
```

#### 13.28.4.8 replace() [2/2]

```
void seq64::keys_perform::replace (
    unsigned int x ) [inline]
```

##### Parameters

x	The key value to assign to the operation.
---	---

**13.28.4.9 queue()** [1/2]

```
unsigned int seq64::keys_perform::queue ( ) const [inline]
```

**13.28.4.10 queue()** [2/2]

```
void seq64::keys_perform::queue (
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.11 keep\_queue()** [1/2]

```
unsigned int seq64::keys_perform::keep_queue ( ) const [inline]
```

**13.28.4.12 keep\_queue()** [2/2]

```
void seq64::keys_perform::keep_queue (
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.13 snapshot\_1()** [1/2]

```
unsigned int seq64::keys_perform::snapshot_1 ( ) const [inline]
```

**13.28.4.14 snapshot\_1()** [2/2]

```
void seq64::keys_perform::snapshot_1 (
    unsigned int x ) [inline]
```

## Parameters

x	The key value to assign to the operation.
---	---

**13.28.4.15 snapshot\_2()** [1/2]

```
unsigned int seq64::keys_perform::snapshot_2 ( ) const [inline]
```

**13.28.4.16 snapshot\_2()** [2/2]

```
void seq64::keys_perform::snapshot_2 (
    unsigned int x ) [inline]
```

## Parameters

x	The key value to assign to the operation.
---	---

**13.28.4.17 screenset\_up()** [1/2]

```
unsigned int seq64::keys_perform::screenset_up ( ) const [inline]
```

**13.28.4.18 screenset\_up()** [2/2]

```
void seq64::keys_perform::screenset_up (
    unsigned int x ) [inline]
```

## Parameters

x	The key value to assign to the operation.
---	---

**13.28.4.19 screenset\_dn()** [1/2]

```
unsigned int seq64::keys_perform::screenset_dn ( ) const [inline]
```

**13.28.4.20** `screenset_dn()` [2/2]

```
void seq64::keys_perform::screenset_dn (
    unsigned int x ) [inline]
```

**Parameters**

<i>x</i>	The key value to assign to the operation.
----------	---

**13.28.4.21** `set_playing_screenset()` [1/2]

```
unsigned int seq64::keys_perform::set_playing_screenset ( ) const [inline]
```

**13.28.4.22** `set_playing_screenset()` [2/2]

```
void seq64::keys_perform::set_playing_screenset (
    unsigned int x ) [inline]
```

**Parameters**

<i>x</i>	The key value to assign to the operation.
----------	---

**13.28.4.23** `group_on()` [1/2]

```
unsigned int seq64::keys_perform::group_on ( ) const [inline]
```

**13.28.4.24** `group_on()` [2/2]

```
void seq64::keys_perform::group_on (
    unsigned int x ) [inline]
```

**Parameters**

<i>x</i>	The key value to assign to the operation.
----------	---

**13.28.4.25 group\_off()** [1/2]

```
unsigned int seq64::keys_perform::group_off ( ) const [inline]
```

**13.28.4.26 group\_off()** [2/2]

```
void seq64::keys_perform::group_off (
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.27 group\_learn()** [1/2]

```
unsigned int seq64::keys_perform::group_learn ( ) const [inline]
```

**13.28.4.28 group\_learn()** [2/2]

```
void seq64::keys_perform::group_learn (
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.29 start()** [1/2]

```
unsigned int seq64::keys_perform::start ( ) const [inline]
```

**13.28.4.30 start()** [2/2]

```
void seq64::keys_perform::start (
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.31 pause()** [1/2]

```
unsigned int seq64::keys_perform::pause ( ) const [inline]
```

**13.28.4.32 pause()** [2/2]

```
void seq64::keys_perform::pause (  
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.33 pattern\_edit()** [1/2]

```
unsigned int seq64::keys_perform::pattern_edit ( ) const [inline]
```

**13.28.4.34 pattern\_edit()** [2/2]

```
void seq64::keys_perform::pattern_edit (  
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.35 event\_edit()** [1/2]

```
unsigned int seq64::keys_perform::event_edit ( ) const [inline]
```

**13.28.4.36 event\_edit()** [2/2]

```
void seq64::keys_perform::event_edit (
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.37 stop()** [1/2]

```
unsigned int seq64::keys_perform::stop ( ) const [inline]
```

**13.28.4.38 stop()** [2/2]

```
void seq64::keys_perform::stop (
    unsigned int x ) [inline]
```

**Parameters**

x	The key value to assign to the operation.
---	---

**13.28.4.39 song\_mode()** [1/2]

```
unsigned int seq64::keys_perform::song_mode ( ) const [inline]
```

**13.28.4.40 song\_mode()** [2/2]

```
void seq64::keys_perform::song_mode (
    unsigned int key ) [inline]
```

**13.28.4.41 menu\_mode()** [1/2]

```
unsigned int seq64::keys_perform::menu_mode ( ) const [inline]
```

**13.28.4.42 menu\_mode()** [2/2]

```
void seq64::keys_perform::menu_mode (
    unsigned int key ) [inline]
```

**13.28.4.43 follow\_transport()** [1/2]

```
unsigned int seq64::keys_perform::follow_transport ( ) const [inline]
```

**13.28.4.44 follow\_transport()** [2/2]

```
void seq64::keys_perform::follow_transport (
    unsigned int key ) [inline]
```

**13.28.4.45 fast\_forward()** [1/2]

```
unsigned int seq64::keys_perform::fast_forward ( ) const [inline]
```

**13.28.4.46 fast\_forward()** [2/2]

```
void seq64::keys_perform::fast_forward (
    unsigned int key ) [inline]
```

**13.28.4.47 rewind()** [1/2]

```
unsigned int seq64::keys_perform::rewind ( ) const [inline]
```

**13.28.4.48 rewind()** [2/2]

```
void seq64::keys_perform::rewind (
    unsigned int key ) [inline]
```



**13.28.4.49 pointer\_position()** [1/2]

```
unsigned int seq64::keys_perform::pointer_position ( ) const [inline]
```

**13.28.4.50 pointer\_position()** [2/2]

```
void seq64::keys_perform::pointer_position (
    unsigned int key ) [inline]
```

**13.28.4.51 toggle\_mutes()** [1/2]

```
unsigned int seq64::keys_perform::toggle_mutes ( ) const [inline]
```

**13.28.4.52 toggle\_mutes()** [2/2]

```
void seq64::keys_perform::toggle_mutes (
    unsigned int key ) [inline]
```

**13.28.4.53 toggle\_jack()** [1/2]

```
unsigned int seq64::keys_perform::toggle_jack ( ) const [inline]
```

**13.28.4.54 toggle\_jack()** [2/2]

```
void seq64::keys_perform::toggle_jack (
    unsigned int key ) [inline]
```

**13.28.4.55 tap\_bpm()** [1/2]

```
unsigned int seq64::keys_perform::tap_bpm ( ) const [inline]
```

**13.28.4.56 tap\_bpm()** [2/2]

```
void seq64::keys_perform::tap_bpm (
    unsigned int key ) [inline]
```

**13.28.4.57 show\_ui\_sequence\_key()** [1/2]

```
bool seq64::keys_perform::show_ui_sequence_key ( ) const [inline]
```

Used in mainwid, options, optionsfile, userfile, and perform.

**13.28.4.58 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.
-------------	--

**13.28.4.59 show\_ui\_sequence\_number()** [1/2]

```
bool seq64::keys_perform::show_ui_sequence_number ( ) const [inline]
```

Used in mainwid, options, optionsfile, userfile, and perform.

**13.28.4.60 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.
-------------	--

**13.28.4.61 get\_key\_events()**

```
SlotMap& seq64::keys_perform::get_key_events ( ) [inline]
```

**13.28.4.62 get\_key\_groups()**

```
SlotMap& seq64::keys_perform::get_key_groups ( ) [inline]
```

**13.28.4.63 get\_key\_events\_rev()**

```
RevSlotMap& seq64::keys_perform::get_key_events_rev ( ) [inline]
```

**13.28.4.64 get\_key\_groups\_rev()**

```
RevSlotMap& seq64::keys_perform::get_key_groups_rev ( ) [inline]
```

**13.28.4.65 lookup\_keyevent\_key()**

```
unsigned int seq64::keys_perform::lookup_keyevent_key (
    long seqnum ) [inline]
```

**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 question mark character is returned. Not checked for maximum!
---------------	---

**13.28.4.66 lookup\_keyevent\_seq()**

```
long seq64::keys_perform::lookup_keyevent_seq (
    unsigned int keycode ) [inline]
```

**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.
----------------	--

**13.28.4.67 lookup\_keygroup\_key()**

```
unsigned int seq64::keys_perform::lookup_keygroup_key (
    long groupnum ) [inline]
```

## Parameters

<i>groupnum</i>	Provides the group number to look up in the reverse key map for groups. If the count for this value is 0, then a question mark character is returned.
-----------------	---

**13.28.4.68** `lookup_keygroup_group()`

```
long seq64::keys_perform::lookup_keygroup_group (
    unsigned int keycode ) [inline]
```

## 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.
----------------	--

**13.28.4.69** `key_name()`

```
std::string seq64::keys_perform::key_name (
    unsigned int 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](#).

**13.28.4.70** `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](#).

**13.28.4.71 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](#).

**13.28.4.72 set\_key\_event()**

```
void seq64::keys_perform::set_key_event (
    unsigned int keycode,
    long 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.

**13.28.4.73 set\_key\_group()**

```
void seq64::keys_perform::set_key_group (
    unsigned int keycode,
    long 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.

**13.28.4.74 at\_bpm\_up()**

```
unsigned int* 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.

**13.28.4.75 at\_bpm\_dn()**

```
unsigned int* seq64::keys_perform::at_bpm_dn ( ) [inline], [protected]
```

Address getter for the bpm\_dn operation.

**13.28.4.76 at\_replace()**

```
unsigned int* seq64::keys_perform::at_replace ( ) [inline], [protected]
```

Address getter for the replace operation.

**13.28.4.77 at\_queue()**

```
unsigned int* seq64::keys_perform::at_queue ( ) [inline], [protected]
```

Address getter for the queue operation.

**13.28.4.78 at\_keep\_queue()**

```
unsigned int* seq64::keys_perform::at_keep_queue ( ) [inline], [protected]
```

Address getter for the keep\_queue operation.

**13.28.4.79 at\_snapshot\_1()**

```
unsigned int* seq64::keys_perform::at_snapshot_1 ( ) [inline], [protected]
```

Address getter for the snapshot\_1 operation.

**13.28.4.80 at\_snapshot\_2()**

```
unsigned int* seq64::keys_perform::at_snapshot_2 ( ) [inline], [protected]
```

Address getter for the snapshot\_2 operation.

**13.28.4.81 at\_screenset\_up()**

```
unsigned int* seq64::keys_perform::at_screenset_up ( ) [inline], [protected]
```

Address getter for the screenset\_up operation.

**13.28.4.82 at\_screenset\_dn()**

```
unsigned int* seq64::keys_perform::at_screenset_dn ( ) [inline], [protected]
```

Address getter for the screenset\_dn operation.

**13.28.4.83 at\_set\_playing\_screenset()**

```
unsigned int* seq64::keys_perform::at_set_playing_screenset ( ) [inline], [protected]
```

Address getter for the set\_playing\_screenset operation.

**13.28.4.84 at\_group\_on()**

```
unsigned int* seq64::keys_perform::at_group_on ( ) [inline], [protected]
```

Address getter for the group\_on operation.

**13.28.4.85 at\_group\_off()**

```
unsigned int* seq64::keys_perform::at_group_off ( ) [inline], [protected]
```

Address getter for the group\_off operation.

**13.28.4.86 at\_group\_learn()**

```
unsigned int* seq64::keys_perform::at_group_learn ( ) [inline], [protected]
```

Address getter for the group\_learn operation.

**13.28.4.87 at\_start()**

```
unsigned int* seq64::keys_perform::at_start ( ) [inline], [protected]
```

Address getter for the start operation.

**13.28.4.88 at\_pause()**

```
unsigned int* seq64::keys_perform::at_pause ( ) [inline], [protected]
```

Address getter for the pause operation.

**13.28.4.89 at\_song\_mode()**

```
unsigned int* seq64::keys_perform::at_song_mode ( ) [inline], [protected]
```

Address getter for the song-mode operation.

**13.28.4.90 at\_toggle\_jack()**

```
unsigned int* seq64::keys_perform::at_toggle_jack ( ) [inline], [protected]
```

Address getter for the toggle-jack operation.

**13.28.4.91 at\_menu\_mode()**

```
unsigned int* seq64::keys_perform::at_menu_mode ( ) [inline], [protected]
```

Address getter for the menu-mode operation.

**13.28.4.92 at\_follow\_transport()**

```
unsigned int* seq64::keys_perform::at_follow_transport ( ) [inline], [protected]
```

Address getter for the follow-transport operation.

**13.28.4.93 at\_fast\_forward()**

```
unsigned int* seq64::keys_perform::at_fast_forward ( ) [inline], [protected]
```

Address getter for the fast-forward operation.

**13.28.4.94 at\_rewind()**

```
unsigned int* seq64::keys_perform::at_rewind ( ) [inline], [protected]
```

Address getter for the rewind operation.

**13.28.4.95 at\_pointer\_position()**

```
unsigned int* seq64::keys_perform::at_pointer_position ( ) [inline], [protected]
```

Address getter for the pointer operation.

**13.28.4.96 at\_toggle\_mutes()**

```
unsigned int* seq64::keys_perform::at_toggle_mutes ( ) [inline], [protected]
```

Address getter for the toggle-mutes operation.

**13.28.4.97 at\_tap\_bpm()**

```
unsigned int* seq64::keys_perform::at_tap_bpm ( ) [inline], [protected]
```

Address getter for the tap\_bpm operation.

**13.28.4.98 at\_pattern\_edit()**

```
unsigned int* seq64::keys_perform::at_pattern_edit ( ) [inline], [protected]
```

Address getter for the pattern-edit operation.



#### 13.28.4.99 at\_event\_edit()

```
unsigned int* seq64::keys_perform::at_event_edit ( ) [inline], [protected]
```

Address getter for the event-edit operation.

#### 13.28.4.100 at\_stop()

```
unsigned int* seq64::keys_perform::at_stop ( ) [inline], [protected]
```

Address getter for the stop operation.

#### 13.28.4.101 at\_show\_ui\_sequence\_key()

```
bool* seq64::keys_perform::at_show_ui_sequence_key ( ) [inline], [protected]
```

Address getter for the show\_ui\_sequence\_key value.

#### 13.28.4.102 at\_show\_ui\_sequence\_number()

```
bool* seq64::keys_perform::at_show_ui_sequence_number ( ) [inline], [protected]
```

Address getter for the show\_ui\_sequence\_number value.

### 13.28.5 Friends And Related Function Documentation

#### 13.28.5.1 options

```
friend class options [friend]
```

#### 13.28.5.2 perform

```
friend class perform [friend]
```

#### 13.28.5.3 optionsfile

```
friend class optionsfile [friend]
```

## 13.28.6 Field Documentation

### 13.28.6.1 m\_key\_show\_ui\_sequence\_key

```
bool seq64::keys_perform::m_key_show_ui_sequence_key [private]
```

### 13.28.6.2 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.

### 13.28.6.3 m\_key\_events

```
SlotMap seq64::keys_perform::m_key_events [private]
```

Do not access directly, use the set/lookup functions declared below.

### 13.28.6.4 m\_key\_groups

```
SlotMap seq64::keys_perform::m_key_groups [private]
```

Do not access directly, use the set/lookup functions declared below.

### 13.28.6.5 m\_key\_events\_rev

```
RevSlotMap seq64::keys_perform::m_key_events_rev [private]
```

Do not access directly, use the set/lookup functions declared below.

### 13.28.6.6 m\_key\_groups\_rev

```
RevSlotMap seq64::keys_perform::m_key_groups_rev [private]
```

Do not access directly, use the set/lookup functions declared below.

### 13.28.6.7 m\_key\_bpm\_up

```
unsigned int 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!!!

**13.28.6.8 m\_key\_bpm\_dn**

```
unsigned int seq64::keys_perform::m_key_bpm_dn [private]
```

**13.28.6.9 m\_key\_replace**

```
unsigned int seq64::keys_perform::m_key_replace [private]
```

**13.28.6.10 m\_key\_queue**

```
unsigned int seq64::keys_perform::m_key_queue [private]
```

**13.28.6.11 m\_key\_keep\_queue**

```
unsigned int seq64::keys_perform::m_key_keep_queue [private]
```

**13.28.6.12 m\_key\_snapshot\_1**

```
unsigned int seq64::keys_perform::m_key_snapshot_1 [private]
```

**13.28.6.13 m\_key\_snapshot\_2**

```
unsigned int seq64::keys_perform::m_key_snapshot_2 [private]
```

**13.28.6.14 m\_key\_screenset\_up**

```
unsigned int seq64::keys_perform::m_key_screenset_up [private]
```

**13.28.6.15 m\_key\_screenset\_dn**

```
unsigned int seq64::keys_perform::m_key_screenset_dn [private]
```

**13.28.6.16 m\_key\_set\_playing\_screensset**

```
unsigned int seq64::keys_perform::m_key_set_playing_screensset [private]
```

**13.28.6.17 m\_key\_group\_on**

```
unsigned int seq64::keys_perform::m_key_group_on [private]
```

**13.28.6.18 m\_key\_group\_off**

```
unsigned int seq64::keys_perform::m_key_group_off [private]
```

**13.28.6.19 m\_key\_group\_learn**

```
unsigned int seq64::keys_perform::m_key_group_learn [private]
```

**13.28.6.20 m\_key\_start**

```
unsigned int seq64::keys_perform::m_key_start [private]
```

**13.28.6.21 m\_key\_pause**

```
unsigned int seq64::keys_perform::m_key_pause [private]
```

**13.28.6.22 m\_key\_song\_mode**

```
unsigned int seq64::keys_perform::m_key_song_mode [private]
```

**13.28.6.23 m\_key\_toggle\_jack**

```
unsigned int seq64::keys_perform::m_key_toggle_jack [private]
```

**13.28.6.24 m\_key\_menu\_mode**

```
unsigned int seq64::keys_perform::m_key_menu_mode [private]
```

**13.28.6.25 m\_key\_follow\_transport**

```
unsigned int seq64::keys_perform::m_key_follow_transport [private]
```

**13.28.6.26 m\_key\_rewind**

```
unsigned int seq64::keys_perform::m_key_rewind [private]
```

**13.28.6.27 m\_key\_fast\_forward**

```
unsigned int seq64::keys_perform::m_key_fast_forward [private]
```

**13.28.6.28 m\_key\_pointer\_position**

```
unsigned int seq64::keys_perform::m_key_pointer_position [private]
```

**13.28.6.29 m\_key\_toggle\_mutes**

```
unsigned int seq64::keys_perform::m_key_toggle_mutes [private]
```

**13.28.6.30 m\_key\_tap\_bpm**

```
unsigned int seq64::keys_perform::m_key_tap_bpm [private]
```

**13.28.6.31 m\_key\_pattern\_edit**

```
unsigned int seq64::keys_perform::m_key_pattern_edit [private]
```

#### 13.28.6.32 m\_key\_event\_edit

```
unsigned int seq64::keys_perform::m_key_event_edit [private]
```

#### 13.28.6.33 m\_key\_stop

```
unsigned int seq64::keys_perform::m_key_stop [private]
```

### 13.29 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 int key) const

- 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

### 13.29.1 Detailed Description

It has way too many data members, many of the public. Might be ripe for refactoring.

### 13.29.2 Constructor & Destructor Documentation

#### 13.29.2.1 [keys\\_perform\\_gtk2\(\)](#)

```
seq64::keys_perform_gtk2::keys_perform_gtk2 ( )
```

#### 13.29.2.2 [~keys\\_perform\\_gtk2\(\)](#)

```
seq64::keys_perform_gtk2::~~keys_perform_gtk2 ( ) [virtual]
```

No action.

### 13.29.3 Member Function Documentation

#### 13.29.3.1 [key\\_name\(\)](#)

```
virtual std::string seq64::keys_perform_gtk2::key_name (
    unsigned int key ) const [inline], [virtual]
```

Reimplemented from [seq64::keys\\_perform](#).



### 13.29.3.2 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.

Reimplemented from [seq64::keys\\_perform](#).

### 13.29.3.3 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.

Reimplemented from [seq64::keys\\_perform](#).

## 13.30 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 int [kpt\\_bpm\\_up](#)
- unsigned int [kpt\\_bpm\\_dn](#)
- unsigned int [kpt\\_screenset\\_up](#)
- unsigned int [kpt\\_screenset\\_dn](#)
- unsigned int [kpt\\_set\\_playing\\_screenset](#)
- unsigned int [kpt\\_group\\_on](#)
- unsigned int [kpt\\_group\\_off](#)
- unsigned int [kpt\\_group\\_learn](#)
- unsigned int [kpt\\_replace](#)
- unsigned int [kpt\\_queue](#)
- unsigned int [kpt\\_keep\\_queue](#)
- unsigned int [kpt\\_snapshot\\_1](#)
- unsigned int [kpt\\_snapshot\\_2](#)
- unsigned int [kpt\\_start](#)
- unsigned int [kpt\\_stop](#)
- bool [kpt\\_show\\_ui\\_sequence\\_key](#)
- bool [kpt\\_show\\_ui\\_sequence\\_number](#)
- unsigned int [kpt\\_pattern\\_edit](#)
- unsigned int [kpt\\_event\\_edit](#)
- unsigned int [kpt\\_tap\\_bpm](#)
- unsigned int [kpt\\_pause](#)
- unsigned int [kpt\\_song\\_mode](#)
- unsigned int [kpt\\_toggle\\_jack](#)
- unsigned int [kpt\\_menu\\_mode](#)
- unsigned int [kpt\\_follow\\_transport](#)
- unsigned int [kpt\\_fast\\_forward](#)
- unsigned int [kpt\\_rewind](#)
- unsigned int [kpt\\_pointer\\_position](#)
- unsigned int [kpt\\_toggle\\_mutes](#)

### 13.30.1 Field Documentation

#### 13.30.1.1 kpt\_bpm\_up

unsigned int seq64::keys\_perform\_transfer::kpt\_bpm\_up

#### 13.30.1.2 kpt\_bpm\_dn

unsigned int seq64::keys\_perform\_transfer::kpt\_bpm\_dn

#### 13.30.1.3 kpt\_screensset\_up

unsigned int seq64::keys\_perform\_transfer::kpt\_screensset\_up

#### 13.30.1.4 kpt\_screensset\_dn

unsigned int seq64::keys\_perform\_transfer::kpt\_screensset\_dn

#### 13.30.1.5 kpt\_set\_playing\_screensset

unsigned int seq64::keys\_perform\_transfer::kpt\_set\_playing\_screensset

#### 13.30.1.6 kpt\_group\_on

unsigned int seq64::keys\_perform\_transfer::kpt\_group\_on

#### 13.30.1.7 kpt\_group\_off

unsigned int seq64::keys\_perform\_transfer::kpt\_group\_off

**13.30.1.8 kpt\_group\_learn**

unsigned int seq64::keys\_perform\_transfer::kpt\_group\_learn

**13.30.1.9 kpt\_replace**

unsigned int seq64::keys\_perform\_transfer::kpt\_replace

**13.30.1.10 kpt\_queue**

unsigned int seq64::keys\_perform\_transfer::kpt\_queue

**13.30.1.11 kpt\_keep\_queue**

unsigned int seq64::keys\_perform\_transfer::kpt\_keep\_queue

**13.30.1.12 kpt\_snapshot\_1**

unsigned int seq64::keys\_perform\_transfer::kpt\_snapshot\_1

**13.30.1.13 kpt\_snapshot\_2**

unsigned int seq64::keys\_perform\_transfer::kpt\_snapshot\_2

**13.30.1.14 kpt\_start**

unsigned int seq64::keys\_perform\_transfer::kpt\_start

**13.30.1.15 kpt\_stop**

unsigned int seq64::keys\_perform\_transfer::kpt\_stop

**13.30.1.16 kpt\_show\_ui\_sequence\_key**

```
bool seq64::keys_perform_transfer::kpt_show_ui_sequence_key
```

**13.30.1.17 kpt\_show\_ui\_sequence\_number**

```
bool seq64::keys_perform_transfer::kpt_show_ui_sequence_number
```

**13.30.1.18 kpt\_pattern\_edit**

```
unsigned int seq64::keys_perform_transfer::kpt_pattern_edit
```

**13.30.1.19 kpt\_event\_edit**

```
unsigned int seq64::keys_perform_transfer::kpt_event_edit
```

**13.30.1.20 kpt\_tap\_bpm**

```
unsigned int seq64::keys_perform_transfer::kpt_tap_bpm
```

**13.30.1.21 kpt\_pause**

```
unsigned int seq64::keys_perform_transfer::kpt_pause
```

**13.30.1.22 kpt\_song\_mode**

```
unsigned int seq64::keys_perform_transfer::kpt_song_mode
```

**13.30.1.23 kpt\_toggle\_jack**

```
unsigned int seq64::keys_perform_transfer::kpt_toggle_jack
```

#### 13.30.1.24 kpt\_menu\_mode

unsigned int seq64::keys\_perform\_transfer::kpt\_menu\_mode

#### 13.30.1.25 kpt\_follow\_transport

unsigned int seq64::keys\_perform\_transfer::kpt\_follow\_transport

#### 13.30.1.26 kpt\_fast\_forward

unsigned int seq64::keys\_perform\_transfer::kpt\_fast\_forward

#### 13.30.1.27 kpt\_rewind

unsigned int seq64::keys\_perform\_transfer::kpt\_rewind

#### 13.30.1.28 kpt\_pointer\_position

unsigned int seq64::keys\_perform\_transfer::kpt\_pointer\_position

#### 13.30.1.29 kpt\_toggle\_mutes

unsigned int seq64::keys\_perform\_transfer::kpt\_toggle\_mutes

## 13.31 seq64::keystroke Class Reference

Encapsulates any practical keystroke.

## Public Member Functions

- [keystroke](#) ()  
*The default constructor for class keystroke.*
- [keystroke](#) (unsigned int [key](#), bool press=SEQ64\_KEYSTROKE\_PRESS, int modkey=int([SEQ64\\_NO\\_MODIFIER](#) SK))  
*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 int ch=SEQ64\_KEYSTROKE\_BAD\_VALUE) const  
*'Getter' function for member m\_key to test letters, handles ASCII only.*
- bool [is](#) (unsigned int ch) 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 int [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 int [m\\_key](#)  
*The key that was pressed or released.*
- [seq\\_modifier\\_t](#) [m\\_modifier](#)  
*The optional modifier value.*

### 13.31.1 Detailed Description

Useful in passing more generic events to non-GUI classes.

### 13.31.2 Constructor & Destructor Documentation

13.31.2.1 `keystroke()` [1/3]

```
seq64::keystroke::keystroke ( )
```

13.31.2.2 `keystroke()` [2/3]

```
seq64::keystroke::keystroke (
    unsigned int 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 <code>gdk_basic_keys</code> module. Common mask values are <code>SEQ64_SHIFT_MASK</code> , <code>SEQ64_CONTROL_MASK</code> , <code>SEQ64_MOD1_MASK</code> , and <code>SEQ64_MOD4_MASK</code> . If no modifier, this value is <code>SEQ64_NO_MASK</code> .

13.31.2.3 `keystroke()` [3/3]

```
seq64::keystroke::keystroke (
    const keystroke & rhs )
```

## Parameters

<i>rhs</i>	The object to be copied.
------------	--------------------------

## 13.31.3 Member Function Documentation

13.31.3.1 `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.

**13.31.3.2 is\_press()**

```
bool seq64::keystroke::is_press ( ) const [inline]
```

**13.31.3.3 is\_letter()**

```
bool seq64::keystroke::is_letter (
    unsigned int 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****13.31.3.4 is()**

```
bool seq64::keystroke::is (
    unsigned int ch ) const [inline]
```

**Parameters**

<i>ch</i>	The character to be tested.
-----------	-----------------------------

**Returns**

Returns true if `m_key == ch`.

**13.31.3.5 is\_delete()**

```
bool seq64::keystroke::is_delete ( ) const [inline]
```



**13.31.3.6 key()**

```
unsigned int seq64::keystroke::key ( ) const [inline]
```

**13.31.3.7 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.

**13.31.3.8 modifier()**

```
seq_modifier_t seq64::keystroke::modifier ( ) const [inline]
```

**13.31.3.9 mod\_control()**

```
bool seq64::keystroke::mod_control ( ) const [inline]
```

**13.31.3.10 mod\_control\_shift()**

```
bool seq64::keystroke::mod_control_shift ( ) const [inline]
```

**13.31.3.11 mod\_super()**

```
bool seq64::keystroke::mod_super ( ) const [inline]
```

### 13.31.4 Field Documentation

#### 13.31.4.1 m\_is\_press

```
bool seq64::keystroke::m_is_press [private]
```

See the SEQ64\_KEYSTROKE\_PRESS and SEQ64\_KEYSTROKE\_RELEASE readability macros.

#### 13.31.4.2 m\_key

```
unsigned int 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.

#### 13.31.4.3 m\_modifier

```
seq_modifier_t seq64::keystroke::m_modifier [private]
```

Note that SEQ64\_NO\_MASK is our word for 0, meaning "no modifier".

## 13.32 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.*

### 13.32.1 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.

### 13.32.2 Constructor & Destructor Documentation

#### 13.32.2.1 `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.

### 13.32.3 Member Function Documentation

#### 13.32.3.1 `set_alsa_client_id()`

```
void seq64::lash::set_alsa_client_id (
    int id )
```

/param *id* The ALSA client ID to be set.

### 13.32.3.2 start()

```
void seq64::lash::start ( )
```

### 13.32.3.3 process\_events()

```
bool seq64::lash::process_events ( )
```

#### Returns

Always returns true.

### 13.32.3.4 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.

### 13.32.3.5 handle\_event()

```
void seq64::lash::handle_event (
    lash_event_t * ev ) [private]
```

#### Parameters

ev	Provides the event to be handled.
----	-----------------------------------

### 13.32.3.6 handle\_config()

```
void seq64::lash::handle_config (
    lash_config_t * conf ) [private]
```

Currently incomplete.

## Parameters

<i>conf</i>	Provides the configuration item to handle.
-------------	--

### 13.32.4 Field Documentation

#### 13.32.4.1 m\_perform

```
perform& seq64::lash::m_perform [private]
```

#### 13.32.4.2 m\_client

```
lash_client_t* seq64::lash::m_client [private]
```

#### 13.32.4.3 m\_lash\_args

```
lash_args_t* seq64::lash::m_lash_args [private]
```

#### 13.32.4.4 m\_is\_lash\_supported

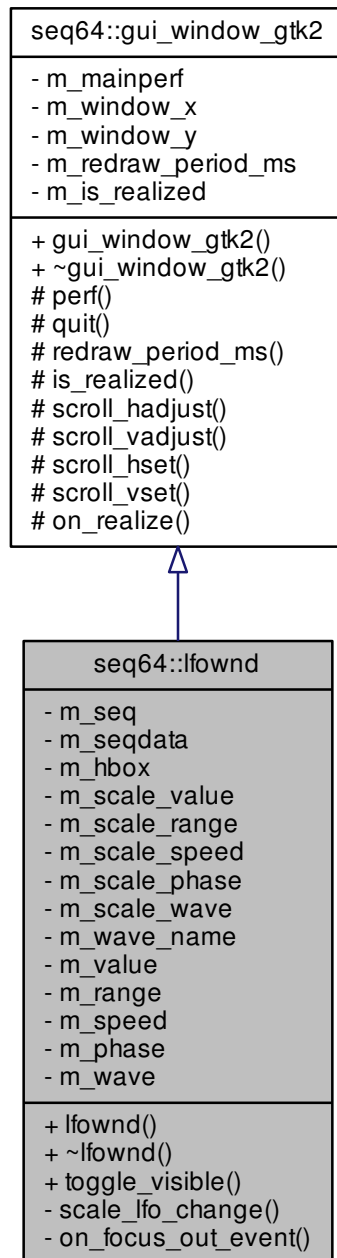
```
bool seq64::lash::m_is_lash_supported [private]
```

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.

### 13.33 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 ()`
- void `toggle_visible ()`

### Private Member Functions

- void `scale_lfo_change ()`
- bool `on_focus_out_event (GdkEventFocus *p0)`

### 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

#### 13.33.1 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.

### 13.33.2 Constructor & Destructor Documentation

#### 13.33.2.1 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.

#### 13.33.2.2 ~lfownd()

```
seq64::lfownd::~~lfownd ( ) [virtual]
```

### 13.33.3 Member Function Documentation

#### 13.33.3.1 toggle\_visible()

```
void seq64::lfownd::toggle_visible ( )
```

#### 13.33.3.2 scale\_lfo\_change()

```
void seq64::lfownd::scale_lfo_change ( ) [private]
```

#### 13.33.3.3 on\_focus\_out\_event()

```
bool seq64::lfownd::on_focus_out_event (
    GdkEventFocus * p0 ) [private]
```



### 13.33.4 Field Documentation

#### 13.33.4.1 m\_seq

`sequence& seq64::lfownd::m_seq [private]`

#### 13.33.4.2 m\_seqdata

`seqdata& seq64::lfownd::m_seqdata [private]`

#### 13.33.4.3 m\_hbox

`Gtk::HBox* seq64::lfownd::m_hbox [private]`

#### 13.33.4.4 m\_scale\_value

`Gtk::VScale* seq64::lfownd::m_scale_value [private]`

#### 13.33.4.5 m\_scale\_range

`Gtk::VScale* seq64::lfownd::m_scale_range [private]`

#### 13.33.4.6 m\_scale\_speed

`Gtk::VScale* seq64::lfownd::m_scale_speed [private]`

#### 13.33.4.7 m\_scale\_phase

`Gtk::VScale* seq64::lfownd::m_scale_phase [private]`

**13.33.4.8 m\_scale\_wave**

```
Gtk::VScale* seq64::lfownd::m_scale_wave [private]
```

**13.33.4.9 m\_wave\_name**

```
Gtk::Label* seq64::lfownd::m_wave_name [private]
```

**13.33.4.10 m\_value**

```
double seq64::lfownd::m_value [private]
```

**13.33.4.11 m\_range**

```
double seq64::lfownd::m_range [private]
```

**13.33.4.12 m\_speed**

```
double seq64::lfownd::m_speed [private]
```

**13.33.4.13 m\_phase**

```
double seq64::lfownd::m_phase [private]
```

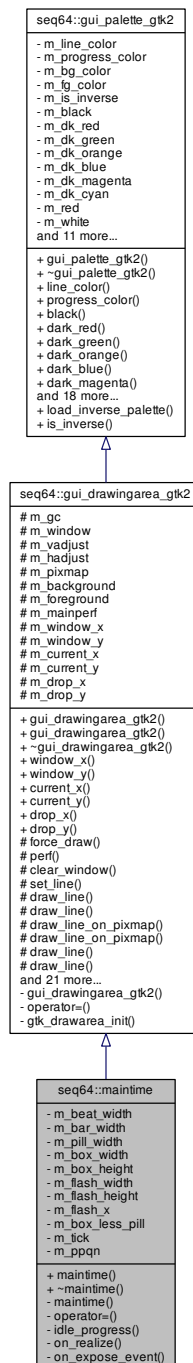
**13.33.4.14 m\_wave**

```
wave_type_t seq64::lfownd::m_wave [private]
```

## 13.34 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.*

- `void on_realize ()`

*Handles realization of the window.*

- `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

### 13.34.1 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.

### 13.34.2 Constructor & Destructor Documentation

#### 13.34.2.1 maintime() [1/2]

```
seq64::maintime::maintime (
    const maintime & ) [private]
```

#### 13.34.2.2 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.

#### 13.34.2.3 ~maintime()

```
virtual seq64::maintime::~~maintime ( ) [inline], [virtual]
```

### 13.34.3 Member Function Documentation

#### 13.34.3.1 operator=()

```
maintime& seq64::maintime::operator= (
    const maintime & ) [private]
```

#### 13.34.3.2 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 <a href="#">mainwnd()</a> , in its timer callback.
--------------	--

**Returns**

Always returns 1 (it used to return "true!").

**13.34.3.3 on\_realize()**

```
void seq64::maintime::on_realize ( ) [private]
```

It performs the base class's [on\\_realize\(\)](#) function. It then allocates some additional resources: a window, a GC (?), and it clears the window. Then it sets the default size of the window, specified by GUI constructor parameters.

**13.34.3.4 on\_expose\_event()**

```
bool seq64::maintime::on_expose_event (
    GdkEventExpose * a_e ) [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.)

**13.34.4 Friends And Related Function Documentation****13.34.4.1 mainwnd**

```
friend class mainwnd [friend]
```

**13.34.5 Field Documentation****13.34.5.1 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\(\)](#).

#### 13.34.5.2 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()`.

#### 13.34.5.3 m\_pill\_width

```
const int seq64::maintime::m_pill_width [private]
```

#### 13.34.5.4 m\_box\_width

```
const int seq64::maintime::m_box_width [private]
```

This item absolutely depends on the main window being non-resizable.

#### 13.34.5.5 m\_box\_height

```
const int seq64::maintime::m_box_height [private]
```

This item absolutely depends on the main window being non-resizable.

#### 13.34.5.6 m\_flash\_width

```
const int seq64::maintime::m_flash_width [private]
```

Just a bit smaller than `m_box_width`.

#### 13.34.5.7 m\_flash\_height

```
const int seq64::maintime::m_flash_height [private]
```

Just a bit smaller than `m_box_width`.

#### 13.34.5.8 m\_flash\_x

```
const int seq64::maintime::m_flash_x [private]
```

#### 13.34.5.9 m\_box\_less\_pill

```
const int seq64::maintime::m_box_less_pill [private]
```

#### 13.34.5.10 m\_tick

```
midipulse seq64::maintime::m_tick [private]
```

It might actually be useful after all. And the overhead is tiny.

#### 13.34.5.11 m\_ppqn

```
int seq64::maintime::m_ppqn [private]
```

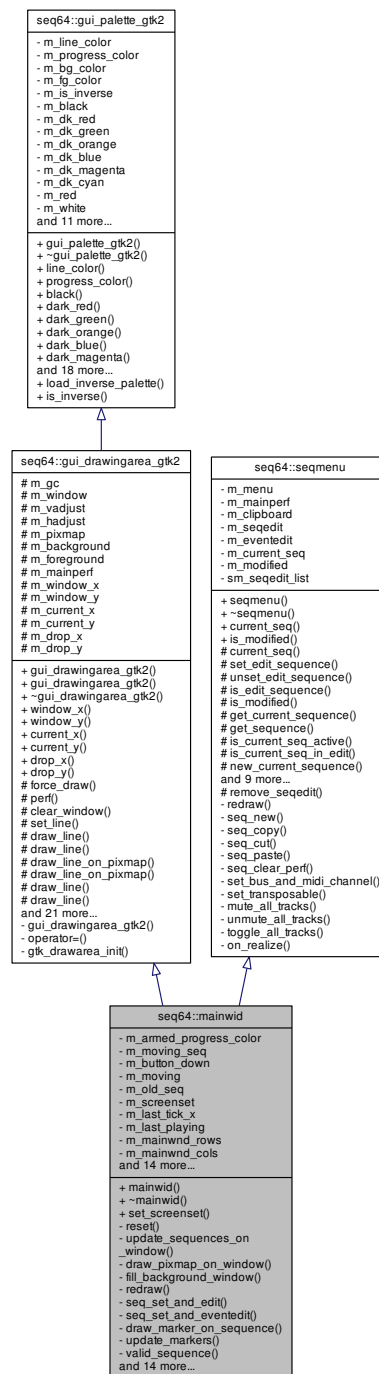
While this is effectively a constant for the duration of a tune, it might change as different tunes are loaded.

### 13.35 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)  
*This constructor sets all of the members.*
- virtual [~mainwid](#) ()  
*A rote destructor.*
- void [set\\_screenset](#) (int ss, bool setperf=false)  
*Set the current screen-set.*

## Private Member Functions

- 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.*
- 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 coordiinates 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.*
- long [m\\_last\\_playing](#) [c\_max\_sequence]  
*Holds the last playing tick for each sequence.*
- 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.*
- int [m\\_mainwnd\\_cols](#)  
*Number of columns, unused in settings.*
- int [m\\_seqarea\\_x](#)  
*Roughly with width of the main window.*
- int [m\\_seqarea\\_y](#)  
*Roughly with height of the main window.*
- int [m\\_seqarea\\_seq\\_x](#)  
*To be determined.*
- int [m\\_seqarea\\_seq\\_y](#)  
*To be determined.*
- int [m\\_mainwid\\_x](#)  
*To be determined.*
- int [m\\_mainwid\\_y](#)  
*To be determined.*
- int [m\\_mainwid\\_border](#)  
*Main-window border, unused setting.*
- int [m\\_mainwid\\_spacing](#)  
*Main-window spacing, unused setting.*
- int [m\\_text\\_size\\_x](#)  
*Text width, varies with font in use.*
- int [m\\_text\\_size\\_y](#)  
*Text height, varies with font in use.*
- int [m\\_max\\_sets](#)  
*The maximum number of sets, use all over.*

- int [m\\_screenset\\_slots](#)  
*Provides a convenience variable for avoiding multiplications.*
- int [m\\_screenset\\_offset](#)  
*Provides a convenience variable for avoiding multiplications.*
- 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 `mainwid :: update_sequences_on_window()`, if the global `mainwid` object exists.*

## Additional Inherited Members

### 13.35.1 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.

### 13.35.2 Constructor & Destructor Documentation

#### 13.35.2.1 `mainwid()`

```
seq64::mainwid::mainwid (
    perform & p )
```

And it asks for a size of `c_mainwid_x` by `c_mainwid_y`. 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 <code>perform</code> object.
----------	--

#### 13.35.2.2 `~mainwid()`

```
seq64::mainwid::~~mainwid ( ) [virtual]
```

### 13.35.3 Member Function Documentation

#### 13.35.3.1 set\_screenset()

```
void seq64::mainwid::set_screenset (
    int ss,
    bool setperf = false )
```

The clamping algorithm for the screeset is a bit weird: if less than 0, we set m\_screenset to its maximum, and if greater than the maximum, we set it to its minimum. Not sure if this matters.

Note that m\_screenset\_slots = m\_mainwnd\_rows \* m\_mainwnd\_cols.

We will likely replace this with [perform::set\\_screenset\(\)](#), which recapitulates the 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(). Defaults to false. It might be better if it defaults to true.

#### 13.35.3.2 reset()

```
void seq64::mainwid::reset ( ) [inline], [private]
```

#### 13.35.3.3 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.

#### 13.35.3.4 draw\_pixmap\_on\_window()

```
void seq64::mainwid::draw_pixmap_on_window ( ) [inline], [private]
```

### 13.35.3.5 fill\_background\_window()

```
void seq64::mainwid::fill_background_window ( ) [inline], [private]
```

### 13.35.3.6 redraw()

```
void seq64::mainwid::redraw (
    int seqnum ) [private], [virtual]
```

#### Parameters

<i>seqnum</i>	Provides the number of the sequence to draw.
---------------	--

Implements [seq64::seqmenu](#).

### 13.35.3.7 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](#).

### 13.35.3.8 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](#).

### 13.35.3.9 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_last_tick()</code> value. This causes correct stop/pause/play progress-bar behavior in each pattern slot.

13.35.3.10 `update_markers()`

```
void seq64::mainwid::update_markers (
    int tick ) [private]
```

## Parameters

<i>tick</i>	Starting point for drawing the markers.
-------------	---

13.35.3.11 `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.

13.35.3.12 `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.
---------------	---

**13.35.3.13 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.

**13.35.3.14 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.
---------------	--

**13.35.3.15 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.



## 13.35.3.16 timeout()

```
int seq64::mainwid::timeout ( ) [private]
```

**Todo** We should use this callback to display the current time in the playback.

## Returns

Always returns true.

## 13.35.3.17 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.

## 13.35.3.18 select\_fg\_bg\_colors()

```
void seq64::mainwid::select_fg_bg_colors (
    int seqnum ) [private]
```

## 13.35.3.19 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 was to do this!

## 13.35.3.20 on\_expose\_event()

```
bool seq64::mainwid::on_expose_event (
    GdkEventExpose * ev ) [private]
```

#### Parameters

<i>ev</i>	The expose event.
-----------	-------------------

#### Returns

Always returns true.

#### 13.35.3.21 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

<i>ev</i>	Provides the parameters of the button event.
-----------	--

#### Returns

Always returns true.

#### 13.35.3.22 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

<code>ev</code>	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).

#### 13.35.3.23 `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

<code>ev</code>	Provides the parameters of the button event.
-----------------	--

#### Returns

Always returns true.

#### 13.35.3.24 `on_focus_in_event()`

```
bool seq64::mainwid::on_focus_in_event (
    GdkEventFocus * ) [private]
```

Just sets the `Gtk::HAS_FOCUS` flag.

#### Returns

Always returns false.

#### 13.35.3.25 `on_focus_out_event()`

```
bool seq64::mainwid::on_focus_out_event (
    GdkEventFocus * ) [private]
```

Just unsets the `Gtk::HAS_FOCUS` flag.

#### Returns

Always returns false.

### 13.35.4 Friends And Related Function Documentation

#### 13.35.4.1 mainwnd

```
friend class mainwnd [friend]
```

#### 13.35.4.2 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).

### 13.35.5 Field Documentation

#### 13.35.5.1 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.

#### 13.35.5.2 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.

#### 13.35.5.3 m\_button\_down

```
bool seq64::mainwid::m_button_down [private]
```

Used in the drag-and-drop functionality.

#### 13.35.5.4 m\_moving

```
bool seq64::mainwid::m_moving [private]
```

#### 13.35.5.5 m\_old\_seq

```
int seq64::mainwid::m_old_seq [private]
```

#### 13.35.5.6 m\_screenset

```
int seq64::mainwid::m_screenset [private]
```

#### 13.35.5.7 m\_last\_tick\_x

```
long seq64::mainwid::m_last_tick_x[c_max_sequence] [private]
```

#### 13.35.5.8 m\_last\_playing

```
long seq64::mainwid::m_last_playing[c_max_sequence] [private]
```

This doesn't seem to be used anywhere, even though values are logged, so it is macroed out.

#### 13.35.5.9 m\_mainwnd\_rows

```
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.

#### 13.35.5.10 m\_mainwnd\_cols

```
int seq64::mainwid::m_mainwnd_cols [private]
```

#### 13.35.5.11 m\_seqarea\_x

```
int seq64::mainwid::m_seqarea_x [private]
```

#### 13.35.5.12 m\_seqarea\_y

```
int seq64::mainwid::m_seqarea_y [private]
```

**13.35.5.13 m\_seqarea\_seq\_x**

```
int seq64::mainwid::m_seqarea_seq_x [private]
```

**13.35.5.14 m\_seqarea\_seq\_y**

```
int seq64::mainwid::m_seqarea_seq_y [private]
```

**13.35.5.15 m\_mainwid\_x**

```
int seq64::mainwid::m_mainwid_x [private]
```

**13.35.5.16 m\_mainwid\_y**

```
int seq64::mainwid::m_mainwid_y [private]
```

**13.35.5.17 m\_mainwid\_border**

```
int seq64::mainwid::m_mainwid_border [private]
```

**13.35.5.18 m\_mainwid\_spacing**

```
int seq64::mainwid::m_mainwid_spacing [private]
```

**13.35.5.19 m\_text\_size\_x**

```
int seq64::mainwid::m_text_size_x [private]
```

**13.35.5.20 m\_text\_size\_y**

```
int seq64::mainwid::m_text_size_y [private]
```

#### 13.35.5.21 m\_max\_sets

```
int seq64::mainwid::m_max_sets [private]
```

#### 13.35.5.22 m\_screenset\_slots

```
int seq64::mainwid::m_screenset_slots [private]
```

It is equal to `m_mainwnd_rows * m_mainwnd_cols`.

#### 13.35.5.23 m\_screenset\_offset

```
int seq64::mainwid::m_screenset_offset [private]
```

It is equally to `m_screenset_slots * m_screenset`.

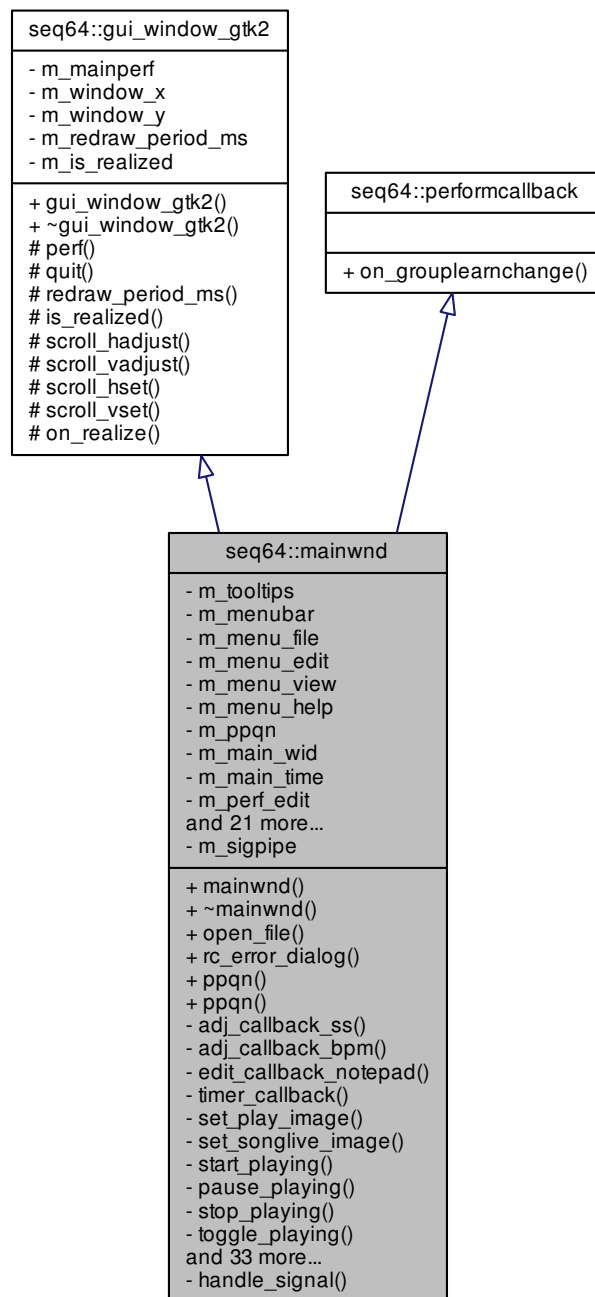
#### 13.35.5.24 m\_progress\_height

```
int seq64::mainwid::m_progress_height [private]
```

## 13.36 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)  
*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.*
- 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 Member Functions

- void `adj_callback_ss` ()  
*This function is the callback for adjusting the 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.*
- bool `timer_callback` ()  
*This function is the GTK timer callback, used to draw our current time and BPM on \_events (the main window).*
- 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.*
- 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.*
- void `apply_song_transpose` ()  
*Apply full song transposition, if enabled.*
- void `update_window_title` ()  
*Updates the title shown in the title bar of the window.*
- 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.*
- 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](#) (bool do\_export=false)  
*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.*
- bool [on\\_delete\\_event](#) (GdkEventAny \*ev)  
*This callback function handles a delete event from ...?*
- 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::Tooltips \* [m\\_tooltips](#)

*A repository for tooltips.*

- 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\\_edit](#)

*The (new) Edit menu entry.*

- Gtk::Menu \* [m\\_menu\\_view](#)

*The View menu entry.*

- Gtk::Menu \* [m\\_menu\\_help](#)

*The Help menu entry.*

- 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.*

- mainwid \* [m\\_main\\_wid](#)

*The biggest sub-components of mainwnd.*

- maintime \* [m\\_main\\_time](#)

*Is this the bar at the top that shows moving squares, also known as "pills"? Why yes, it is.*

- [perfedit](#) \* [m\\_perf\\_edit](#)

*A pointer to the first song/performance editor.*

- [perfedit](#) \* [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\\_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\\_perfedit](#)

*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::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::Adjustment \* [m\\_adjust\\_ss](#)  
*The spin/adjustment controls for the screenset value.*
- Gtk::SpinButton \* [m\\_spinbutton\\_ss](#)  
*Screenset adjustment.*
- 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.*
- 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 [m\\_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.*

## Additional Inherited Members

### 13.36.1 Constructor & Destructor Documentation

#### 13.36.1.1 mainwnd()

```
seq64::mainwnd::mainwnd (
    perform & p,
    bool allowperf2 = true,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

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.

**Todo** Offload most of the work into an initialization function like options does; make the perform parameter a reference; valgrind flags m\_tooltips as lost data, but if we try to manage it ourselves, many more leaks occur.

Edit menu items and their hot keys.

View menu items and their hot keys. It repeats the song editor edit command, just to help those whose muscle memory is already seq32-oriented.

View menu items and their hot keys.

Help menu items

Top panel items, including the logo (updated for the new version of this application) and the "timeline" progress bar.

### 13.36.1.2 ~mainwnd()

```
seq64::mainwnd::~~mainwnd ( ) [virtual]
```

## 13.36.2 Member Function Documentation

### 13.36.2.1 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.
-----------	--

### 13.36.2.2 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.
----------------	--

**13.36.2.3 ppqn()** [1/2]

```
int seq64::mainwnd::ppqn ( ) const [inline]
```

**13.36.2.4 ppqn()** [2/2]

```
void seq64::mainwnd::ppqn (
    int ppqn ) [inline]
```

```
m_ppqn = choose_ppqn(ppqn);
```

**13.36.2.5 handle\_signal()**

```
void seq64::mainwnd::handle_signal (
    int sig ) [static], [private]
```

**13.36.2.6 adj\_callback\_ss()**

```
void seq64::mainwnd::adj_callback_ss ( ) [private]
```

Its sets the 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.

**13.36.2.7 adj\_callback\_bpm()**

```
void seq64::mainwnd::adj_callback_bpm ( ) [private]
```

Let the perform object keep track of modifications.

**13.36.2.8 edit\_callback\_notepad()**

```
void seq64::mainwnd::edit_callback_notepad ( ) [private]
```

Let the perform object keep track of modifications.

## 13.36.2.9 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.

## 13.36.2.10 set\_play\_image()

```
void seq64::mainwnd::set_play_image (
    bool isrunning ) [private]
```

## 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.
------------------	---

## 13.36.2.11 set\_songlive\_image()

```
void seq64::mainwnd::set_songlive_image (
    bool issong ) [private]
```

## Parameters

<i>issong</i>	If true, set the image to the "Song" icon. Otherwise, set it to the "Live" button.
---------------	--

## 13.36.2.12 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.

We still need to see if pause\_key() is workable with Stazed JACK support in force. Doesn't pause at present.

### 13.36.2.13 pause\_playing()

```
void seq64::mainwnd::pause_playing ( ) [private]
```

Currently, it is just the same as [stop\\_playing\(\)](#), but we will get it to work.

### 13.36.2.14 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.

### 13.36.2.15 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.

### 13.36.2.16 learn\_toggle()

```
void seq64::mainwnd::learn_toggle ( ) [inline], [private]
```

Simply forwards the call to [perform::learn\\_toggle\(\)](#).

### 13.36.2.17 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.

### 13.36.2.18 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.



#### 13.36.2.19 enregister\_perfedits()

```
void seq64::mainwnd::enregister_perfedits ( ) [private]
```

#### 13.36.2.20 sequence\_key()

```
void seq64::mainwnd::sequence_key (
    int seq ) [inline], [private]
```

#### 13.36.2.21 apply\_song\_transpose()

```
void seq64::mainwnd::apply_song_transpose ( ) [private]
```

Then reset the perfedit transpose setting to 0.

#### 13.36.2.22 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.

#### 13.36.2.23 toLower()

```
void seq64::mainwnd::toLower (
    std::string & s ) [private]
```

#### 13.36.2.24 file\_new()

```
void seq64::mainwnd::file_new ( ) [inline], [private]
```

#### 13.36.2.25 file\_open()

```
void seq64::mainwnd::file_open ( ) [inline], [private]
```

#### 13.36.2.26 file\_save()

```
void seq64::mainwnd::file_save ( ) [inline], [private]
```

#### 13.36.2.27 set\_song\_mute()

```
void seq64::mainwnd::set_song_mute (
    perform::mute_op_t op ) [inline], [private]
```

#### 13.36.2.28 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.

#### 13.36.2.29 options\_dialog()

```
void seq64::mainwnd::options_dialog ( ) [private]
```

#### 13.36.2.30 jack\_dialog()

```
void seq64::mainwnd::jack_dialog ( ) [private]
```

#### 13.36.2.31 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....

**13.36.2.32 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.

**13.36.2.33 query\_save\_changes()**

```
int seq64::mainwnd::query_save_changes ( ) [private]
```

**13.36.2.34 new\_open\_error\_dialog()**

```
void seq64::mainwnd::new_open_error_dialog ( ) [private]
```

**13.36.2.35 file\_save\_as()**

```
void seq64::mainwnd::file_save_as (
    bool do_export = false ) [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>do_export</i>	If true, then just write out the file and don't change the name of the current file based on the file-name the user selected. The default value of this parameter is false.
------------------	---

**13.36.2.36 file\_exit()**

```
void seq64::mainwnd::file_exit ( ) [private]
```

**13.36.2.37 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.

**13.36.2.38 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.

**13.36.2.39 choose\_file()**

```
void seq64::mainwnd::choose_file ( ) [private]
```

*Change Note* layk 2016-10-11 Issue #43 Added filters for upper-case MIDI-file extensions.

**13.36.2.40 is\_save()**

```
bool seq64::mainwnd::is_save ( ) [private]
```

**13.36.2.41 install\_signal\_handlers()**

```
bool seq64::mainwnd::install_signal_handlers ( ) [private]
```

### 13.36.2.42 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.

### 13.36.2.43 edit\_field\_has\_focus()

```
bool seq64::mainwnd::edit_field_has_focus ( ) const [private]
```

#### Returns

Returns true if one of the three editable/modifiable fields has the keyboard focus.

### 13.36.2.44 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\(\)](#).

### 13.36.2.45 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.

### 13.36.2.46 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.

#### 13.36.2.47 on\_realize()

```
void seq64::mainwnd::on_realize ( ) [private]
```

without first moving the mouse to the main window. Weird with a beard!

#### 13.36.2.48 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](#).

### 13.36.3 Field Documentation

#### 13.36.3.1 m\_sigpipe

```
int seq64::mainwnd::m_sigpipe [static], [private]
```

This static member provides a couple of pipes for signalling/messaging.

#### 13.36.3.2 m\_tooltips

```
Gtk::Tooltips* seq64::mainwnd::m_tooltips [private]
```

#### 13.36.3.3 m\_menubar

```
Gtk::MenuBar* seq64::mainwnd::m_menubar [private]
```

The whole menu bar.

#### 13.36.3.4 m\_menu\_file

```
Gtk::Menu* seq64::mainwnd::m_menu_file [private]
```

### 13.36.3.5 m\_menu\_edit

```
Gtk::Menu* seq64::mainwnd::m_menu_edit [private]
```

### 13.36.3.6 m\_menu\_view

```
Gtk::Menu* seq64::mainwnd::m_menu_view [private]
```

### 13.36.3.7 m\_menu\_help

```
Gtk::Menu* seq64::mainwnd::m_menu_help [private]
```

### 13.36.3.8 m\_ppqn

```
int seq64::mainwnd::m_ppqn [private]
```

We need it early here to be able to pass it along to child objects.

### 13.36.3.9 m\_main\_wid

```
mainwid* seq64::mainwnd::m_main_wid [private]
```

The first is the Patterns Panel, which the mainwid helps implement. 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.

### 13.36.3.10 m\_main\_time

```
maintime* seq64::mainwnd::m_main_time [private]
```

### 13.36.3.11 m\_perf\_edit

```
perfedit* seq64::mainwnd::m_perf_edit [private]
```

#### 13.36.3.12 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.

#### 13.36.3.13 m\_options

```
options* seq64::mainwnd::m_options [private]
```

#### 13.36.3.14 m\_main\_cursor

```
Gdk::Cursor seq64::mainwnd::m_main_cursor [private]
```

#### 13.36.3.15 m\_image\_play

```
Gtk::Image* seq64::mainwnd::m_image_play [private]
```

#### 13.36.3.16 m\_button\_learn

```
Gtk::Button* seq64::mainwnd::m_button_learn [private]
```

#### 13.36.3.17 m\_button\_stop

```
Gtk::Button* seq64::mainwnd::m_button_stop [private]
```

#### 13.36.3.18 m\_button\_play

```
Gtk::Button* seq64::mainwnd::m_button_play [private]
```

If configured to support pause, it also supports the pause pixmap and functionality.



#### 13.36.3.19 m\_button\_perfedit

Gtk::Button\* seq64::mainwnd::m\_button\_perfedit [private]

#### 13.36.3.20 m\_button\_jack

Gtk::Button\* seq64::mainwnd::m\_button\_jack [private]

#### 13.36.3.21 m\_adjust\_bpm

Gtk::Adjustment\* seq64::mainwnd::m\_adjust\_bpm [private]

BPM adjustment object.

#### 13.36.3.22 m\_spinbutton\_bpm

Gtk::SpinButton\* seq64::mainwnd::m\_spinbutton\_bpm [private]

#### 13.36.3.23 m\_adjust\_ss

Gtk::Adjustment\* seq64::mainwnd::m\_adjust\_ss [private]

Screenset adjustment.

#### 13.36.3.24 m\_spinbutton\_ss

Gtk::SpinButton\* seq64::mainwnd::m\_spinbutton\_ss [private]

#### 13.36.3.25 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.

**13.36.3.26 m\_spinbutton\_load\_offset**

```
Gtk::SpinButton* seq64::mainwnd::m_spinbutton_load_offset [private]
```

**13.36.3.27 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.

**13.36.3.28 m\_is\_running**

```
bool seq64::mainwnd::m_is_running [private]
```

**13.36.3.29 m\_timeout\_connect**

```
sigc::connection seq64::mainwnd::m_timeout_connect [private]
```

**13.36.3.30 m\_menu\_mode**

```
bool seq64::mainwnd::m_menu_mode [private]
```

This is a "stazed" feature that might be generally useful.

**13.36.3.31 m\_call\_seq\_edit**

```
bool seq64::mainwnd::m_call_seq_edit [private]
```

Currently, the hard-wired key for this function is the equals key.

**13.36.3.32 m\_call\_seq\_eventedit**

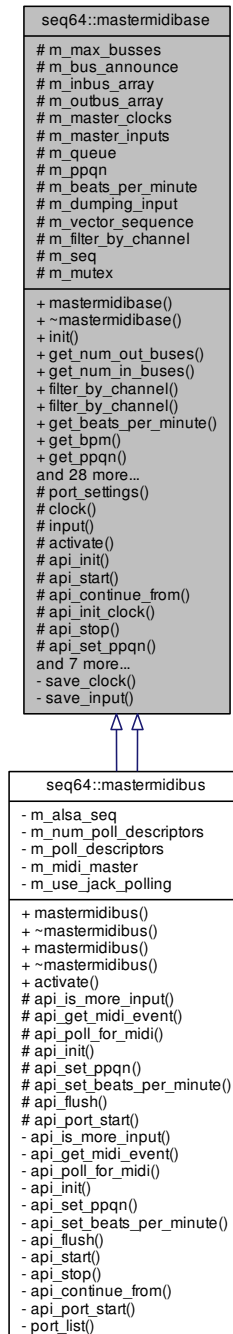
```
bool seq64::mainwnd::m_call_seq_eventedit [private]
```

Currently, the hard-wired key for this function is the minus key.

## 13.37 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.*
  - 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*
  - 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 `set_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 ()`  
*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 `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.*

- bool [initialize\\_buses](#) ()  
*Initializes all for the busses in the input and output buss arrays.*
- 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 [port\\_settings](#) (const std::vector< clock\_e > &clocks, const std::vector< bool > &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_is_more_input` ()=0
- virtual bool `api_get_midi_event` (event \*inev)=0
- virtual int `api_poll_for_midi` ()=0

## 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)
- 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`

## 13.37.1 Constructor & Destructor Documentation

### 13.37.1.1 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 <a href="#">mastermidibase::set_ppqn()</a> to set up the proper PPQN value.
<i>bpm</i>	Provides the beats per minute value, which defaults to c_beats_per_minute.

### 13.37.1.2 ~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]

## 13.37.2 Member Function Documentation

### 13.37.2.1 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.

#### 13.37.2.2 `get_num_out_buses()`

```
int seq64::mastermidibase::get_num_out_buses ( ) const [inline]
```

#### 13.37.2.3 `get_num_in_buses()`

```
int seq64::mastermidibase::get_num_in_buses ( ) const [inline]
```

#### 13.37.2.4 `filter_by_channel()` [1/2]

```
bool seq64::mastermidibase::filter_by_channel ( ) const [inline]
```

#### 13.37.2.5 `filter_by_channel()` [2/2]

```
void seq64::mastermidibase::filter_by_channel (
    bool flag ) [inline]
```

#### 13.37.2.6 `get_beats_per_minute()`

```
midibpm seq64::mastermidibase::get_beats_per_minute ( ) const [inline]
```

#### 13.37.2.7 `get_bpm()`

```
midibpm seq64::mastermidibase::get_bpm ( ) const [inline]
```

#### 13.37.2.8 `get_ppqn()`

```
int seq64::mastermidibase::get_ppqn ( ) const [inline]
```



**13.37.2.9 is\_dumping()**

```
bool seq64::mastermidibase::is_dumping ( ) const [inline]
```

**13.37.2.10 get\_sequence()**

```
sequence* seq64::mastermidibase::get_sequence ( ) const [inline]
```

**13.37.2.11 start()**

```
void seq64::mastermidibase::start ( )
```

Calls the implementation-specific API function for starting.

*Threadsafe*

**13.37.2.12 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*

**13.37.2.13 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.

**13.37.2.14 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.

**13.37.2.15 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.

**13.37.2.16 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.
-------------	---

**13.37.2.17** `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.
-------------	--

**13.37.2.18** `set_clock()` [1/2]

```
void seq64::mastermidibase::set_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.
-------------	---

**13.37.2.19** `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**

<i>ev</i>	Provides the event pointer to be set.
-----------	---------------------------------------

**13.37.2.20 print()**

```
void seq64::mastermidibase::print ( )
```

**13.37.2.21 flush()**

```
void seq64::mastermidibase::flush ( )
```

For example, ALSA provides a function to "drain" the output.

*Threadsafe***13.37.2.22 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.
<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.

**13.37.2.23 dump\_midi\_input()**

```
void seq64::mastermidibase::dump_midi_input (
    event ev )
```

**Parameters**

<i>ev</i>	The event that was recorded, passed as a copy.
-----------	--

#### 13.37.2.24 initialize\_buses()

```
bool seq64::mastermidibase::initialize_buses ( )
```

##### Returns

Returns true if both busses were successfully initialized.

#### 13.37.2.25 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.

#### 13.37.2.26 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.

**13.37.2.27 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.

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.

**13.37.2.28 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.

**13.37.2.29 get\_midi\_event()**

```
bool seq64::mastermidibase::get_midi_event (
    event * ev )
```

**Parameters**

<b>ev</b>	The event to be set based on the found input event.
-----------	---

**13.37.2.30 set\_clock()** [2/2]

```
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.

**13.37.2.31 set\_input()**

```
bool seq64::mastermidibase::set_input (
    bussbyte bus,
    bool inputing )
```

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>inputing</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.

**13.37.2.32 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.

**13.37.2.33 `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.

**13.37.2.34 `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_off` is returned.

**13.37.2.35 `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.
-------------	---------------------------

**13.37.2.36 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.
------------	---

**13.37.2.37 port\_settings()**

```
void seq64::mastermidibase::port_settings (
    const std::vector< clock_e > & clocks,
    const std::vector< bool > & inputs ) [inline], [protected]
```

**13.37.2.38 clock()**

```
clock_e seq64::mastermidibase::clock (
    int bus ) [inline], [protected]
```

**13.37.2.39 input()**

```
bool seq64::mastermidibase::input (
    int bus ) [inline], [protected]
```

**13.37.2.40 activate()**

```
virtual bool seq64::mastermidibase::activate ( ) [inline], [protected], [virtual]
```

Currently re-implemented only in the rtmidi JACK API.

Reimplemented in [seq64::mastermidibus](#).

#### 13.37.2.41 api\_init()

```
virtual void seq64::mastermidibase::api_init (
    int ppqn,
    midibpm bpm ) [protected], [pure virtual]
```

Implemented in [seq64::mastermidibus](#), and [seq64::mastermidibus](#).

#### 13.37.2.42 api\_start()

```
virtual void seq64::mastermidibase::api_start ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#).

#### 13.37.2.43 api\_continue\_from()

```
virtual void seq64::mastermidibase::api_continue_from (
    midipulse ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#).

#### 13.37.2.44 api\_init\_clock()

```
virtual void seq64::mastermidibase::api_init_clock (
    midipulse ) [inline], [protected], [virtual]
```

#### 13.37.2.45 api\_stop()

```
virtual void seq64::mastermidibase::api_stop ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#).

#### 13.37.2.46 api\_set\_ppqn()

```
virtual void seq64::mastermidibase::api_set_ppqn (
    int ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#), and [seq64::mastermidibus](#).

**13.37.2.47 api\_set\_beats\_per\_minute()**

```
virtual void seq64::mastermidibase::api_set_beats_per_minute (
    midibpm ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#), and [seq64::mastermidibus](#).

**13.37.2.48 api\_flush()**

```
virtual void seq64::mastermidibase::api_flush ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#), and [seq64::mastermidibus](#).

**13.37.2.49 api\_clock()**

```
virtual void seq64::mastermidibase::api_clock ( ) [inline], [protected], [virtual]
```

**13.37.2.50 api\_port\_start()**

```
virtual void seq64::mastermidibase::api_port_start (
    int ,
    int ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::mastermidibus](#).

**13.37.2.51 api\_is\_more\_input()**

```
virtual bool seq64::mastermidibase::api_is_more_input ( ) [protected], [pure virtual]
```

Implemented in [seq64::mastermidibus](#), and [seq64::mastermidibus](#).

**13.37.2.52 api\_get\_midi\_event()**

```
virtual bool seq64::mastermidibase::api_get_midi_event (
    event * inev ) [protected], [pure virtual]
```

Implemented in [seq64::mastermidibus](#), and [seq64::mastermidibus](#).

13.37.2.53 `api_poll_for_midi()`

```
virtual int seq64::mastermidibase::api_poll_for_midi ( ) [protected], [pure virtual]
```

Implemented in [seq64::mastermidibus](#), and [seq64::mastermidibus](#).

13.37.2.54 `save_clock()`

```
bool seq64::mastermidibase::save_clock (
    bussbyte bus,
    clock_e clock ) [private]
```

13.37.2.55 `save_input()`

```
bool seq64::mastermidibase::save_input (
    bussbyte bus,
    bool inputting ) [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>inputting</i>	True if the input bus will be inputting MIDI data.

## Returns

Returns true, always.

## 13.37.3 Friends And Related Function Documentation

13.37.3.1 `perform`

```
friend class perform [friend]
```

### 13.37.3.2 midi\_alsa\_info

```
friend class midi_alsa_info [friend]
```

## 13.37.4 Field Documentation

### 13.37.4.1 m\_max\_busses

```
int seq64::mastermidibase::m_max_busses [protected]
```

Set to c\_max\_busses (SEQ64\_DEFAULT\_BUSS\_MAX = 32) for now.

### 13.37.4.2 m\_bus\_announce

```
midibus* seq64::mastermidibase::m_bus_announce [protected]
```

### 13.37.4.3 m\_inbus\_array

```
busarray seq64::mastermidibase::m_inbus_array [protected]
```

### 13.37.4.4 m\_outbus\_array

```
busarray seq64::mastermidibase::m_outbus_array [protected]
```

### 13.37.4.5 m\_master\_clocks

```
std::vector<clock_e> seq64::mastermidibase::m_master_clocks [protected]
```

### 13.37.4.6 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.

#### 13.37.4.7 m\_queue

```
int seq64::mastermidibase::m_queue [protected]
```

#### 13.37.4.8 m\_ppqn

```
int seq64::mastermidibase::m_ppqn [protected]
```

#### 13.37.4.9 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".

#### 13.37.4.10 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.

#### 13.37.4.11 m\_vector\_sequence

```
std::vector<sequence *> seq64::mastermidibase::m_vector_sequence [protected]
```

We want to make this a run-time, non-legacy option.

#### 13.37.4.12 m\_filter\_by\_channel

```
bool seq64::mastermidibase::m_filter_by_channel [protected]
```

#### 13.37.4.13 m\_seq

```
sequence* seq64::mastermidibase::m_seq [protected]
```

#### 13.37.4.14 m\_mutex

```
mutex seq64::mastermidibase::m_mutex [protected]
```

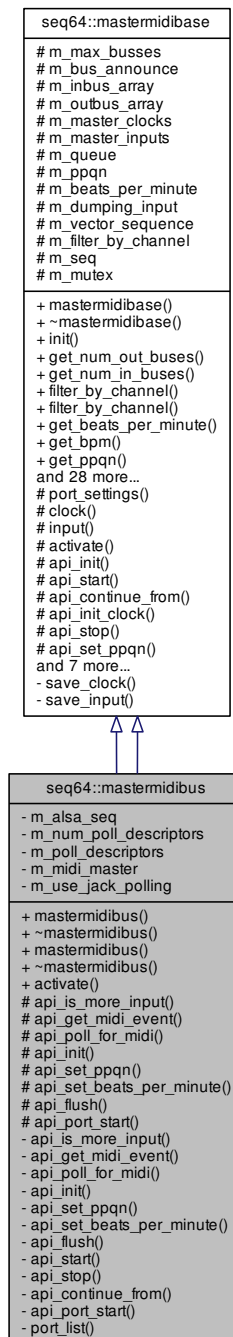
This object is passed to an automutex object that lends exception-safety to the mutex locking.

## 13.38 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)  
The base-class constructor fills the array for our busses.
- virtual [~mastermidibus](#) ()  
The destructor deletes all of the output busses, and terminates the Windows MIDI manager.
- [mastermidibus](#) (int ppqn=SEQ64\_USE\_DEFAULT\_PPQN, [midibpm](#) bpm=SEQ64\_DEFAULT\_BPM)

- virtual `~mastermidibus ()`
- virtual bool `activate ()`

*Activates the mastermidibase code and the `rtmidi_info` object via its `api_connect()` function.*

## Protected Member Functions

- virtual bool `api_is_more_input ()`
- 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)`
- virtual void `api_flush ()`
- virtual void `api_port_start (mastermidibus &masterbus, int bus, int port)`

*Provides MIDI API-specific functionality for the `set_beats_per_minute()` function.*

## Private Member Functions

- virtual bool `api_is_more_input ()`
- 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 ppqn)`
- virtual void `api_set_beats_per_minute (midibpm bpm)`
- virtual void `api_flush ()`
- virtual void `api_start ()`
- virtual void `api_stop ()`
- virtual void `api_continue_from (midipulse tick)`
- virtual void `api_port_start (int client, int port)`
- void `port_list (const std::string &tag)`

*Test the ALSA sequencer to see if any more input is pending.*

*Grab a MIDI event.*

*Initiate a poll() on the existing poll descriptors.*

*Initializes the RtMidi implementation.*

*Shows a list of discovered ports in debug mode.*

## Private Attributes

- `snd_seq_t * m_alsa_seq`
- int `m_num_poll_descriptors`
- struct pollfd \* `m_poll_descriptors`
- `rtmidi_info m_midi_master`
- bool `m_use_jack_polling`

*The ALSA sequencer client handle.*

*The number of descriptors for polling.*

*Points to the list of descriptors for polling.*

*Holds the basic MIDI input and output information for later re-use in the construction of midibus objects.*

*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

### 13.38.1 Detailed Description

This implementation uses the PortMidi library, which supports Linux and Windows, but not JACK or Mac OSX.

### 13.38.2 Constructor & Destructor Documentation

#### 13.38.2.1 mastermidibus() [1/2]

```
seq64::mastermidibus::mastermidibus (
    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 value, SEQ64_USE_DEFAULT_PPQN should be specified.
<i>bpm</i>	Provides the beats per minute value, which defaults to c_beats_per_minute.

#### 13.38.2.2 ~mastermidibus() [1/2]

```
seq64::mastermidibus::~~mastermidibus ( ) [virtual]
```

#### 13.38.2.3 mastermidibus() [2/2]

```
seq64::mastermidibus::mastermidibus (
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

#### 13.38.2.4 ~mastermidibus() [2/2]

```
virtual seq64::mastermidibus::~~mastermidibus ( ) [virtual]
```

### 13.38.3 Member Function Documentation

#### 13.38.3.1 `api_is_more_input()` [1/2]

```
bool seq64::mastermidibus::api_is_more_input ( ) [private], [virtual]
```

Similar to [api\\_poll\\_for\\_midi\(\)](#), except it is threadsafe. We got some cleanup to do!

*Threadsafe*

Implements [seq64::mastermidibase](#).

#### 13.38.3.2 `api_get_midi_event()` [1/2]

```
bool seq64::mastermidibus::api_get_midi_event (
    event * inev ) [private], [virtual]
```

For the ALSA implementation, this call

*Threadsafe*

Implements [seq64::mastermidibase](#).

#### 13.38.3.3 `api_poll_for_midi()` [1/2]

```
int seq64::mastermidibus::api_poll_for_midi ( ) [private], [virtual]
```

This is a primitive poll, which exits when some data is obtained.

```
m_midi_master.api_poll_for_midi(); // NON-FUNCTIONAL!
```

Implements [seq64::mastermidibase](#).

13.38.3.4 `api_init()` [1/2]

```
void seq64::mastermidibus::api_init (
    int ppqn,
    midibpm bpm ) [private], [virtual]
```

Two different styles are supported. If the `--manual-alsa-ports` option is in force, then 16 virtual output ports and one virtual input port are created. They are given names that make it clear which application ([seq64](#)) has set them up. They are not connected to anything. The user will have to use a connection GUI (such as `qjackctl`) or a session manager to make the connections.

Otherwise, the system MIDI input and output ports are scanned (via the [rtmidi\\_info](#) member) and passed to the midibus constructor calls. For every MIDI input port found on the system, this function creates a corresponding output port, and connects to the system MIDI input. For example, for an input port found called "qmidiarp:in 1", we want to create a "shadow" output port called "seq64:qmidiarp in 1".

For every MIDI output found on the system this function creates a corresponding input port, and connects it to the system MIDI output. For For example, for an output port found called "qmidiarp:out 1", we want to create a "shadow" input port called "seq64:qmidiarp out 1".

This code creates a midibus in the conventional manner. Then the [busarray::add\(\)](#) function makes a new businfo object with the desired "output" and "isvirtual" parameters; the businfo object then decides whether to call `init_in()`, `init_out()`, `init_in_sub()`, or `init_out_sub()`.

Are these good conventions, or potentially confusing to users? They match what the legacy `seq24` and `sequencer64` do for ALSA.

## Parameters

<i>ppqn</i>	Provides the (possibly new) value of PPQN to set. ALSA has a function that sets its idea of the PPQN. JACK, as far as we know, does not.
<i>bpm</i>	Provides the (possibly new) value of BPM (beats per minute) to set. ALSA has a function that sets its idea of the BPM. JACK, as far as we know, does not.

Implements [seq64::mastermidibase](#).

### 13.38.3.5 `api_set_ppqn()` [1/2]

```
virtual void seq64::mastermidibus::api_set_ppqn (
    int ppqn ) [private], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

### 13.38.3.6 `api_set_beats_per_minute()` [1/2]

```
virtual void seq64::mastermidibus::api_set_beats_per_minute (
    midibpm bpm ) [private], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

### 13.38.3.7 `api_flush()` [1/2]

```
virtual void seq64::mastermidibus::api_flush ( ) [private], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

### 13.38.3.8 `api_start()`

```
virtual void seq64::mastermidibus::api_start ( ) [private], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

### 13.38.3.9 api\_stop()

```
virtual void seq64::mastermidibus::api_stop ( ) [private], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

### 13.38.3.10 api\_continue\_from()

```
virtual void seq64::mastermidibus::api_continue_from (
    midipulse tick ) [private], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

### 13.38.3.11 api\_port\_start() [1/2]

```
virtual void seq64::mastermidibus::api_port_start (
    int client,
    int port ) [private], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

### 13.38.3.12 activate()

```
bool seq64::mastermidibus::activate ( ) [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

### 13.38.3.13 api\_is\_more\_input() [2/2]

```
virtual bool seq64::mastermidibus::api_is_more_input ( ) [protected], [virtual]
```

Implements [seq64::mastermidibase](#).

### 13.38.3.14 api\_get\_midi\_event() [2/2]

```
virtual bool seq64::mastermidibus::api_get_midi_event (
    event * in ) [protected], [virtual]
```

Implements [seq64::mastermidibase](#).

**13.38.3.15** `api_poll_for_midi()` [2/2]

```
virtual int seq64::mastermidibus::api_poll_for_midi ( ) [protected], [virtual]
```

Implements [seq64::mastermidibase](#).

**13.38.3.16** `api_init()` [2/2]

```
virtual void seq64::mastermidibus::api_init (
    int ppqn,
    midibpm bpm ) [protected], [virtual]
```

Implements [seq64::mastermidibase](#).

**13.38.3.17** `api_set_ppqn()` [2/2]

```
virtual void seq64::mastermidibus::api_set_ppqn (
    int p ) [inline], [protected], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

**13.38.3.18** `api_set_beats_per_minute()` [2/2]

```
virtual void seq64::mastermidibus::api_set_beats_per_minute (
    midibpm b ) [inline], [protected], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

**13.38.3.19** `api_flush()` [2/2]

```
virtual void seq64::mastermidibus::api_flush ( ) [inline], [protected], [virtual]
```

Reimplemented from [seq64::mastermidibase](#).

**13.38.3.20** `api_port_start()` [2/2]

```
virtual void seq64::mastermidibus::api_port_start (
    mastermidibus & masterbus,
    int bus,
    int port ) [inline], [protected], [virtual]
```

**13.38.3.21** `port_list()`

```
void seq64::mastermidibus::port_list (
    const std::string & tag ) [private]
```



## Parameters

<i>tag</i>	Provides a string label indicate the context of this list.
------------	--

## 13.38.4 Friends And Related Function Documentation

### 13.38.4.1 midi\_alsa\_info

```
friend class midi_alsa_info [friend]
```

### 13.38.4.2 midi\_jack\_info

```
friend class midi_jack_info [friend]
```

## 13.38.5 Field Documentation

### 13.38.5.1 m\_alsa\_seq

```
snd_seq_t* seq64::mastermidibus::m_alsa_seq [private]
```

### 13.38.5.2 m\_num\_poll\_descriptors

```
int seq64::mastermidibus::m_num_poll_descriptors [private]
```

### 13.38.5.3 m\_poll\_descriptors

```
struct pollfd* seq64::mastermidibus::m_poll_descriptors [private]
```

#### 13.38.5.4 m\_midi\_master

```
rtmidi_info seq64::mastermidibus::m_midi_master [private]
```

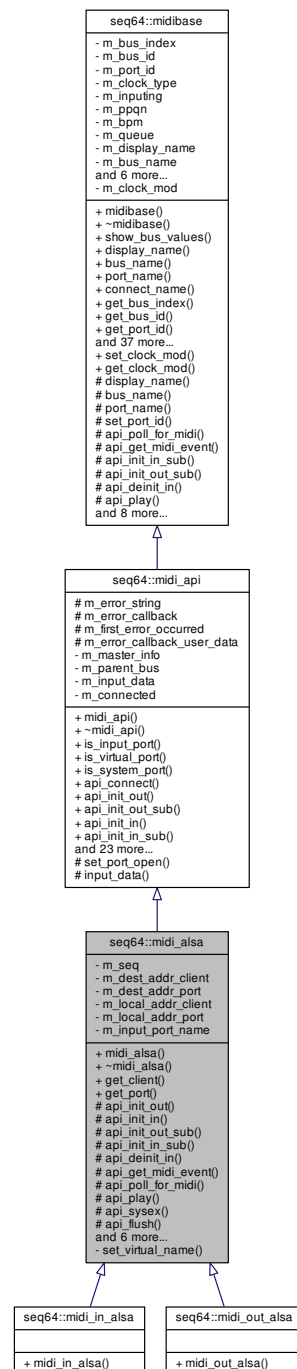
#### 13.38.5.5 m\_use\_jack\_polling

```
bool seq64::mastermidibus::m_use_jack_polling [private]
```

### 13.39 seq64::midi\_alsa Class Reference

This class implements the ALSA version of the [midi\\_api](#).

Inheritance diagram for seq64::midi\_alsa:



## Public Member Functions

- `midin_alsa` (`midibus` & `parentbus`, `midin_info` & `masterinfo`)

*Provides a constructor with client number, port number, ALSA sequencer support, name of client, name of port, etc., mostly contained within an already-initialized `midin_info` object.*

- virtual `~midin_alsa` ()

*A rote empty destructor.*

- 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](#) ()  
*Initialize the MIDI output port.*
- virtual bool [api\\_init\\_in](#) ()  
*Initialize the MIDI input port.*
- virtual bool [api\\_init\\_out\\_sub](#) ()  
*Initialize the output in a different way.*
- virtual bool [api\\_init\\_in\\_sub](#) ()  
*Initialize the output in a different way?*
- virtual bool [api\\_deinit\\_in](#) ()  
*Deinitialize the MIDI input.*
- virtual bool [api\\_get\\_midi\\_event](#) (event \*)  
*ALSA get MIDI events via the [midi\\_alsa\\_info](#) object at present.*
- virtual int [api\\_poll\\_for\\_midi](#) ()  
*This function is supposed to poll for MIDI data, but the current ALSA implementation DOES NOT USE THIS FUNCTION.*
- virtual void [api\\_play](#) (event \*e24, midibyte channel)  
*This [play\(\)](#) function takes a native event, encodes it to an ALSA 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.*
- virtual void [api\\_sysex](#) (event \*e24)  
*Takes a native SYSEX event, encodes it to an ALSA event, and then puts it in the queue.*
- virtual void [api\\_flush](#) ()  
*Flushes our local queue events out into ALSA.*
- virtual void [api\\_continue\\_from](#) (midipulse tick, midipulse beats)
- virtual void [api\\_start](#) ()  
*This function gets the MIDI clock a-runnin', if the clock type is not e\_clock\_off.*
- virtual void [api\\_stop](#) ()  
*Stop the MIDI buss.*
- virtual void [api\\_clock](#) (midipulse tick)  
*Generates the MIDI clock, starting at the given tick value.*
- virtual void [api\\_set\\_ppqn](#) (int ppqn)
- virtual void [api\\_set\\_beats\\_per\\_minute](#) (midibpm bpm)  
*Set the BPM value (beats per minute).*

## Private Member Functions

- bool [set\\_virtual\\_name](#) (int portid, const std::string &portname)  
*Gets information directly from ALSA.*

## 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

### 13.39.1 Constructor & Destructor Documentation

#### 13.39.1.1 midi\_alsa()

```
seq64::midi_alsa::midi_alsa (
    midibus & parentbus,
    midi_info & masterinfo )
```

This constructor is the only one that is used for the MIDI input and output busses, whether the [manual-alsa-ports] option is in force or not. The actual setup of a normal or virtual port is done in the `api_*_init_*`() routines.

Also used for the announce buss, and in the `mastermidi_alsa::port_start()` function. There's currently some overlap between local/dest client and port numbers and the buss and port numbers of the new midibase interface.

Also, note that the optionsfile module uses the master buss to get the buss names when it writes the file.

#### Parameters

<i>parentbus</i>	Provides much of the infor about this ALSA buss.
<i>masterinfo</i>	Provides the information about the desired port, and more.

#### 13.39.1.2 ~midi\_alsa()

```
seq64::midi_alsa::~~midi_alsa ( ) [virtual]
```

### 13.39.2 Member Function Documentation

### 13.39.2.1 `get_client()`

```
virtual int seq64::midi_alsa::get_client ( ) const [inline], [virtual]
```

Can we replace it with `get_client_id()`?

### 13.39.2.2 `get_port()`

```
virtual int seq64::midi_alsa::get_port ( ) const [inline], [virtual]
```

### 13.39.2.3 `api_init_out()`

```
bool seq64::midi_alsa::api_init_out ( ) [protected], [virtual]
```

This initialization is done when the "manual ALSA ports" option is not in force.

This initialization is like the "open\_port()" function of the RtMidi library, with the addition of the `snd_seq_connect_to()` call involving the local and destination ports.

**Tricky Code** One important thing to note is that this output port is initialized with the `SND_SEQ_PORT_CAP_READ` flag, which means this is really an input port. We connect this input port with a system output port that was discovered. This is backwards of the way RtMidi does it.

#### Returns

Returns true unless setting up ALSA MIDI failed in some way.

Implements [seq64::midi\\_api](#).

### 13.39.2.4 `api_init_in()`

```
bool seq64::midi_alsa::api_init_in ( ) [protected], [virtual]
```

Subscription handlers:

In ALSA library, subscription is done via `snd_seq_subscribe_port()` function. It takes the argument of `snd_seq_port_subscribe_t` record pointer. Suppose that you have a client which will receive data from a MIDI input device. The source and destination addresses are like the below:

```
snd_seq_addr_t sender, dest;
sender.client = MIDI_input_client;
sender.port = MIDI_input_port;
dest.client = my_client;
dest.port = my_port;
```

To set these values as the connection call like this.

```
snd_seq_port_subscribe_t *subs;
snd_seq_port_subscribe_alloca(&subs);
snd_seq_port_subscribe_set_sender(subs, &sender);
snd_seq_port_subscribe_set_dest(subs, &dest);
snd_seq_subscribe_port(handle, subs);
```

**Tricky Code** One important thing to note is that this input port is initialized with the `SND_SEQ_PORT_CAP_WRITE` flag, which means this is really an output port. We connect this output port with a system input port that was discovered. This is backwards of the way RtMidi does it.

#### Returns

Returns true unless setting up ALSA MIDI failed in some way.

Implements [seq64::midi\\_api](#).

### 13.39.2.5 api\_init\_out\_sub()

```
bool seq64::midi_alsa::api_init_out_sub ( ) [protected], [virtual]
```

This version of initialization is used by mastermidi\_alsa in the "manual ALSA ports" clause. This code is also very similar to the same function in the [midibus::api\\_init\\_out\\_sub\(\)](#) function of [midibus::api\\_init\\_out\\_sub\(\)](#).

#### Returns

Returns true unless setting up the ALSA port failed in some way.

Implements [seq64::midi\\_api](#).

### 13.39.2.6 api\_init\_in\_sub()

```
bool seq64::midi_alsa::api_init_in_sub ( ) [protected], [virtual]
```

#### Returns

Returns true unless setting up the ALSA port failed in some way.

Implements [seq64::midi\\_api](#).

### 13.39.2.7 api\_deinit\_in()

```
bool seq64::midi_alsa::api_deinit_in ( ) [protected], [virtual]
```

Set the input and the output ports. The destination port is actually our local port.

#### Returns

Returns true, unless an error occurs.

Implements [seq64::midi\\_api](#).

### 13.39.2.8 api\_get\_midi\_event()

```
virtual bool seq64::midi_alsa::api_get_midi_event (
    event * ) [inline], [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

### 13.39.2.9 `api_poll_for_midi()`

```
int seq64::midi_alsa::api_poll_for_midi ( ) [protected], [virtual]
```

TODO? See `seq_alsamidi`'s `mastermidibus::api_poll_for_midi()`. Right now we'd need to forward this call to [midi↔\\_alsa\\_info](#).

```
return poll(m_poll_descriptors, m_num_poll_descriptors, 1000);
```

This kills startup:

```
return master_info().api_poll_for_midi();
```

#### Returns

Always returns 0.

Implements [seq64::midi\\_api](#).

### 13.39.2.10 `api_play()`

```
void seq64::midi_alsa::api_play (
    event * e24,
    midibyte channel ) [protected], [virtual]
```

#### 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.

Implements [seq64::midi\\_api](#).

### 13.39.2.11 `api_sysex()`

```
void seq64::midi_alsa::api_sysex (
    event * e24 ) [protected], [virtual]
```

#### Parameters

<i>e24</i>	The event to be handled.
------------	--------------------------

Implements [seq64::midi\\_api](#).



#### 13.39.2.12 api\_flush()

```
void seq64::midi_alsa::api_flush ( ) [protected], [virtual]
```

This is also a [midi\\_alsa\\_info](#) function.

Implements [seq64::midi\\_api](#).

#### 13.39.2.13 api\_continue\_from()

```
void seq64::midi_alsa::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.39.2.14 api\_start()

```
void seq64::midi_alsa::api_start ( ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.39.2.15 api\_stop()

```
void seq64::midi_alsa::api_stop ( ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.39.2.16 api\_clock()

```
void seq64::midi_alsa::api_clock (
    midipulse tick ) [protected], [virtual]
```

*Threadsafe*

Parameters

<i>tick</i>	Provides the starting tick, unused in the ALSA implementation.
-------------	--

Implements [seq64::midi\\_api](#).

### 13.39.2.17 api\_set\_ppqn()

```
void seq64::midi_alsa::api_set_ppqn (
    int ppqn ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

### 13.39.2.18 api\_set\_beats\_per\_minute()

```
void seq64::midi_alsa::api_set_beats_per_minute (
    midibpm bpm ) [protected], [virtual]
```

This is done by creating an ALSA tempo structure, adding tempo information to it, and then setting the ALSA sequencer object with this information.

We fill the ALSA tempo structure (`snd_seq_queue_tempo_t`) with the current tempo information, set the BPM value, put it in the tempo structure, and give the tempo value to the ALSA queue.

#### Note

Consider using `snd_seq_change_queue_tempo()` here if the ALSA queue has already been started. It's arguments would be `m_alsa_seq`, `m_queue`, tempo (microseconds), and null.

#### Threadsafe

#### Parameters

<i>bpm</i>	Provides the beats-per-minute value to set.
------------	---

Implements [seq64::midi\\_api](#).

### 13.39.2.19 set\_virtual\_name()

```
bool seq64::midi_alsa::set_virtual_name (
    int portid,
    const std::string & portname ) [private]
```

The problem this function solves is that the midibus constructor for a virtual ALSA port doesn't not have all of the information it needs at that point. Here, we can get this information and get the actual data we need to rename the port to something accurate. Same as the `seq_alsamidi` version of this function.

**Returns**

Returns true if all of the information could be obtained. If false is returned, then the caller should not use the side-effects.

**Side-effect(s)** Passes back the values found.

### 13.39.3 Field Documentation

#### 13.39.3.1 m\_seq

```
snd_seq_t* const seq64::midi_alsa::m_seq [private]
```

#### 13.39.3.2 m\_dest\_addr\_client

```
const int seq64::midi_alsa::m_dest_addr_client [private]
```

Could potentially be replaced by [midibase::m\\_bus\\_id](#).

#### 13.39.3.3 m\_dest\_addr\_port

```
const int seq64::midi_alsa::m_dest_addr_port [private]
```

Could potentially be replaced by [midibase::m\\_port\\_id](#).

#### 13.39.3.4 m\_local\_addr\_client

```
const int seq64::midi_alsa::m_local_addr_client [private]
```

#### 13.39.3.5 m\_local\_addr\_port

```
int seq64::midi_alsa::m_local_addr_port [private]
```

#### 13.39.3.6 m\_input\_port\_name

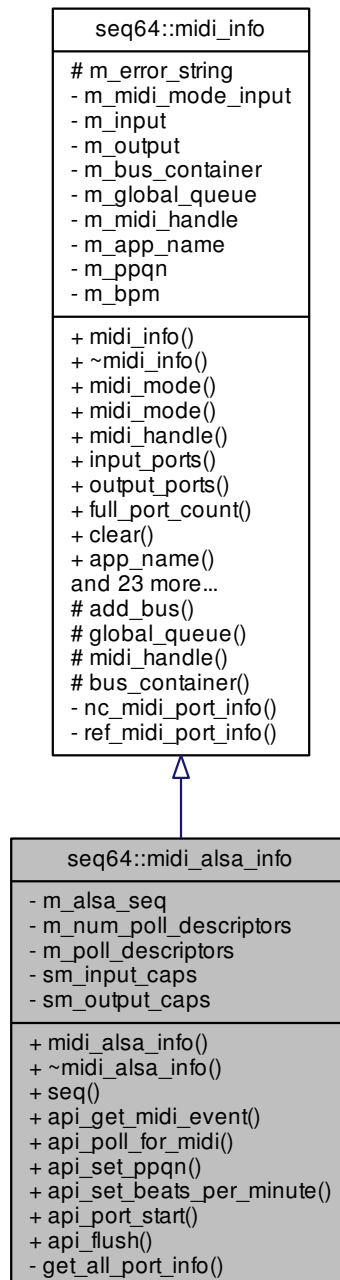
```
const std::string seq64::midi_alsa::m_input_port_name [private]
```

It is derived from the (optionally configured) official client name for the application with the word "in" appended.

## 13.40 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)

- *Principal constructor.*
- virtual [~midi\\_alsa\\_info](#) ()
- *Destructor.*
- `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)`
- *Grab a MIDI event.*
- virtual int `api\_poll\_for\_midi ()`
- *Polls for any ALSA MIDI information using a timeout value of 1000 milliseconds.*
- virtual void `api\_set\_ppqn (int p)`
- *Sets the PPQN numeric value, then makes ALSA calls to set up the PPQ tempo.*
- virtual void `api\_set\_beats\_per\_minute (midibpm b)`
- *Sets the BPM numeric value, then makes ALSA calls to set up the BPM tempo.*
- virtual void `api\_port\_start (mastermidibus &masterbus, int bus, int port)`
- *Start the given ALSA MIDI port.*
- virtual void `api\_flush ()`
- *Flushes our local queue events out into ALSA.*

### Private Member Functions

- virtual int `get\_all\_port\_info ()`
- *Gets information on ALL ports, putting input data into one [midi\\_info](#) container, and putting output data into another container.*

### 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

#### 13.40.1 Constructor & Destructor Documentation

##### 13.40.1.1 midi\_alsa\_info()

```
seq64::midi_alsa_info::midi_alsa_info (
    const std::string & appname,
    int ppqn = SEQ64_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

## Parameters

<i>appname</i>	Provides the name of the application.
<i>ppqn</i>	Provides the PPQN value needed by this object.
<i>bpm</i>	Provides the beats/minute value needed by this object.

13.40.1.2 `~midi_alsa_info()`

```
seq64::midi_alsa_info::~~midi_alsa_info ( ) [virtual]
```

Closes a connection if it exists, shuts down the input thread, and then cleans up any API resources in use.

## 13.40.2 Member Function Documentation

13.40.2.1 `seq()`

```
snd_seq_t* seq64::midi_alsa_info::seq ( ) [inline]
```

13.40.2.2 `api_get_midi_event()`

```
bool seq64::midi_alsa_info::api_get_midi_event (
    event * inev ) [virtual]
```

First, a rather large buffer is allocated on the stack to hold the MIDI event data. Next, if the `--alsa-manual-ports` option is not in force, then we check to see if the event is a port-start, port-exit, or port-change event, and we process it, and are done.

Otherwise, we create a "MIDI event parser" and decode the MIDI event.

## Parameters

<i>inev</i>	The event to be set based on the found input event.
-------------	---

## Returns

This function returns false if we are not using virtual/manual ports and the event is an ALSA port-start, port-exit, or port-change event. It also returns false if there is no event to decode. Otherwise, it returns true.

We will only get `EVENT_SYSEX` on the first packet of MIDI data; the rest we have to poll for. SysEx processing is currently disabled.

Implements [seq64::midi\\_info](#).

#### 13.40.2.3 api\_poll\_for\_midi()

```
int seq64::midi_alsa_info::api_poll_for_midi ( ) [virtual]
```

##### Returns

Returns the result of the call to poll() on the global ALSA poll descriptors.

Implements [seq64::midi\\_info](#).

#### 13.40.2.4 api\_set\_ppqn()

```
void seq64::midi_alsa_info::api_set_ppqn (
    int p ) [virtual]
```

##### Parameters

<i>p</i>	The desired new PPQN value to set.
----------	------------------------------------

Reimplemented from [seq64::midi\\_info](#).

#### 13.40.2.5 api\_set\_beats\_per\_minute()

```
void seq64::midi_alsa_info::api_set_beats_per_minute (
    midibpm b ) [virtual]
```

##### Parameters

<i>b</i>	The desired new BPM value to set.
----------	-----------------------------------

Reimplemented from [seq64::midi\\_info](#).

#### 13.40.2.6 api\_port\_start()

```
void seq64::midi_alsa_info::api_port_start (
    mastermidibus & masterbus,
```

```
int bus,
int port ) [virtual]
```

This function is called by [api\\_get\\_midi\\_event\(\)](#) when an ALSA event SND\_SEQ\_EVENT\_PORT\_START is received.

- Get the API's client and port information.
- Do some capability checks.
- Find the client/port combination among the set of input/output busses. If it exists and is not active, then mark it as a replacement. If it is not a replacement, it will increment the number of input/output busses.

We can simplify this code a bit by using elements already present in [midi\\_alsa\\_info](#).

#### Parameters

<i>masterbus</i>	Provides the object that is need to get access to the busses that need to be started.
<i>bus</i>	Provides the ALSA bus/client number.
<i>port</i>	Provides the ALSA client port.

Reimplemented from [seq64::midi\\_info](#).

#### 13.40.2.7 api\_flush()

```
void seq64::midi_alsa_info::api_flush ( ) [virtual]
```

This is also a [midi\\_alsa](#) function.

Implements [seq64::midi\\_info](#).

#### 13.40.2.8 get\_all\_port\_info()

```
int seq64::midi_alsa_info::get_all_port_info ( ) [private], [virtual]
```

For ALSA input, the first item added is the ALSA MIDI system "announce" buss. It has the client:port value of "0:1", denoted by the ALSA macros SND\_SEQ\_CLIENT\_SYSTEM:SND\_SEQ\_PORT\_SYSTEM\_ANNOUNCE.

#### Returns

Returns the total number of ports found. For an ALSA setup, finding no ALSA ports can be considered an error. However, finding no ports for other APIS may be fine. So, we set the result to -1 to flag a true error.

Implements [seq64::midi\\_info](#).



### 13.40.3 Field Documentation

#### 13.40.3.1 sm\_input\_caps

```
unsigned seq64::midi_alsa_info::sm_input_caps [static], [private]
```

#### 13.40.3.2 sm\_output\_caps

```
unsigned seq64::midi_alsa_info::sm_output_caps [static], [private]
```

#### 13.40.3.3 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.

#### 13.40.3.4 m\_num\_poll\_descriptors

```
int seq64::midi_alsa_info::m_num_poll_descriptors [private]
```

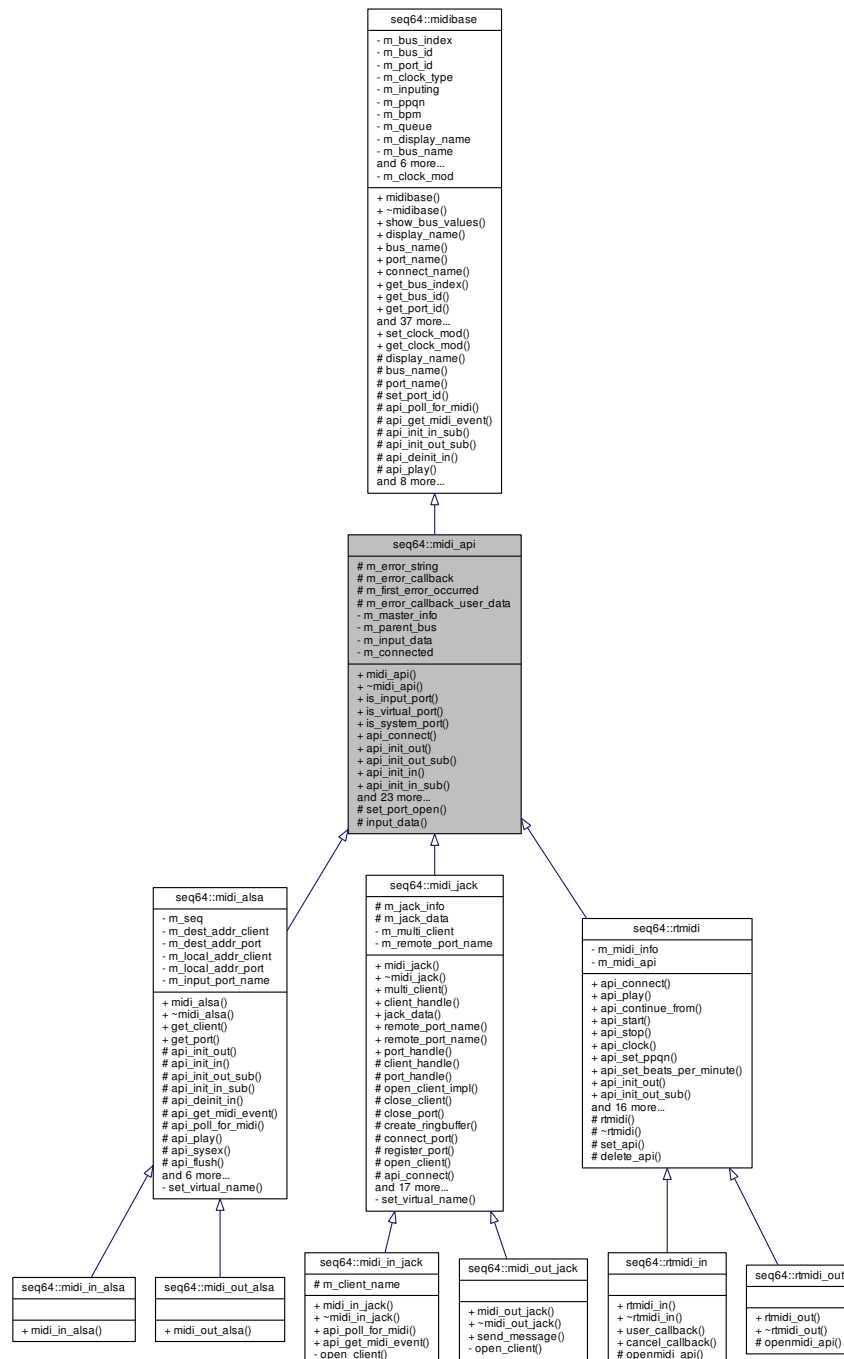
#### 13.40.3.5 m\_poll\_descriptors

```
struct pollfd* seq64::midi_alsa_info::m_poll_descriptors [private]
```

## 13.41 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)

*Principle constructor.*

- virtual `~midi_api ()`

*Destructor, needed because it is virtual.*

- 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 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 int `api_poll_for_midi ()`=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)`

*'Getter' function for member `m_master_info.midi_mode()` This function makes it a bit simpler on the caller.*

- void `error (rterror::Type type, const std::string &errorstring)`

*Provides an error handler that can support an error callback.*

- void `user_callback (rtmidi_callback_t callback, void *userdata)`

*Wires in a MIDI input callback function.*

- void `cancel_callback ()`

*Removes the MIDI input callback and some items related to it.*

## 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

### 13.41.1 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.

### 13.41.2 Constructor & Destructor Documentation

#### 13.41.2.1 `midi_api()`

```
seq64::midi_api::midi_api (
    midibus & parentbus,
    midi_info & masterinfo )
```

#### 13.41.2.2 `~midi_api()`

```
seq64::midi_api::~~midi_api ( ) [virtual]
```

### 13.41.3 Member Function Documentation

#### 13.41.3.1 is\_input\_port()

```
bool seq64::midi_api::is_input_port ( ) const
```

#### 13.41.3.2 is\_virtual\_port()

```
bool seq64::midi_api::is_virtual_port ( ) const
```

#### 13.41.3.3 is\_system\_port()

```
bool seq64::midi_api::is_system_port ( ) const
```

#### 13.41.3.4 api\_connect()

```
virtual bool seq64::midi_api::api_connect ( ) [inline], [virtual]
```

Reimplemented in [seq64::midi\\_jack](#), and [seq64::rtmidi](#).

#### 13.41.3.5 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](#).

#### 13.41.3.6 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](#).

#### 13.41.3.7 `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](#).

#### 13.41.3.8 `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](#).

#### 13.41.3.9 `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](#).

#### 13.41.3.10 `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](#).

#### 13.41.3.11 `api_poll_for_midi()`

```
virtual int seq64::midi_api::api_poll_for_midi ( ) [pure virtual]
```

Reimplemented from [seq64::midibase](#).

Implemented in [seq64::midi\\_in\\_jack](#), [seq64::midi\\_jack](#), [seq64::midi\\_alsa](#), and [seq64::rtmidi](#).

#### 13.41.3.12 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](#).

#### 13.41.3.13 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](#).

#### 13.41.3.14 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](#).

#### 13.41.3.15 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](#).

#### 13.41.3.16 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](#).

#### 13.41.3.17 `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](#).

#### 13.41.3.18 `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](#).

#### 13.41.3.19 `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](#).

#### 13.41.3.20 `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](#).

#### 13.41.3.21 `api_get_bus_name()`

```
virtual std::string seq64::midi_api::api_get_bus_name ( ) [inline], [virtual]
```

#### 13.41.3.22 `api_get_port_name()`

```
virtual std::string seq64::midi_api::api_get_port_name ( ) [inline], [virtual]
```

Reimplemented in [seq64::midi\\_jack](#).



**13.41.3.23 is\_port\_open()**

```
bool seq64::midi_api::is_port_open ( ) const [inline]
```

**13.41.3.24 master\_info()** [1/2]

```
midi_info& seq64::midi_api::master_info ( ) [inline]
```

**13.41.3.25 master\_info()** [2/2]

```
const midi_info& seq64::midi_api::master_info ( ) const [inline]
```

**13.41.3.26 parent\_bus()** [1/2]

```
midibus& seq64::midi_api::parent_bus ( ) [inline]
```

**13.41.3.27 parent\_bus()** [2/2]

```
const midibus& seq64::midi_api::parent_bus ( ) const [inline]
```

**13.41.3.28 master\_midi\_mode()**

```
void seq64::midi_api::master_midi_mode (
    bool input )
```

**13.41.3.29 error()**

```
void seq64::midi_api::error (
    rterror::Type type,
    const std::string & errorstring )
```

**Exceptions**

<i>If</i>	the error is not just a warning, then an rterror object is thrown.
-----------	--

## Parameters

<i>type</i>	The type of the error.
<i>errorstring</i>	The error message, which gets copied if this is the first error.

13.41.3.30 `user_callback()`

```
void seq64::midi_api::user_callback (
    rtmidi_callback_t callback,
    void * userdata )
```

We moved it into the base class, trading convenience for the chance of confusion.

## Parameters

<i>callback</i>	Provides the callback function.
<i>userdata</i>	Provides the user data needed by the callback function.

13.41.3.31 `cancel_callback()`

```
void seq64::midi_api::cancel_callback ( )
```

We moved it into the base class, trading convenience for the chance of confusion.

13.41.3.32 `set_port_open()`

```
void seq64::midi_api::set_port_open ( ) [inline], [protected]
```

13.41.3.33 `input_data()`

```
rtmidi_in_data* seq64::midi_api::input_data ( ) [inline], [protected]
```

## 13.41.4 Field Documentation

13.41.4.1 `m_master_info`

```
midi_info& seq64::midi_api::m_master_info [private]
```

#### 13.41.4.2 m\_parent\_bus

```
midibus& 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 [midi\\_info](#) object.

#### 13.41.4.3 m\_input\_data

```
rtmidi_in_data seq64::midi_api::m_input_data [private]
```

#### 13.41.4.4 m\_connected

```
bool seq64::midi_api::m_connected [private]
```

#### 13.41.4.5 m\_error\_string

```
std::string seq64::midi_api::m_error_string [protected]
```

This is an original RtMidi concept.

#### 13.41.4.6 m\_error\_callback

```
rterror_callback seq64::midi_api::m_error_callback [protected]
```

This is an original RtMidi concept.

#### 13.41.4.7 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.

#### 13.41.4.8 m\_error\_callback\_user\_data

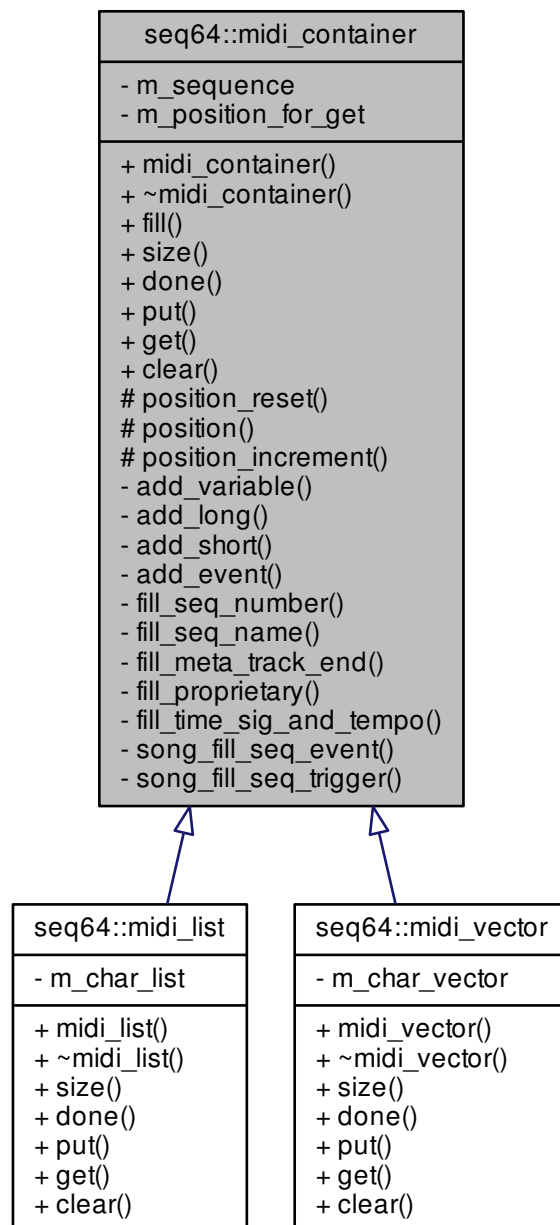
```
void* seq64::midi_api::m_error_callback_user_data [protected]
```

This is an original RtMidi concept. I have to confess I am not sure how it is/should be used, yet.

## 13.42 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)`  
*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 int `position_reset () const`  
*'Setter' function for member `m_position_for_get` Sets the position to 0 and then returns that value.*
- unsigned int `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 `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.*
- void `fill_time_sig_and_tempo (const perform &p)`  
*Fill in the time-signature and tempo information.*
- `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 int [m\\_position\\_for\\_get](#)  
Provides the position in the container when making a series of [get\(\)](#) calls on the container.

## Friends

- class [midifile](#)

### 13.42.1 Detailed Description

It is the base class for [midi\\_list](#) and [midi\\_vector](#).

### 13.42.2 Constructor & Destructor Documentation

#### 13.42.2.1 midi\_container()

```
seq64::midi_container::midi_container (
    sequence & seq )
```

#### Parameters

<a href="#">seq</a>	Provides a reference to the sequence/track for which this container holds MIDI data.
---------------------	--

#### 13.42.2.2 ~midi\_container()

```
virtual seq64::midi_container::~~midi_container ( ) [inline], [virtual]
```

### 13.42.3 Member Function Documentation

#### 13.42.3.1 fill()

```
void seq64::midi_container::fill (
    int tracknumber,
    const perform & p )
```

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.

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`.

*Not threadsafe* The sequence object bound to this container needs to provide the locking mechanism when calling this function.

Parameters

<i>tracknumber</i>	Provides the track number. 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.

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.

### 13.42.3.2 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](#).

### 13.42.3.3 done()

```
virtual bool seq64::midi_container::done ( ) const [inline], [virtual]
```

Reimplemented in [seq64::midi\\_vector](#), and [seq64::midi\\_list](#).

#### 13.42.3.4 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](#).

#### 13.42.3.5 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](#).

#### 13.42.3.6 clear()

```
virtual void seq64::midi_container::clear ( ) [pure virtual]
```

Implemented in [seq64::midi\\_vector](#), and [seq64::midi\\_list](#).

#### 13.42.3.7 position\_reset()

```
unsigned int seq64::midi_container::position_reset ( ) const [inline], [protected]
```

#### 13.42.3.8 position()

```
unsigned int seq64::midi_container::position ( ) const [inline], [protected]
```

#### 13.42.3.9 position\_increment()

```
void seq64::midi_container::position_increment ( ) const [inline], [protected]
```

#### 13.42.3.10 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.
---	--

## 13.42.3.11 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.
---	--

## 13.42.3.12 add\_short()

```
void seq64::midi_container::add_short (
    midishort x ) [private]
```

## Parameters

x	Provides the timestamp (pulse value) to be added to the container.
---	--

## 13.42.3.13 add\_event()

```
void seq64::midi_container::add_event (
    const event & e,
    midipulse deltetime ) [private]
```

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.

## 13.42.3.14 fill\_seq\_number()

```
void seq64::midi_container::fill_seq_number (
    int seq ) [private]
```

Writes `0xFF 0x00 0x02`, and then the 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()`.

## Parameters

<i>seq</i>	The sequence/track number to write.
------------	-------------------------------------

13.42.3.15 `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.
-------------	--

13.42.3.16 `fill_meta_track_end()`

```
void seq64::midi_container::fill_meta_track_end (
    midipulse deltetime ) [private]
```

13.42.3.17 `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.

### 13.42.3.18 fill\_time\_sig\_and\_tempo()

```
void seq64::midi_container::fill_time_sig_and_tempo (
    const perform & p ) [private]
```

This function is used only for the first track, The sizes of these meta events are defined as SEQ64\_TIME\_TEMPO\_SIZE. However, we do not have to add that value in, as it is already counted in the intrinsic size of the container.

We now make sure that the proper values are part of the perform object for usage in this particular track. For export, we cannot guarantee that the first (0th) track/sequence is exportable.

## Parameters

<i>p</i>	Provides the performance object from which we get some global MIDI parameters.
----------	--

## 13.42.3.19 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.

## 13.42.3.20 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.

## 13.42.4 Friends And Related Function Documentation

## 13.42.4.1 midifile

```
friend class midifile [friend]
```

### 13.42.5 Field Documentation

#### 13.42.5.1 m\_sequence

`sequence& seq64::midi_container::m_sequence [private]`

#### 13.42.5.2 m\_position\_for\_get

`unsigned int seq64::midi_container::m_position_for_get [mutable], [private]`

## 13.43 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.*

### 13.43.1 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`.

### 13.43.2 Member Enumeration Documentation

#### 13.43.2.1 action

```
enum seq64::midi_control::action
```

## Enumerator

action_toggle	Provides the kind of MIDI control event found, used in the new <a href="#">perform::handle_midi_control_ex()</a> 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.

### 13.43.3 Constructor & Destructor Documentation

#### 13.43.3.1 midi\_control()

```
seq64::midi_control::midi_control ( ) [inline]
```

Every member is either false or zero.

### 13.43.4 Member Function Documentation

#### 13.43.4.1 active()

```
bool seq64::midi_control::active ( ) const [inline]
```

#### 13.43.4.2 inverse\_active()

```
bool seq64::midi_control::inverse_active ( ) const [inline]
```

#### 13.43.4.3 status()

```
int seq64::midi_control::status ( ) const [inline]
```

**13.43.4.4 data()**

```
int seq64::midi_control::data ( ) const [inline]
```

**13.43.4.5 min\_value()**

```
int seq64::midi_control::min_value ( ) const [inline]
```

**13.43.4.6 max\_value()**

```
int seq64::midi_control::max_value ( ) const [inline]
```

**13.43.4.7 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.
---------------	--

**13.43.4.8 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.
---------------	--



#### 13.43.4.9 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.

#### 13.43.4.10 in\_range()

```
bool seq64::midi_control::in_range (
    midibyte data ) const [inline]
```

### 13.43.5 Field Documentation

#### 13.43.5.1 m\_active

```
bool seq64::midi_control::m_active [private]
```

#### 13.43.5.2 m\_inverse\_active

```
bool seq64::midi_control::m_inverse_active [private]
```

#### 13.43.5.3 m\_status

```
int seq64::midi_control::m_status [private]
```

#### 13.43.5.4 m\_data

```
int seq64::midi_control::m_data [private]
```

#### 13.43.5.5 m\_min\_value

```
int seq64::midi_control::m_min_value [private]
```

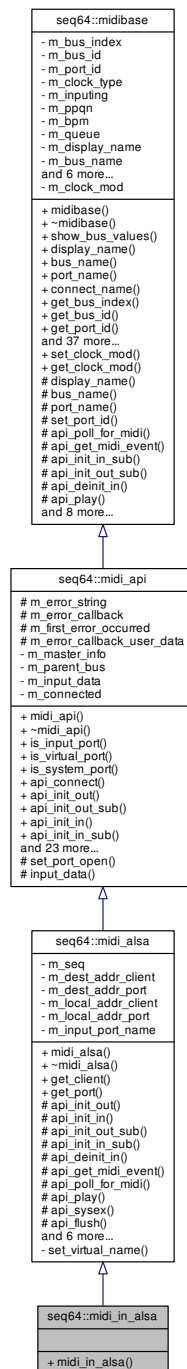
#### 13.43.5.6 m\_max\_value

```
int seq64::midi_control::m_max_value [private]
```

### 13.44 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)

ALSA MIDI input normal port or virtual port constructor.

## Additional Inherited Members

### 13.44.1 Constructor & Destructor Documentation

#### 13.44.1.1 `midi_in_alsa()`

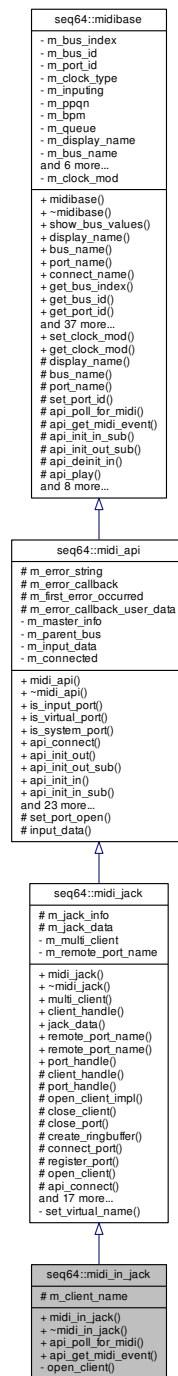
```
seq64::midi_in_alsa::midi_in_alsa (
    midibus & parentbus,
    midi_info & masterinfo )
```

The kind of port is determine by which port-initialization function the mastermidibus calls.

### 13.45 `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)  
*Principal constructor.*
- virtual [~midi\\_in\\_jack](#) ()  
*Destructor.*
- virtual int [api\\_poll\\_for\\_midi](#) ()

Checks the `rtmidi_in_data` queue for the number of items in the queue.

- virtual bool `api_get_midi_event` (`event *`)

Gets a MIDI 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

### 13.45.1 Constructor & Destructor Documentation

#### 13.45.1.1 `midi_in_jack()`

```
seq64::midi_in_jack::midi_in_jack (
    midibus & parentbus,
    midi_info & masterinfo )
```

For Sequencer64, we don't current need to create a `midi_in_jack` object; all that is needed is created via the `api_↔init_in*()` functions. Also, this constructor still needs to do something with queue size.

#### Parameters

<code>parentbus</code>	Provides the buss object that determines buss-specific parameters of this class.
<code>masterinfo</code>	Provides information about the JACK system as found on this machine.

#### 13.45.1.2 `~midi_in_jack()`

```
seq64::midi_in_jack::~~midi_in_jack ( ) [virtual]
```

Currently the base class closes the port, closes the JACK client, and cleans up the API data structure.

### 13.45.2 Member Function Documentation

## 13.45.2.1 api\_poll\_for\_midi()

```
int seq64::midi_in_jack::api_poll_for_midi ( ) [virtual]
```

WE MAY NEED LOCKING.

## Returns

Returns the value of `rtindata->queue().count()`, unless the caller is using an `rtmidi` callback function, in which case 0 is always returned.

Reimplemented from [seq64::midi\\_jack](#).

## 13.45.2.2 api\_get\_midi\_event()

```
bool seq64::midi_in_jack::api_get_midi_event (
    event * inev ) [virtual]
```

This implementation gets a [midi\\_message](#) off the front of the queue and converts it to an `sequence64` event.

## Parameters

<i>inev</i>	Provides the destination for the MIDI event.
-------------	--

## Returns

Returns true if a MIDI event was obtained, indicating that the return parameter can be used.

We will only get `EVENT_SYSEX` on the first packet of MIDI data; the rest we have to poll for. SysEx processing is currently disabled.

Reimplemented from [seq64::midi\\_jack](#).

## 13.45.2.3 open\_client()

```
virtual bool seq64::midi_in_jack::open_client ( ) [inline], [private], [virtual]
```

This function replaces the `RtMidi` function "`connect()`".

Implements [seq64::midi\\_jack](#).

## 13.45.3 Field Documentation

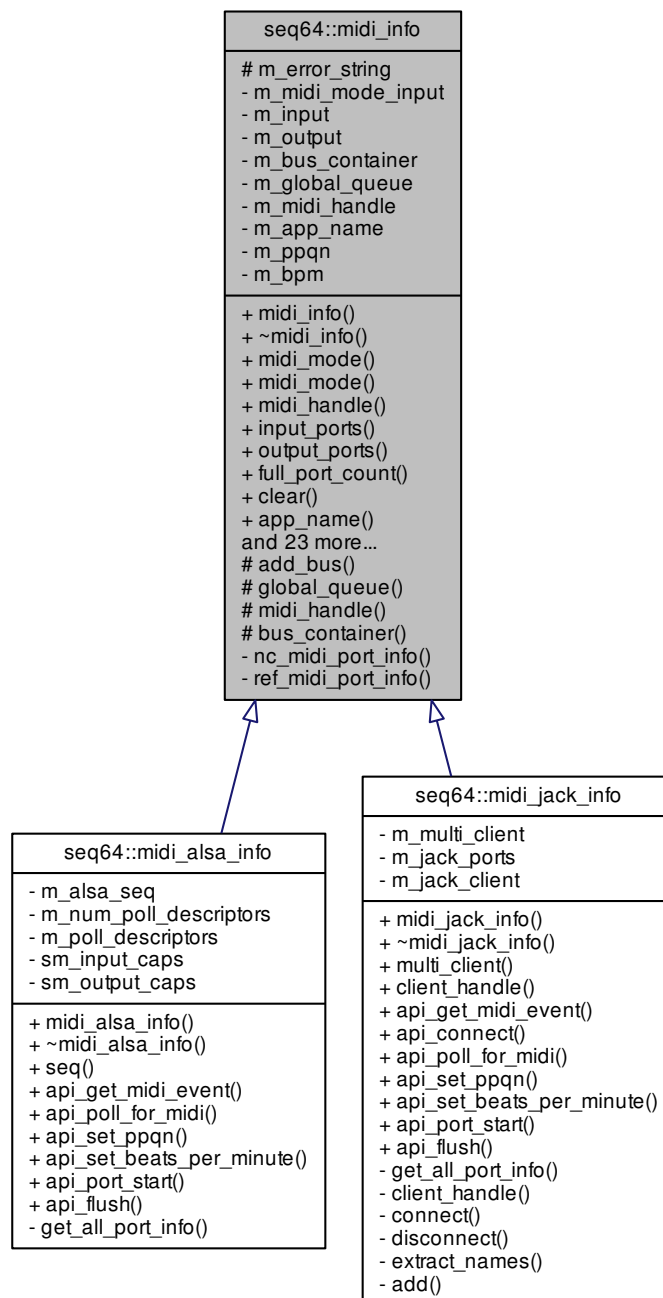
### 13.45.3.1 m\_client\_name

```
std::string seq64::midi_in_jack::m_client_name [protected]
```

## 13.46 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)  
*Principal constructor.*
- 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  
*Generates a string listing all of the ports present in the port container.*
- int [global\\_queue](#) () const
- void [error](#) ([rterror::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`

## 13.46.1 Constructor & Destructor Documentation

### 13.46.1.1 midi\_info()

```
seq64::midi_info::midi_info (
    const std::string & appname,
    int ppqn = SEQ64_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

### 13.46.1.2 ~midi\_info()

```
virtual seq64::midi_info::~~midi_info ( ) [inline], [virtual]
```

## 13.46.2 Member Function Documentation

### 13.46.2.1 midi\_mode() [1/2]

```
bool seq64::midi_info::midi_mode ( ) const [inline]
```

### 13.46.2.2 midi\_mode() [2/2]

```
void seq64::midi_info::midi_mode (
    bool flag ) [inline]
```

### 13.46.2.3 midi\_handle() [1/2]

```
void* seq64::midi_info::midi_handle ( ) [inline]
```

### 13.46.2.4 input\_ports()

```
midi\_port\_info& seq64::midi_info::input_ports ( ) [inline]
```

#### 13.46.2.5 output\_ports()

```
midi_port_info& seq64::midi_info::output_ports ( ) [inline]
```

#### 13.46.2.6 full\_port\_count()

```
int seq64::midi_info::full_port_count ( ) const [inline]
```

#### 13.46.2.7 clear()

```
void seq64::midi_info::clear ( ) [inline]
```

#### 13.46.2.8 app\_name()

```
const std::string& seq64::midi_info::app_name ( ) const [inline]
```

#### 13.46.2.9 ppqn()

```
int seq64::midi_info::ppqn ( ) const [inline]
```

#### 13.46.2.10 bpm()

```
midibpm seq64::midi_info::bpm ( ) const [inline]
```

#### 13.46.2.11 api\_set\_ppqn()

```
virtual void seq64::midi_info::api_set_ppqn (
    int p ) [inline], [virtual]
```

Reimplemented in [seq64::midi\\_jack\\_info](#), and [seq64::midi\\_alsa\\_info](#).

#### 13.46.2.12 api\_set\_beats\_per\_minute()

```
virtual void seq64::midi_info::api_set_beats_per_minute (
    midibpm b ) [inline], [virtual]
```

Reimplemented in [seq64::midi\\_jack\\_info](#), and [seq64::midi\\_alsa\\_info](#).

#### 13.46.2.13 api\_port\_start()

```
virtual void seq64::midi_info::api_port_start (
    mastermidibus & ,
    int ,
    int ) [inline], [virtual]
```

Reimplemented in [seq64::midi\\_jack\\_info](#), and [seq64::midi\\_alsa\\_info](#).

#### 13.46.2.14 api\_get\_midi\_event()

```
virtual bool seq64::midi_info::api_get_midi_event (
    event * inev ) [pure virtual]
```

Implemented in [seq64::midi\\_jack\\_info](#), and [seq64::midi\\_alsa\\_info](#).

#### 13.46.2.15 api\_poll\_for\_midi()

```
virtual int seq64::midi_info::api_poll_for_midi ( ) [pure virtual]
```

Implemented in [seq64::midi\\_jack\\_info](#), and [seq64::midi\\_alsa\\_info](#).

#### 13.46.2.16 api\_flush()

```
virtual void seq64::midi_info::api_flush ( ) [pure virtual]
```

Implemented in [seq64::midi\\_jack\\_info](#), and [seq64::midi\\_alsa\\_info](#).

#### 13.46.2.17 api\_connect()

```
virtual bool seq64::midi_info::api_connect ( ) [inline], [virtual]
```

Reimplemented in [seq64::midi\\_jack\\_info](#).

**13.46.2.18 get\_port\_count()**

```
virtual int seq64::midi_info::get_port_count ( ) const [inline], [virtual]
```

**13.46.2.19 get\_bus\_id()**

```
virtual int seq64::midi_info::get_bus_id (
    int index ) const [inline], [virtual]
```

**13.46.2.20 get\_bus\_name()**

```
virtual std::string seq64::midi_info::get_bus_name (
    int index ) const [inline], [virtual]
```

**13.46.2.21 get\_port\_id()**

```
virtual int seq64::midi_info::get_port_id (
    int index ) const [inline], [virtual]
```

**13.46.2.22 get\_port\_name()**

```
virtual std::string seq64::midi_info::get_port_name (
    int index ) const [inline], [virtual]
```

**13.46.2.23 get\_input()**

```
virtual bool seq64::midi_info::get_input (
    int index ) const [inline], [virtual]
```

**13.46.2.24 get\_virtual()**

```
virtual bool seq64::midi_info::get_virtual (
    int index ) const [inline], [virtual]
```

#### 13.46.2.25 get\_system()

```
virtual bool seq64::midi_info::get_system (
    int index ) const [inline], [virtual]
```

#### 13.46.2.26 queue\_number()

```
virtual int seq64::midi_info::queue_number (
    int index ) const [inline], [virtual]
```

#### 13.46.2.27 connect\_name()

```
std::string seq64::midi_info::connect_name (
    int index ) const [inline]
```

#### 13.46.2.28 port\_list()

```
std::string seq64::midi_info::port_list ( ) const
```

Useful for debugging and probing.

##### Returns

Returns a multi-line ASCII string enumerating all of the ports.

#### 13.46.2.29 global\_queue() [1/2]

```
int seq64::midi_info::global_queue ( ) const [inline]
```

#### 13.46.2.30 error()

```
void seq64::midi_info::error (
    rterror::Type type,
    const std::string & errorstring )
```

Provides an error handler.

Unlike the [midi\\_api](#) version, it cannot support an error callback.

## Exceptions

<i>If</i>	the error is not just a warning, then an <code>rterror</code> object is thrown.
-----------	---

## Parameters

<i>type</i>	The type of the error.
<i>errorstring</i>	The error message, which gets copied if this is the first error.

13.46.2.31 `get_all_port_info()`

```
virtual int seq64::midi_info::get_all_port_info ( ) [pure virtual]
```

Implemented in [seq64::midi\\_jack\\_info](#), and [seq64::midi\\_alsa\\_info](#).

13.46.2.32 `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.

13.46.2.33 `global_queue()` [2/2]

```
void seq64::midi_info::global_queue (
    int q ) [inline], [protected]
```

13.46.2.34 `midi_handle()` [2/2]

```
void seq64::midi_info::midi_handle (
    void * h ) [inline], [protected]
```

13.46.2.35 `bus_container()`

```
std::vector<midibus *>& seq64::midi_info::bus_container ( ) [inline], [protected]
```



### 13.46.2.36 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.

### 13.46.2.37 ref\_midi\_port\_info()

```
midi_port_info& seq64::midi_info::ref_midi_port_info ( ) [inline], [private]
```

## 13.46.3 Friends And Related Function Documentation

### 13.46.3.1 rtmidi\_info

```
friend class rtmidi_info [friend]
```

## 13.46.4 Field Documentation

### 13.46.4.1 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.

### 13.46.4.2 m\_input

```
midi_port_info seq64::midi_info::m_input [private]
```

### 13.46.4.3 m\_output

```
midi_port_info seq64::midi_info::m_output [private]
```

#### 13.46.4.4 m\_bus\_container

```
std::vector<midibus *> seq64::midi_info::m_bus_container [private]
```

See the [add\\_bus\(\)](#) and [bus\\_container\(\)](#) member functions.

#### 13.46.4.5 m\_global\_queue

```
int seq64::midi_info::m_global_queue [private]
```

#### 13.46.4.6 m\_midi\_handle

```
void* seq64::midi_info::m_midi_handle [private]
```

Created by the class derived from [midi\\_info](#).

#### 13.46.4.7 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.

#### 13.46.4.8 m\_ppqn

```
int seq64::midi_info::m_ppqn [private]
```

Some APIs can use this value.

#### 13.46.4.9 m\_bpm

```
midibpm seq64::midi_info::m_bpm [private]
```

Some APIs can use this value.

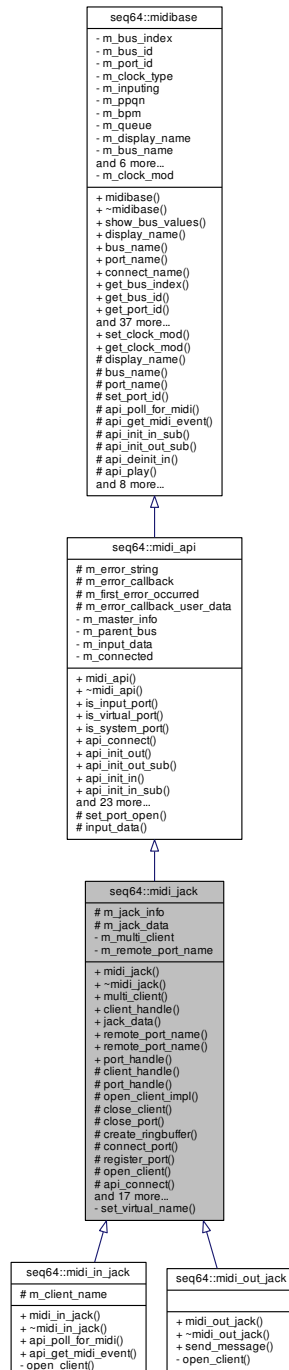
#### 13.46.4.10 m\_error\_string

```
std::string seq64::midi_info::m_error_string [protected]
```

## 13.47 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)

*We still need to figure out if we want a "master" client handle, or a handle to each port.*

- virtual `~midi_jack ()`  
*This could be a rote empty destructor if we offload this destruction to the `midi_jack_data` structure.*
- bool `multi_client ()` const  
*'Getter' function for member `m_multi_client`*
- `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)`  
*Opens input or output JACK clients, sets up the input or output callback, and activates the JACK client.*
- void `close_client ()`  
*Closes the JACK client handle.*
- void `close_port ()`  
*Closes the MIDI port by calling `jack_port_unregister()` and nullifying the port pointer.*
- bool `create_ringbuffer (size_t rbsize)`  
*Creates the JACK ring-buffers.*
- bool `connect_port (bool input, const std::string &sourceportname, const std::string &destportname)`  
*Connects two named JACK ports.*
- bool `register_port (bool input, const std::string &portname)`  
*Registers a named JACK port.*
- virtual bool `open_client ()=0`
- virtual bool `api_connect ()`  
*Assumes that the port has already been registered, and that JACK activation has already occurred.*
- virtual bool `api_init_out ()`  
*Initialize the MIDI output port.*
- virtual bool `api_init_in ()`  
*This function is called when we are processing the list of system input ports.*
- virtual bool `api_init_out_sub ()`
- virtual bool `api_init_in_sub ()`  
*Initializes a virtual/manual input port.*
- virtual bool `api_deinit_in ()`  
*We could define these in the opposite order.*
- virtual bool `api_get_midi_event (event *)`
- virtual int `api_poll_for_midi ()`
- virtual void `api_play (event *e24, midibyte channel)`  
*We could push the bytes of the event into a midibyte vector, as done in `send_message()`.*

- virtual void [api\\_sysex](#) ([event](#) \*e24)
- virtual void [api\\_flush](#) ()  
*It seems like JACK doesn't have the concept of flushing event.*
- virtual void [api\\_continue\\_from](#) ([midipulse](#) tick, [midipulse](#) beats)  
*jack\_transport\_locate(), jack\_transport\_reposition(), or something else?*
- virtual void [api\\_start](#) ()  
*Starts this JACK client.*
- virtual void [api\\_stop](#) ()  
*Starts this JACK client.*
- 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](#) ()  
*Gets the name of the current port via jack\_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

- bool [set\\_virtual\\_name](#) (int portid, const std::string &portname)  
*Gets information directly from JACK.*

## Private Attributes

- bool [m\\_multi\\_client](#)  
*Set to true if each JACK port should be its own client.*
- 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

### 13.47.1 Constructor & Destructor Documentation

#### 13.47.1.1 `midi_jack()`

```
seq64::midi_jack::midi_jack (
    midibus & parentbus,
    midi_info & masterinfo )
```

Same for the JACK data item. Currently, we provide the `m_multi_client` member so that we can experiment, but "multi-client" mode is currently incompletely implemented.

Note that this constructor also adds its object to the [midi\\_jack\\_info](#) port list, so that the JACK callback functions can iterate through all of the JACK ports in use by this application, performing work on them.

## Parameters

<i>parentbus</i>	Provides a reference to the midibus that represents this object.
<i>masterinfo</i>	Provides a reference to the <a href="#">midi_jack_info</a> object that may provide extra informatino that is needed by this port. Too many entities!

## 13.47.1.2 ~midi\_jack()

```
seq64::midi_jack::~~midi_jack ( ) [virtual]
```

However, other than the initialization, that structure is currently "dumb".

## 13.47.2 Member Function Documentation

## 13.47.2.1 multi\_client()

```
bool seq64::midi_jack::multi_client ( ) const [inline]
```

## 13.47.2.2 client\_handle() [1/2]

```
jack_client_t* seq64::midi_jack::client_handle ( ) [inline]
```

## 13.47.2.3 jack\_data()

```
midi\_jack\_data& seq64::midi_jack::jack_data ( ) [inline]
```

## 13.47.2.4 remote\_port\_name() [1/2]

```
const std::string& seq64::midi_jack::remote_port_name ( ) const [inline]
```

**13.47.2.5 remote\_port\_name()** [2/2]

```
void seq64::midi_jack::remote_port_name (
    const std::string & s ) [inline]
```

**13.47.2.6 port\_handle()** [1/2]

```
jack_port_t* seq64::midi_jack::port_handle ( ) [inline]
```

**13.47.2.7 client\_handle()** [2/2]

```
void seq64::midi_jack::client_handle (
    jack_client_t * handle ) [inline], [protected]
```

**13.47.2.8 port\_handle()** [2/2]

```
void seq64::midi_jack::port_handle (
    jack_port_t * handle ) [inline], [protected]
```

**13.47.2.9 open\_client\_impl()**

```
bool seq64::midi_jack::open_client_impl (
    bool input ) [protected]
```

This code is combined from the former versions of the `midi_in_jack::connect()` and `midi_out_jack::connect()` functions for better readability and re-use in the input and output `open_client()` functions.

For input, it connects the MIDI input port. The following calls are made:

- `jack_client_open()`, to initialize JACK client.
- `jack_set_process_callback()`, to set `jack_process_rtmidi_input()` or `jack_process_rtmidi_output()`.

Note that `jack_activate()` is no longer called here.

For output, connects the MIDI output port. The following calls are made:

- `jack_ringbuffer_create()`, called twice, to initialize the output ringbuffers
- `jack_client_open()`, to initialize JACK client
- `jack_set_process_callback()`, to set `jack_process_inpu()`

Note that `jack_activate()` is no longer called here.

If the `midi_jack_data` client member is already set, this function returns immediately. Only one client needs to be open for each `midi_jack` object.

Let's replace `JackNullOption` with `JackNoStartServer`. We might also want to OR in the `JackUseExactName` option.

The former name of this function was a bit of a misnomer, since it does not actually call `jack_connect()`. That call is made in other functions.

Which "client" name? Let's start with the full name, `connect_name()`. Is UUID an output-only, input-only option, or both?

```
const char * name = master_info().get_bus_name(get_bus_index()).c_str();
const char * name = master_info().get_port_name(get_bus_index()).c_str();
```



## Parameters

<i>input</i>	True if an input connection is to be made, and false if an output connection is to be made.
--------------	---

## 13.47.2.10 close\_client()

```
void seq64::midi_jack::close_client ( ) [protected]
```

## 13.47.2.11 close\_port()

```
void seq64::midi_jack::close_port ( ) [protected]
```

## 13.47.2.12 create\_ringbuffer()

```
bool seq64::midi_jack::create_ringbuffer (
    size_t rbsize ) [protected]
```

## 13.47.2.13 connect\_port()

```
bool seq64::midi_jack::connect_port (
    bool input,
    const std::string & srcportname,
    const std::string & destportname ) [protected]
```

First, we register the local port. If this is nominally a local input port, it is really doing output, and this is the source-port name. If this is nominally a local output port, it is really accepting input, and this is the destination-port name.

This code is disabled for now because the order of JACK setup calls that works is

```
- jack_port_register()
- jack_activate()
- jack_connect()
```

So we have to break this up.

## Parameters

<i>input</i>	Indicates true if the port to register and connect is an input port, and false if the port is an output port. Useful macros for readability: SEQ64_MIDI_INPUT_PORT and SEQ64_MIDI_OUTPUT_PORT.
<i>srcportname</i> Generated by Doxygen	Provides the destination port-name for the connection. For input, this should be the name associated with the JACK client handle; it is the port that gets registered. For output, this should be the full name of the port that was enumerated at start-up. The JackPortFlags of the source port must include JackPortIsOutput.
<i>destportname</i>	For input, this should be full name of port that was enumerated at start-up. For output, this

**Returns**

If the `jack_connect()` call succeeds, `true` is returned. If the port is a virtual (manual) port, then it is not connected, and `true` is returned without any action.

**13.47.2.14 register\_port()**

```
bool seq64::midi_jack::register_port (
    bool input,
    const std::string & portname ) [protected]
```

We made this function to encapsulate some otherwise cut-and-paste functionality.

For `jack_port_register()`, the port-name must be the actual port-name, not the full-port name ("busname:portname").

Note that the buffer size of non-built-in port type is 0, and so it is ignored.

**Tricky Code** If we are registering an input port, this means that we got the input port from the system. In order to connect to that port, we have to register as an output port, even though the application calls it in input port ([midi\\_in\\_jack](#)). Confusing, but the same thing was implicit in the ALSA implementation, and so we have to apply that same setup here.

**Parameters**

<i>input</i>	Indicates true if the port to register input port, and false if the port is an output port. Two macros can be used for this purpose: <code>SEQ64_MIDI_INPUT_PORT</code> and <code>SEQ64_MIDI_OUTPUT_PORT</code> .
<i>portname</i>	Provides the local name of the port. This is the full name ("clientname:portname"), but the "portname" part is extracted to fit the requirements of the <code>jack_port_register()</code> function. There is an issue here when <code>a2jmidid</code> is running. We may see a client name of "seq64", but a port name of "a2j Midi Through [1] capture: Midi Through Port-0", which as a colon in it. What to do? Just not extract the port name from the portname parameter. If we have an issue here, we'll have to fix it in the caller.

**13.47.2.15 open\_client()**

```
virtual bool seq64::midi_jack::open_client ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi\\_out\\_jack](#), and [seq64::midi\\_in\\_jack](#).

**13.47.2.16 api\_connect()**

```
bool seq64::midi_jack::api_connect ( ) [protected], [virtual]
```

**Returns**

Returns true if all steps of the connection succeeded.

Reimplemented from [seq64::midi\\_api](#).

## 13.47.2.17 api\_init\_out()

```
bool seq64::midi_jack::api_init_out ( ) [protected], [virtual]
```

This initialization is done when the "manual ports" option is not in force. This code is basically what was done by `midi_out_jack::open_port()` in `RtMidi`.

For `jack_connect()`, the first port-name is the source port, and the source port must include the `JackPortIsOutput` flag. The second port-name is the destination port, and the destination port must include the `JackPortIsInput` flag. For this function, the source/output port is this port, and the destination/input is....

Note that `connect_port()` [which calls `jack_connect()`] cannot usefully be called until `jack_activate()` has been called.

## Returns

Returns true unless setting up JACK MIDI failed in some way.

Implements [seq64::midi\\_api](#).

## 13.47.2.18 api\_init\_in()

```
bool seq64::midi_jack::api_init_in ( ) [protected], [virtual]
```

We want to create an output port of a similar name, but with the application as client, and connect it to this sytem input port. We want to follow the model we got from `seq24`, rather than the `RtMidi` model, so that we do not have to rework (and probably break) the `seq24` model.

We can't use the API port name here at this time, because it comes up empty. It comes up empty because we haven't yet registered the ports, including the source ports. So we register it first; `connect_port()` will not try to register it again.

Based on the comments in the `jack.txt` note file, here is what we need to do:

```
-# open_client_impl(INPUT).
-# Get port name via master_info().connect_name().
-# Call jack_client_open() with or without a UUID.
-# Call jack_set_process_callback() for input/output.
-# Call jack_activate(). Premature?
-# register_port(INPUT...). The flag is JackPortIsInput.
-# connect_port(INPUT...). Call jack_connect(srcport, destport).
```

Unlike the corresponding virtual port, this input port is actually an output port.

## Returns

Returns true if the function was successful, and sets the flag indicating that the port is open.

Implements [seq64::midi\\_api](#).

#### 13.47.2.19 `api_init_out_sub()`

```
bool seq64::midi_jack::api_init_out_sub ( ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.47.2.20 `api_init_in_sub()`

```
bool seq64::midi_jack::api_init_in_sub ( ) [protected], [virtual]
```

##### Returns

Returns true if all steps of the initialization succeeded.

Implements [seq64::midi\\_api](#).

#### 13.47.2.21 `api_deinit_in()`

```
bool seq64::midi_jack::api_deinit_in ( ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.47.2.22 `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](#).

### 13.47.2.23 api\_poll\_for\_midi()

```
virtual int seq64::midi_jack::api_poll_for_midi ( ) [inline], [protected], [virtual]
```

#### Returns

Returns 0, 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](#).

### 13.47.2.24 api\_play()

```
void seq64::midi_jack::api_play (
    event * e24,
    midibyte channel ) [protected], [virtual]
```

The ALSA code ([seq\\_alsamidi/src/midibus.cpp](#)) sticks the event bytes in an array, which might be a little faster than using `push_back()`, but let's try the vector first. The `rtmidi` code here is from [midi\\_out\\_jack::send\\_message\(\)](#).

Implements [seq64::midi\\_api](#).

### 13.47.2.25 api\_sysex()

```
void seq64::midi_jack::api_sysex (
    event * e24 ) [protected], [virtual]
```

**Todo** Flesh out this routine.

Implements [seq64::midi\\_api](#).

### 13.47.2.26 api\_flush()

```
void seq64::midi_jack::api_flush ( ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.47.2.27 `api_continue_from()`

```
void seq64::midi_jack::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [virtual]
```

What is used by [jack\\_assistant](#)?

Implements [seq64::midi\\_api](#).

#### 13.47.2.28 `api_start()`

```
void seq64::midi_jack::api_start ( ) [protected], [virtual]
```

Note that the [jack\\_assistant](#) code (which implements JACK transport) checks if JACK is running, but a check of the JACK client handle here should be enough.

Implements [seq64::midi\\_api](#).

#### 13.47.2.29 `api_stop()`

```
void seq64::midi_jack::api_stop ( ) [protected], [virtual]
```

Note that the [jack\\_assistant](#) code (which implements JACK transport) checks if JACK is running, but a check of the JACK client handle here should be enough.

Implements [seq64::midi\\_api](#).

#### 13.47.2.30 `api_clock()`

```
void seq64::midi_jack::api_clock (
    midipulse tick ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.47.2.31 `api_set_ppqn()`

```
void seq64::midi_jack::api_set_ppqn (
    int ppqn ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

## 13.47.2.32 api\_set\_beats\_per\_minute()

```
void seq64::midi_jack::api_set_beats_per_minute (
    midibpm bpm ) [protected], [virtual]
```

Implements [seq64::midi\\_api](#).

## 13.47.2.33 api\_get\_port\_name()

```
std::string seq64::midi_jack::api_get_port_name ( ) [protected], [virtual]
```

This is different from `get_port_name(index)`, which simply looks up the port-name in the attached [midi\\_info](#) object.

## Returns

Returns the full port name ("clientname:portname") if the port has already been opened/registered; otherwise an empty string is returned.

Reimplemented from [seq64::midi\\_api](#).

## 13.47.2.34 set\_virtual\_name()

```
bool seq64::midi_jack::set_virtual_name (
    int portid,
    const std::string & portname ) [private]
```

The problem this function solves is that the midibus constructor for a virtual JACK port doesn't not have all of the information it needs at that point. Here, we can get this information and get the actual data we need to rename the port to something accurate.

```
const char * pname = jack_port_name(const jack_port_t *);
```

## Returns

Returns true if all of the information could be obtained. If false is returned, then the caller should not use the side-effects.

**Side-effect(s)** Passes back the values found.

## 13.47.3 Friends And Related Function Documentation

## 13.47.3.1 midi\_jack\_info

```
friend class midi\_jack\_info [friend]
```

### 13.47.4 Field Documentation

#### 13.47.4.1 `m_multi_client`

```
bool seq64::midi_jack::m_multi_client [private]
```

In this case, the functions `api_init_in()`, `api_init_out()`, `api_init_in_sub()`, and `api_init_out_sub()` need to open their own JACK client. Otherwise, they will use the JACK client created in the `midi_jack_info` class.

#### 13.47.4.2 `m_remote_port_name`

```
std::string seq64::midi_jack::m_remote_port_name [private]
```

#### 13.47.4.3 `m_jack_info`

```
midi_jack_info& seq64::midi_jack::m_jack_info [protected]
```

#### 13.47.4.4 `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.

## 13.48 `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.*

### 13.48.1 Detailed Description

This guy needs a constructor taking parameters for an `rtmidi_in_data` pointer.

### 13.48.2 Constructor & Destructor Documentation

#### 13.48.2.1 `midi_jack_data()`

```
seq64::midi_jack_data::midi_jack_data ( ) [inline]
```

#### 13.48.2.2 `~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.

### 13.48.3 Member Function Documentation

#### 13.48.3.1 `valid_buffer()`

```
bool seq64::midi_jack_data::valid_buffer ( ) const [inline]
```

### 13.48.4 Field Documentation

#### 13.48.4.1 `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).

#### 13.48.4.2 `m_jack_port`

```
jack_port_t* seq64::midi_jack_data::m_jack_port
```

#### 13.48.4.3 `m_jack_buffsize`

```
jack_ringbuffer_t* seq64::midi_jack_data::m_jack_buffsize
```

#### 13.48.4.4 `m_jack_buffmessage`

```
jack_ringbuffer_t* seq64::midi_jack_data::m_jack_buffmessage
```

#### 13.48.4.5 `m_jack_lasttime`

```
jack_time_t seq64::midi_jack_data::m_jack_lasttime
```

Use for calculating the delta time, I would imagine.

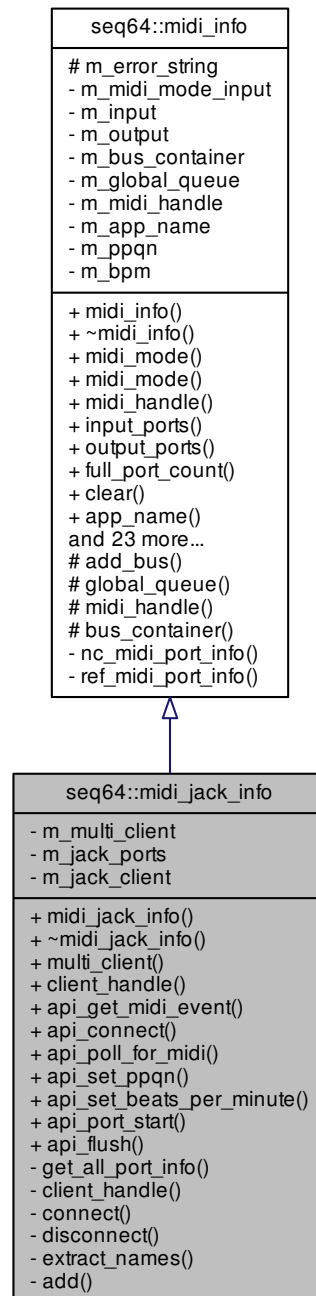
#### 13.48.4.6 `m_jack_rtmidiin`

```
rtmidi_in_data* seq64::midi_jack_data::m_jack_rtmidiin
```

## 13.49 seq64::midi\_jack\_info Class Reference

The class for handling JACK MIDI port enumeration.

Inheritance diagram for seq64::midi\_jack\_info:



### Public Member Functions

- `seq64::midi_jack_info` (const std::string &appname, int ppqn=SEQ64\_DEFAULT\_PPQN, midibpm bpm=SEQ64\_DEFAULT\_BPM)

*Principal constructor.*

- virtual `~midi_jack_info ()`

*Destructor.*

- bool `multi_client () const`

*'Getter' function for member m\_multi\_client*

- 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)`

*Grab a MIDI event.*

- virtual bool `api_connect ()`

*Sets up all of the ports, represented by midibus objects, that have been created.*

- virtual int `api_poll_for_midi ()`

*MUCH TO DO!*

- virtual void `api_set_ppqn (int p)`

*Sets the PPQN numeric value, then makes JACK calls to set up the PPQ tempo.*

- virtual void `api_set_beats_per_minute (midibpm b)`

*Sets the BPM numeric value, then makes JACK calls to set up the BPM tempo.*

- virtual void `api_port_start (mastermidibus &masterbus, int bus, int port)`

*Start the given JACK MIDI port.*

- virtual void `api_flush ()`

*Flushes our local queue events out into JACK.*

## Private Member Functions

- virtual int `get_all_port_info ()`

*Gets information on ALL ports, putting input data into one `midi_info` container, and putting output data into another `midi_info` container.*

- 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 ()`

*Local JACK connection for enumerating the ports.*

- void `disconnect ()`

*The opposite of `connect()`.*

- void `extract_names (const std::string &fullname, std::string &clientname, std::string &portname)`

*Extracts the two names from the JACK port-name format, "clientname:portname".*

- bool `add (midi_jack &mj)`

*Adds a pointer to a JACK port.*

## Private Attributes

- bool `m_multi_client`

*Set to true if each JACK port should be its own client.*

- 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.*

## Friends

- class [midi\\_jack](#)
- int [jack\\_process\\_io](#) (jack\_nframes\_t nframes, void \*arg)

*Provides a JACK callback function that uses the callbacks defined in the [midi\\_jack](#) module.*

## Additional Inherited Members

### 13.49.1 Constructor & Destructor Documentation

#### 13.49.1.1 midi\_jack\_info()

```
seq64::midi_jack_info::midi_jack_info (
    const std::string & appname,
    int ppqn = SEQ64_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

Note the m\_multi\_client member. We may want each JACK port to have its own client, as in the original RtMidi implementation.

#### Parameters

<i>appname</i>	Provides the name of the application.
<i>ppqn</i>	Provides the desired value of the PPQN (pulses per quarter note).
<i>bpm</i>	Provides the desired value of the BPM (beats per minute).

#### 13.49.1.2 ~midi\_jack\_info()

```
seq64::midi_jack_info::~~midi_jack_info ( ) [virtual]
```

Deactivates (disconnects and closes) any ports maintained by the JACK client, then closes the JACK client, shuts down the input thread, and then cleans up any API resources in use.

### 13.49.2 Member Function Documentation

#### 13.49.2.1 multi\_client()

```
bool seq64::midi_jack_info::multi_client ( ) const [inline]
```

**13.49.2.2 client\_handle()** [1/2]

```
jack_client_t* seq64::midi_jack_info::client_handle ( ) [inline]
```

**13.49.2.3 api\_get\_midi\_event()**

```
bool seq64::midi_jack_info::api_get_midi_event (
    event * inev ) [virtual]
```

**Parameters**

<i>inev</i>	The event to be set based on the found input event. We should make this value a reference someday. Not used here.
-------------	---

**Returns**

Always returns false. Will eventually delete this function.

Implements [seq64::midi\\_info](#).

**13.49.2.4 api\_connect()**

```
bool seq64::midi_jack_info::api_connect ( ) [virtual]
```

If multi-client usage has been specified, each non-virtual port that has been set up (with its own JACK client pointer) is activated, and then connected to its corresponding remote system port.

Otherwise, the main JACK client is activated, and then all non-virtual ports are simply connected.

Each JACK port's [midi\\_jack::api\\_connect\(\)](#) function decides, based on multi-client status, whether or not to activate before making the connection.

**Returns**

Returns true if activation succeeds.

Reimplemented from [seq64::midi\\_info](#).

**13.49.2.5 api\_poll\_for\_midi()**

```
int seq64::midi_jack_info::api_poll_for_midi ( ) [virtual]
```

Implements [seq64::midi\\_info](#).

**13.49.2.6 api\_set\_ppqn()**

```
void seq64::midi_jack_info::api_set_ppqn (
    int p ) [virtual]
```

## Parameters

<i>p</i>	The desired new PPQN value to set.
----------	------------------------------------

Reimplemented from [seq64::midi\\_info](#).

## 13.49.2.7 api\_set\_beats\_per\_minute()

```
void seq64::midi_jack_info::api_set_beats_per_minute (
    midibpm b ) [virtual]
```

## Parameters

<i>b</i>	The desired new BPM value to set.
----------	-----------------------------------

Reimplemented from [seq64::midi\\_info](#).

## 13.49.2.8 api\_port\_start()

```
void seq64::midi_jack_info::api_port_start (
    mastermidibus & masterbus,
    int bus,
    int port ) [virtual]
```

This function is called by [api\\_get\\_midi\\_event\(\)](#) when an JACK event SND\_SEQ\_EVENT\_PORT\_START is received.

- Get the API's client and port information.
- Do some capability checks.
- Find the client/port combination among the set of input/output busses. If it exists and is not active, then mark it as a replacement. If it is not a replacement, it will increment the number of input/output busses.

We can simplify this code a bit by using elements already present in [midi\\_jack\\_info](#).

## Parameters

<i>masterbus</i>	Provides the object needed to get access to the array of input and output buss objects.
<i>bus</i>	Provides the JACK bus/client number.
<i>port</i>	Provides the JACK client port.

Reimplemented from [seq64::midi\\_info](#).

### 13.49.2.9 `api_flush()`

```
void seq64::midi_jack_info::api_flush ( ) [virtual]
```

This is also a [midi\\_jack](#) function.

Implements [seq64::midi\\_info](#).

### 13.49.2.10 `get_all_port_info()`

```
int seq64::midi_jack_info::get_all_port_info ( ) [private], [virtual]
```

**Tricky Code** When we want to connect to a system input port, we want to use an output port to do that. When we want to connect to a system output port, we want to use an input port to do that. Therefore, we search for the *opposite* kind of port.

If in multi-client mode, then this function disconnects the JACK client afterward. At this point, we have got all the data we need, and are not providing a client to each JACK port we create.

If there is no system input port, or no system output port, then we add a virtual port of that type so that the application has something to work with.

Note that, at some pointer, we ought to consider how to deal with transitory system JACK clients and ports, and adjust for it. A kind of miniature form of session management. Also, don't forget about the usefulness of `jack_get↔_port_by_id()` and `jack_get_port_by_name()`.

Error handling:

Not having any JACK input ports present isn't necessarily an error. There may not be any, and there may still be at least one output port.

```
m_error_string = func_message("no JACK input ports available");  
error(rterror::WARNING, m_error_string);
```

Also, if there are none, we try to make a virtual port so that the application has something to work with. The only issue is the client number. Currently all virtual ports we create have a client number of 0.

JackPortIsPhysical:

If this flag is added, then only ports corresponding to a physical device are get detected and connected. This might be a useful option to add at a later date.

#### Returns

Returns the total number of ports found. Note that 0 ports is not necessarily an error; there may be no JACK apps running with exposed ports. If there is no JACK client, then -1 is returned.

Implements [seq64::midi\\_info](#).



#### 13.49.2.11 client\_handle() [2/2]

```
void seq64::midi_jack_info::client_handle (
    jack_client_t * j ) [inline], [private]
```

#### 13.49.2.12 connect()

```
jack_client_t * seq64::midi_jack_info::connect ( ) [private]
```

Note that this name will be used for normal ports, so we make sure it reflects the application name.

Note that this function does not call `jack_connect()`. We need to add a call to `jack_on_shutdown()` to set up a shutdown callback. We also need to wait on the activation call until we have registered all the ports. Then we (actually the mastermidibus) can call the [api\\_connect\(\)](#) function to activate this JACK client and connect all the ports.

```
jack_activate(result);
```

#### 13.49.2.13 disconnect()

```
void seq64::midi_jack_info::disconnect ( ) [private]
```

#### 13.49.2.14 extract\_names()

```
void seq64::midi_jack_info::extract_names (
    const std::string & fullname,
    std::string & clientname,
    std::string & portname ) [private]
```

#### 13.49.2.15 add()

```
bool seq64::midi_jack_info::add (
    midi_jack & mj ) [inline], [private]
```

### 13.49.3 Friends And Related Function Documentation

#### 13.49.3.1 midi\_jack

```
friend class midi_jack [friend]
```

### 13.49.3.2 jack\_process\_io

```
int jack_process_io (
    jack_nframes_t nframes,
    void * arg ) [friend]
```

## 13.49.4 Field Documentation

### 13.49.4.1 m\_multi\_client

```
bool seq64::midi_jack_info::m_multi_client [private]
```

In this case, the functions `api_init_in()`, `api_init_out()`, `api_init_in_sub()`, and `api_init_out_sub()` need to open their own JACK client. Otherwise, they will use the JACK client created here. And this class will have to close out its own client so it will not persist in the JACK client list (e.g. in `QJackCtl`).

### 13.49.4.2 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.

### 13.49.4.3 m\_jack\_client

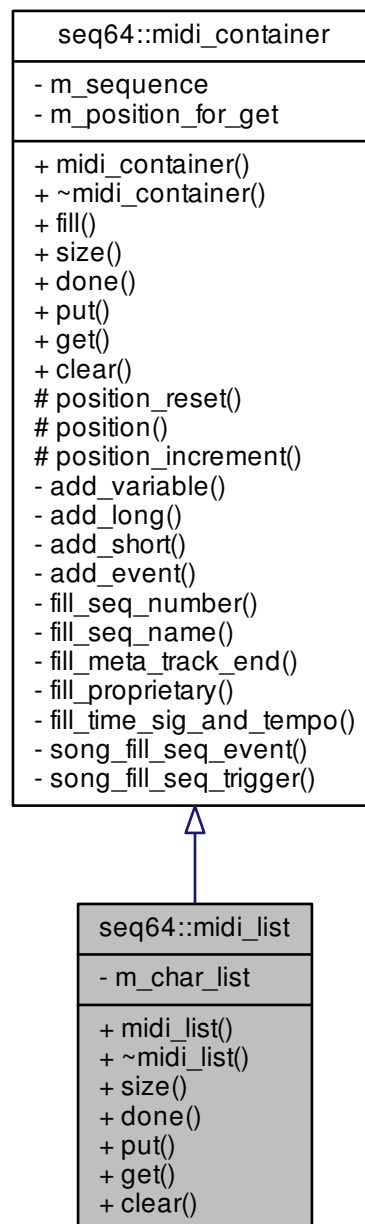
```
jack_client_t* seq64::midi_jack_info::m_jack_client [private]
```

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](#).

## 13.50 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

### 13.50.1 Member Typedef Documentation

#### 13.50.1.1 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.

### 13.50.2 Constructor & Destructor Documentation

#### 13.50.2.1 midi\_list()

```
seq64::midi_list::midi_list (
    sequence & seq )
```

#### Parameters

<code>seq</code>	The sequence/track object that is using this container.
------------------	---

### 13.50.2.2 ~midi\_list()

```
virtual seq64::midi_list::~~midi_list ( ) [inline], [virtual]
```

## 13.50.3 Member Function Documentation

### 13.50.3.1 size()

```
virtual std::size_t seq64::midi_list::size ( ) const [inline], [virtual]
```

Reimplemented from [seq64::midi\\_container](#).

### 13.50.3.2 done()

```
virtual bool seq64::midi_list::done ( ) const [inline], [virtual]
```

Reimplemented from [seq64::midi\\_container](#).

### 13.50.3.3 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](#).

### 13.50.3.4 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](#).

### 13.50.3.5 clear()

```
virtual void seq64::midi_list::clear ( ) [inline], [virtual]
```

Implements [seq64::midi\\_container](#).

## 13.50.4 Field Documentation

### 13.50.4.1 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.

## 13.51 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.*

### 13.51.1 Detailed Description

More commonly known as "bars:beats:ticks", or "BBT".

### 13.51.2 Constructor & Destructor Documentation

#### 13.51.2.1 midi\_measures() [1/2]

```
seq64::midi_measures::midi_measures ( )
```

#### 13.51.2.2 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.

### 13.51.3 Member Function Documentation

#### 13.51.3.1 measures() [1/2]

```
int seq64::midi_measures::measures ( ) const [inline]
```

#### 13.51.3.2 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.
----------	--

**13.51.3.3 beats()** [1/2]

```
int seq64::midi_measures::beats ( ) const [inline]
```

**13.51.3.4 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.
----------	---

**13.51.3.5 divisions()** [1/2]

```
int seq64::midi_measures::divisions ( ) const [inline]
```

**13.51.3.6 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.
----------	---

**13.51.4 Field Documentation****13.51.4.1 m\_measures**

```
int seq64::midi_measures::m_measures [private]
```



## 13.51.4.2 m\_beats

```
int seq64::midi_measures::m_beats [private]
```

## 13.51.4.3 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.

## 13.52 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` `mid_message` ()  
*Constructs an empty MIDI message.*
- `midibyte` `operator[]` (int i) const
- `midibyte` & `at` (int i)
- const `midibyte` & `at` (int i) const
- const char \* `array` () const
- int `count` () const
- bool `empty` () const
- void `push` (`midibyte` b)
- double `timestamp` () const
- void `timestamp` (double t)

## Private Attributes

- `container` `m_bytes`  
*Holds the event status and data bytes.*
- double `m_timestamp`  
*Holds the (optional) timestamp of the MIDI message.*

### 13.52.1 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.

### 13.52.2 Member Typedef Documentation

#### 13.52.2.1 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 [].

### 13.52.3 Constructor & Destructor Documentation

#### 13.52.3.1 midi\_message()

```
seq64::midi_message::midi_message ( )
```

### 13.52.4 Member Function Documentation

#### 13.52.4.1 operator[]()

```
midibyte seq64::midi_message::operator[] (
    int i ) const [inline]
```

#### 13.52.4.2 at() [1/2]

```
midibyte& seq64::midi_message::at (
    int i ) [inline]
```

#### 13.52.4.3 at() [2/2]

```
const midibyte& seq64::midi_message::at (
    int i ) const [inline]
```

#### 13.52.4.4 array()

```
const char* seq64::midi_message::array ( ) const [inline]
```

#### 13.52.4.5 count()

```
int seq64::midi_message::count ( ) const [inline]
```

#### 13.52.4.6 empty()

```
bool seq64::midi_message::empty ( ) const [inline]
```

#### 13.52.4.7 push()

```
void seq64::midi_message::push (
    midibyte b ) [inline]
```

#### 13.52.4.8 timestamp() [1/2]

```
double seq64::midi_message::timestamp ( ) const [inline]
```

#### 13.52.4.9 timestamp() [2/2]

```
void seq64::midi_message::timestamp (
    double t ) [inline]
```

### 13.52.5 Field Documentation

#### 13.52.5.1 m\_bytes

```
container seq64::midi_message::m_bytes [private]
```

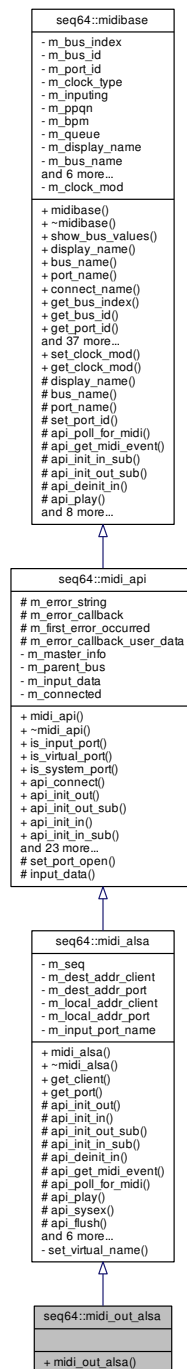
#### 13.52.5.2 m\_timestamp

```
double seq64::midi_message::m_timestamp [private]
```

### 13.53 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)

ALSA MIDI output normal port or virtual port constructor.

## Additional Inherited Members

### 13.53.1 Constructor & Destructor Documentation

#### 13.53.1.1 `midi_out_alsa()`

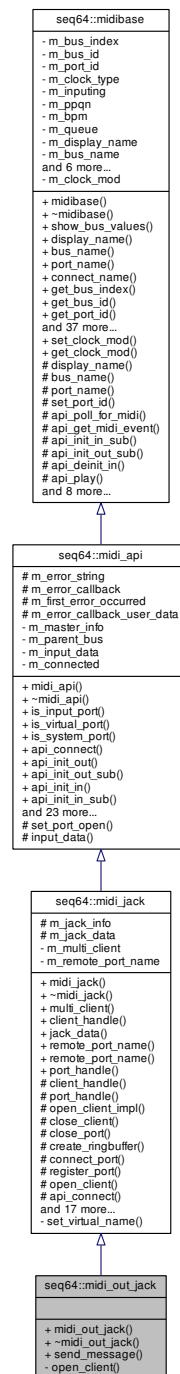
```
seq64::midi_out_alsa::midi_out_alsa (
    midibus & parentbus,
    midi_info & masterinfo )
```

The kind of port is determine by which port-initialization function the mastermidibus calls.

### 13.54 `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)  
*Principal constructor.*
- virtual [~midi\\_out\\_jack](#) ()  
*Destructor.*
- virtual bool [send\\_message](#) (const [midi\\_message::container](#) &message)  
*Sends a JACK MIDI output message.*

## 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

### 13.54.1 Constructor & Destructor Documentation

#### 13.54.1.1 [midi\\_out\\_jack\(\)](#)

```
seq64::midi_out_jack::midi_out_jack (
    midibus & parentbus,
    midi\_info & masterinfo )
```

For Sequencer64, we don't current need to create a [midi\\_out\\_jack](#) object; all that is needed is created via the [api\\_init\\_out\\*\(\)](#) functions.

#### Parameters

<i>parentbus</i>	Provides the buss object that determines buss-specific parameters of this class.
<i>masterinfo</i>	Provides information about the JACK system as found on this machine.

#### 13.54.1.2 [~midi\\_out\\_jack\(\)](#)

```
seq64::midi_out_jack::~~midi_out_jack ( ) [virtual]
```

Currently the base class closes the port, closes the JACK client, and cleans up the API data structure.

### 13.54.2 Member Function Documentation

#### 13.54.2.1 [send\\_message\(\)](#)

```
bool seq64::midi_out_jack::send_message (
    const midi\_message::container & message ) [virtual]
```

It writes the full message size and the message itself to the JACK ring buffer.



## Parameters

<code>message</code>	Provides the vector of message bytes to send.
----------------------	---

## Returns

Returns true if the buffer message and buffer size seem to be written correctly.

## 13.54.2.2 open\_client()

```
virtual bool seq64::midi_out_jack::open_client ( ) [inline], [private], [virtual]
```

This function replaces the RtMidi function "connect()".

Implements [seq64::midi\\_jack](#).

## 13.55 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](#) ()  
*Principal constructor.*
- 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)  
*Adds a set of port information to the port container.*
- void [add](#) (const [midibus](#) \*m)  
*Adds values from a midibus (actually a midibase-derived class).*
- 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".*

## 13.55.1 Constructor & Destructor Documentation

### 13.55.1.1 `midi_port_info()`

```
seq64::midi_port_info::midi_port_info ( )
```

## 13.55.2 Member Function Documentation

### 13.55.2.1 `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 )
```

#### Parameters

<i>clientnumber</i>	Provides the client or buss number for the port. This is a value like
<i>clientname</i>	Provides the system or user-supplied name for the client or buss.
<i>portnumber</i>	Provides the port number, usually re 0.
<i>portname</i>	Provides the system or user-supplied name for the port.
<i>makevirtual</i>	If the system currently has no input or output port available, then we want to create a virtual port so that the application has something to work with.
<i>makesystem</i>	In some systems, we need to create and activate a system port, such as a timer port or an ALSA announce port. For all other ports, this value is false.
<i>makeinput</i>	Indicates if the port is an input port or an output port.
<i>queuenumber</i>	Provides the optional queue number, if applicable. For example, the sequencer64 application grabs the client number (normally valued at 1) from the ALSA subsystem.

### 13.55.2.2 add() [2/2]

```
void seq64::midi_port_info::add (
    const midibus * m )
```

### 13.55.2.3 clear()

```
void seq64::midi_port_info::clear ( ) [inline]
```

### 13.55.2.4 get\_port\_count()

```
int seq64::midi_port_info::get_port_count ( ) const [inline]
```

### 13.55.2.5 get\_bus\_id()

```
int seq64::midi_port_info::get_bus_id (
    int index ) const [inline]
```

### 13.55.2.6 get\_bus\_name()

```
std::string seq64::midi_port_info::get_bus_name (
    int index ) const [inline]
```

### 13.55.2.7 get\_port\_id()

```
int seq64::midi_port_info::get_port_id (
    int index ) const [inline]
```

### 13.55.2.8 get\_port\_name()

```
std::string seq64::midi_port_info::get_port_name (
    int index ) const [inline]
```

#### 13.55.2.9 `get_input()`

```
bool seq64::midi_port_info::get_input (
    int index ) const [inline]
```

#### 13.55.2.10 `get_virtual()`

```
bool seq64::midi_port_info::get_virtual (
    int index ) const [inline]
```

#### 13.55.2.11 `get_system()`

```
bool seq64::midi_port_info::get_system (
    int index ) const [inline]
```

#### 13.55.2.12 `get_queue_number()`

```
int seq64::midi_port_info::get_queue_number (
    int index ) const [inline]
```

#### 13.55.2.13 `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.

### 13.55.3 Field Documentation

#### 13.55.3.1 `m_port_count`

```
int seq64::midi_port_info::m_port_count [private]
```

## 13.55.3.2 m\_port\_container

```
std::vector<port_info_t> seq64::midi_port_info::m_port_container [private]
```

## 13.56 seq64::midi\_queue Class Reference

Provides a queue of [midi\\_message](#) structures.

## Public Member Functions

- [midi\\_queue](#) ()  
*Default constructor.*
- [~midi\\_queue](#) ()  
*Destructor.*
- 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)  
*As long as we haven't reached our queue size limit, push the message.*
- void [pop](#) ()  
*Pops, so to speak, the front message out of the queue, effectively throwing it away.*
- [midi\\_message pop\\_front](#) ()  
*Pops a copy of the front message.*
- void [allocate](#) (unsigned queuesize=SEQ64\_DEFAULT\_QUEUE\_SIZE)  
*This would be better off as a constructor operation.*
- void [deallocate](#) ()  
*This would be better off as a destructor operation.*
- 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](#)

## 13.56.1 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.

## 13.56.2 Constructor & Destructor Documentation

### 13.56.2.1 midi\_queue()

```
seq64::midi_queue::midi_queue ( )
```

### 13.56.2.2 ~midi\_queue()

```
seq64::midi_queue::~~midi_queue ( )
```

## 13.56.3 Member Function Documentation

### 13.56.3.1 empty()

```
bool seq64::midi_queue::empty ( ) const [inline]
```

### 13.56.3.2 count()

```
int seq64::midi_queue::count ( ) const [inline]
```

### 13.56.3.3 full()

```
bool seq64::midi_queue::full ( ) const [inline]
```

#### Returns

Returns true if the queue size is at maximum.

### 13.56.3.4 add()

```
bool seq64::midi_queue::add (
    const midi\_message & mmsg )
```

### 13.56.3.5 pop()

```
void seq64::midi_queue::pop ( )
```

One useful call sequence is:

```
    midi_message latest = queue.front();  
    queue.pop();
```

An alternative is to use the `pop_front()` function instead.

### 13.56.3.6 pop\_front()

```
midi_message seq64::midi_queue::pop_front ( )
```

Could be a little inefficient, since a couple of copies are made, and we cannot use return-code optimization.

Perhaps at some point we could use move semantics?

#### Returns

Returns a copy of the message that was in front before the popping. If the queue is empty, an empty (all zeros) message is returned. Can be checked with the `midi_message::empty()` function.

### 13.56.3.7 allocate()

```
void seq64::midi_queue::allocate (   
    unsigned queuesize = SEQ64_DEFAULT_QUEUE_SIZE )
```

But one step at a time.

### 13.56.3.8 deallocate()

```
void seq64::midi_queue::deallocate ( )
```

But one step at a time.

### 13.56.3.9 front()

```
const midi_message& seq64::midi_queue::front ( ) const [inline]
```

### 13.56.4 Field Documentation

#### 13.56.4.1 m\_front

`unsigned seq64::midi_queue::m_front [private]`

#### 13.56.4.2 m\_back

`unsigned seq64::midi_queue::m_back [private]`

#### 13.56.4.3 m\_size

`unsigned seq64::midi_queue::m_size [private]`

#### 13.56.4.4 m\_ring\_size

`unsigned seq64::midi_queue::m_ring_size [private]`

#### 13.56.4.5 m\_ring

`midi\_message\* seq64::midi_queue::m_ring [private]`

## 13.57 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 new sequences, one for each 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.*

### 13.57.1 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.

## 13.57.2 Constructor & Destructor Documentation

### 13.57.2.1 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"> <li>• Reading. <ul style="list-style-type: none"> <li>– If set to SEQ64_USE_DEFAULT_PPQN, 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>.</li> <li>– 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.</li> </ul> </li> <li>• 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.</li> </ul>
-------------	---

## 13.57.2.2 ~midi\_splitter()

```
seq64::midi_splitter::~~midi_splitter ( )
```

## 13.57.3 Member Function Documentation

## 13.57.3.1 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.

## 13.57.3.2 initialize()

```
void seq64::midi_splitter::initialize ( )
```

## 13.57.3.3 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.
----------------	---

**13.57.3.4 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.

**13.57.3.5 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.

**13.57.3.6 count()**

```
int seq64::midi_splitter::count ( ) const [inline]
```

## 13.57.3.7 split\_channel()

```
bool seq64::midi_splitter::split_channel (
    const sequence & main_seq,
    sequence * s,
    int channel ) [private]
```

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.

## 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*.

## 13.57.4 Field Documentation

## 13.57.4.1 m\_ppqn

```
int seq64::midi_splitter::m_ppqn [private]
```

## 13.57.4.2 m\_use\_default\_ppqn

```
bool seq64::midi_splitter::m_use_default_ppqn [private]
```

#### 13.57.4.3 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.

#### 13.57.4.4 m\_smf0\_channels

```
bool seq64::midi_splitter::m_smf0_channels[16] [private]
```

Obviously, we don't have to worry about multiple MIDI busses.

#### 13.57.4.5 m\_smf0\_main\_sequence

```
sequence* seq64::midi_splitter::m_smf0_main_sequence [private]
```

#### 13.57.4.6 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.

## 13.58 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.*

### 13.58.1 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.

### 13.58.2 Constructor & Destructor Documentation

#### 13.58.2.1 `midi_timing()` [1/2]

```
seq64::midi_timing::midi_timing ( )
```

#### 13.58.2.2 `midi_timing()` [2/2]

```
seq64::midi_timing::midi_timing (
    midibpm bpminute,
    int bpmeasure,
    int beatwidth,
    int ppqn )
```

#### Parameters

<i>bpminute</i>	Copied into the <code>m_beats_per_minute</code> member.
<i>bpmeasure</i>	Copied into the <code>m_beats_per_measure</code> member.
<i>beatwidth</i>	Copied into the <code>m_beat_width</code> member.
<i>ppqn</i>	Copied into the <code>m_ppqn</code> member.

### 13.58.3 Member Function Documentation

**13.58.3.1 beats\_per\_minute()** [1/2]

```
midibpm seq64::midi_timing::beats_per_minute ( ) const [inline]
```

**13.58.3.2 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.
----------	--

**13.58.3.3 beats\_per\_measure()** [1/2]

```
int seq64::midi_timing::beats_per_measure ( ) const [inline]
```

**13.58.3.4 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.
----------	---

**13.58.3.5 beat\_width()** [1/2]

```
int seq64::midi_timing::beat_width ( ) const [inline]
```

**13.58.3.6 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.
-----------	--

## 13.58.3.7 ppqn() [1/2]

```
int seq64::midi_timing::ppqn ( ) const [inline]
```

## 13.58.3.8 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.
----------	---

## 13.58.4 Field Documentation

## 13.58.4.1 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.

## 13.58.4.2 m\_beats\_per\_measure

```
int seq64::midi_timing::m_beats_per_measure [private]
```

This value is most commonly set to 4.

## 13.58.4.3 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.

#### 13.58.4.4 m\_ppqn

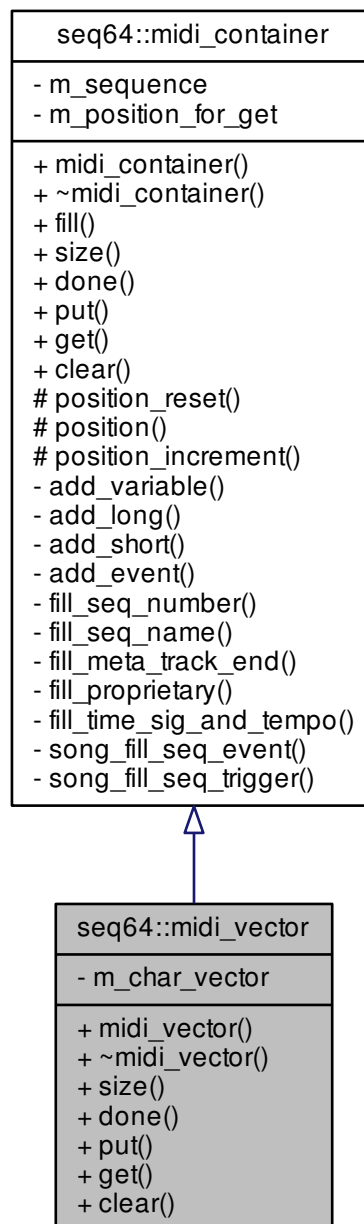
```
int seq64::midi_timing::m_ppqn [private]
```

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.

## 13.59 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

### 13.59.1 Member Typedef Documentation

#### 13.59.1.1 CharVector

```
typedef std::vector<midibyte> seq64::midi_vector::CharVector [private]
```

### 13.59.2 Constructor & Destructor Documentation

#### 13.59.2.1 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.
------------------	--

### 13.59.2.2 ~midi\_vector()

```
virtual seq64::midi_vector::~~midi_vector ( ) [inline], [virtual]
```

## 13.59.3 Member Function Documentation

### 13.59.3.1 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](#).

### 13.59.3.2 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](#).

### 13.59.3.3 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](#).

#### 13.59.3.4 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](#).

#### 13.59.3.5 clear()

```
virtual void seq64::midi_vector::clear ( ) [inline], [virtual]
```

Implements [seq64::midi\\_container](#).

### 13.59.4 Field Documentation

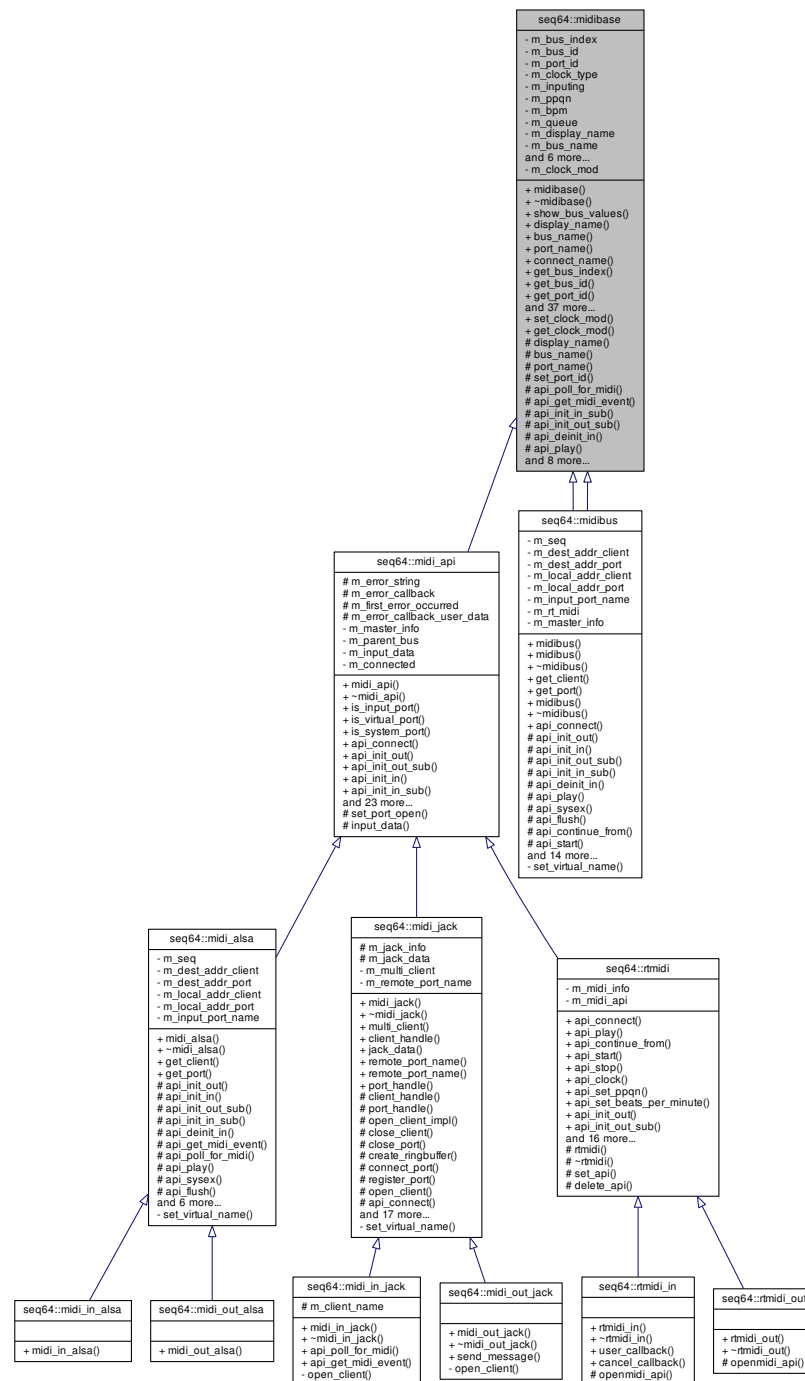
#### 13.59.4.1 m\_char\_vector

```
CharVector seq64::midi_vector::m_char_vector [private]
```

## 13.60 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*
- `clock_e get_clock () const`  
*'Getter' function for member m\_clock\_type*
- void `set_clock_status (clock_e clocktype)`  
*'Setter' 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.*
- 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 [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.*

### 13.60.1 Constructor & Destructor Documentation

## 13.60.1.1 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 <a href="#">init_in()</a> , <a href="#">init_out()</a> , <a href="#">init_in_sub()</a> , or <a href="#">init_out_sub()</a> 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).

### 13.60.1.2 ~midibase()

```
seq64::midibase::~~midibase ( ) [virtual]
```

## 13.60.2 Member Function Documentation

### 13.60.2.1 show\_bus\_values()

```
void seq64::midibase::show_bus_values ( )
```

### 13.60.2.2 display\_name() [1/2]

```
const std::string& seq64::midibase::display_name ( ) const [inline]
```

### 13.60.2.3 bus\_name() [1/2]

```
const std::string& seq64::midibase::bus_name ( ) const [inline]
```

### 13.60.2.4 port\_name() [1/2]

```
const std::string& seq64::midibase::port_name ( ) const [inline]
```

### 13.60.2.5 connect\_name()

```
std::string seq64::midibase::connect_name ( ) const
```

If either name is empty, an empty string is returned.

### 13.60.2.6 get\_bus\_index()

```
int seq64::midibase::get_bus_index ( ) const [inline]
```

#### 13.60.2.7 get\_bus\_id()

```
int seq64::midibase::get_bus_id ( ) const [inline]
```

#### 13.60.2.8 get\_port\_id()

```
int seq64::midibase::get_port_id ( ) const [inline]
```

#### 13.60.2.9 ppqn()

```
int seq64::midibase::ppqn ( ) const [inline]
```

#### 13.60.2.10 bpm()

```
midibpm seq64::midibase::bpm ( ) const [inline]
```

#### 13.60.2.11 match()

```
bool seq64::midibase::match (
    int bus,
    int port ) [inline]
```

#### 13.60.2.12 is\_virtual\_port() [1/2]

```
bool seq64::midibase::is_virtual_port ( ) const [inline]
```

#### 13.60.2.13 is\_virtual\_port() [2/2]

```
void seq64::midibase::is_virtual_port (
    bool flag ) [inline]
```

**13.60.2.14 is\_input\_port()** [1/2]

```
bool seq64::midibase::is_input_port ( ) const [inline]
```

**13.60.2.15 is\_output\_port()**

```
bool seq64::midibase::is_output_port ( ) const [inline]
```

**13.60.2.16 is\_input\_port()** [2/2]

```
void seq64::midibase::is_input_port (
    bool flag ) [inline]
```

**13.60.2.17 is\_system\_port()**

```
bool seq64::midibase::is_system_port ( ) const [inline]
```

**13.60.2.18 set\_system\_port\_flag()**

```
void seq64::midibase::set_system_port_flag ( ) [inline]
```

**13.60.2.19 set\_clock()**

```
void seq64::midibase::set_clock (
    clock_e clocktype ) [inline]
```

**Parameters**

<i>clocktype</i>	The value used to set the clock-type.
------------------	---------------------------------------

**13.60.2.20 get\_clock()**

```
clock_e seq64::midibase::get_clock ( ) const [inline]
```

**13.60.2.21 set\_clock\_status()**

```
void seq64::midibase::set_clock_status (
    clock_e clocktype ) [inline]
```

**13.60.2.22 get\_input()**

```
bool seq64::midibase::get_input ( ) const [inline]
```

**13.60.2.23 set\_input\_status()**

```
void seq64::midibase::set_input_status (
    bool flag ) [inline]
```

**13.60.2.24 queue\_number()**

```
int seq64::midibase::queue_number ( ) const [inline]
```

**13.60.2.25 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.)

**13.60.2.26 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

**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".



**13.60.2.27 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".

**13.60.2.28 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.

13.60.2.29 `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> .
-----------------	---

13.60.2.30 `get_clock_mod()`

```
static int seq64::midibase::get_clock_mod ( ) [inline], [static]
```

13.60.2.31 `poll_for_midi()`

```
int seq64::midibase::poll_for_midi ( )
```

## 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.

13.60.2.32 `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.

**13.60.2.33 init\_out()**

```
bool seq64::midibase::init_out ( )
```

**Returns**

Returns true unless setting up MIDI failed in some way.

**13.60.2.34 init\_in()**

```
bool seq64::midibase::init_in ( )
```

**Returns**

Returns true unless setting up MIDI failed in some way.

**13.60.2.35 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.

**13.60.2.36 init\_out\_sub()**

```
bool seq64::midibase::init_out_sub ( )
```

**Returns**

Returns true unless setting up the ALSA port failed in some way.

### 13.60.2.37 init\_in\_sub()

```
bool seq64::midibase::init_in_sub ( )
```

#### Returns

Returns true unless setting up the ALSA port failed in some way.

### 13.60.2.38 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.

### 13.60.2.39 sysex()

```
void seq64::midibase::sysex (
    event * e24 )
```

#### Parameters

<i>e24</i>	The event to be handled.
------------	--------------------------

### 13.60.2.40 flush()

```
void seq64::midibase::flush ( )
```

### 13.60.2.41 start()

```
void seq64::midibase::start ( )
```

### 13.60.2.42 stop()

```
void seq64::midibase::stop ( )
```

### 13.60.2.43 clock()

```
void seq64::midibase::clock (
    midipulse tick )
```

*Threadsafe*

#### Parameters

<i>tick</i>	Provides the starting tick.
-------------	-----------------------------

### 13.60.2.44 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.
-------------	----------------------

### 13.60.2.45 init\_clock()

```
void seq64::midibase::init_clock (
    midipulse tick )
```

This function doesn't depend upon the MIDI API in use.

#### Parameters

<i>tick</i>	The starting tick.
-------------	--------------------

**13.60.2.46 print()**

```
void seq64::midibase::print ( )
```

**13.60.2.47 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.
-----------------	---

**13.60.2.48 display\_name()** [2/2]

```
void seq64::midibase::display_name (
    const std::string & name ) [inline], [protected]
```

**13.60.2.49 bus\_name()** [2/2]

```
void seq64::midibase::bus_name (
    const std::string & name ) [inline], [protected]
```

**13.60.2.50 port\_name()** [2/2]

```
void seq64::midibase::port_name (
    const std::string & name ) [inline], [protected]
```

**13.60.2.51 set\_port\_id()**

```
void seq64::midibase::set_port_id (
    int id ) [inline], [protected]
```

### 13.60.2.52 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::midi\\_jack](#), [seq64::midi\\_alsa](#), [seq64::rtmidi](#), [seq64::midi\\_api](#), and [seq64::midibus](#).

### 13.60.2.53 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](#).

### 13.60.2.54 api\_init\_in\_sub()

```
virtual bool seq64::midibase::api_init_in_sub ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

### 13.60.2.55 api\_init\_out\_sub()

```
virtual bool seq64::midibase::api_init_out_sub ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

### 13.60.2.56 api\_deinit\_in()

```
virtual bool seq64::midibase::api_deinit_in ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

### 13.60.2.57 `api_play()`

```
virtual void seq64::midibase::api_play (
    event * e24,
    midibyte channel ) [protected], [pure virtual]
```

Implemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

### 13.60.2.58 `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::midibus](#), [seq64::midi\\_alsa](#), [seq64::rtmidi](#), and [seq64::midi\\_api](#).

### 13.60.2.59 `api_flush()`

```
virtual void seq64::midibase::api_flush ( ) [inline], [protected], [virtual]
```

Reimplemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::rtmidi](#), and [seq64::midi\\_api](#).

### 13.60.2.60 `api_init_in()`

```
virtual bool seq64::midibase::api_init_in ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).

### 13.60.2.61 `api_init_out()`

```
virtual bool seq64::midibase::api_init_out ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::rtmidi](#), and [seq64::midibus](#).



### 13.60.2.62 api\_continue\_from()

```
virtual void seq64::midibase::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [pure virtual]
```

Implemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

### 13.60.2.63 api\_start()

```
virtual void seq64::midibase::api_start ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

### 13.60.2.64 api\_stop()

```
virtual void seq64::midibase::api_stop ( ) [protected], [pure virtual]
```

Implemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

### 13.60.2.65 api\_clock()

```
virtual void seq64::midibase::api_clock (
    midipulse tick ) [protected], [pure virtual]
```

Implemented in [seq64::midi\\_jack](#), [seq64::midibus](#), [seq64::midi\\_alsa](#), [seq64::midi\\_api](#), [seq64::midibus](#), and [seq64::rtmidi](#).

## 13.60.3 Friends And Related Function Documentation

### 13.60.3.1 mastermidibus

```
friend class mastermidibus [friend]
```

## 13.60.4 Field Documentation

#### 13.60.4.1 m\_clock\_mod

```
int seq64::midibase::m_clock_mod [static], [private]
```

Initialize this static member.

#### 13.60.4.2 m\_bus\_index

```
const int seq64::midibase::m_bus_index [private]
```

Otherwise, it is currently -1.

#### 13.60.4.3 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).

#### 13.60.4.4 m\_port\_id

```
int seq64::midibase::m_port_id [private]
```

#### 13.60.4.5 m\_clock\_type

```
clock_e seq64::midibase::m_clock_type [private]
```

#### 13.60.4.6 m\_inputting

```
bool seq64::midibase::m_inputting [private]
```

It is turned on if the user selects the port in the Options / MIDI Input tab.

#### 13.60.4.7 m\_ppqn

```
int seq64::midibase::m_ppqn [private]
```

Some APIs can control or use this value.

#### 13.60.4.8 m\_bpm

```
midibpm seq64::midibase::m_bpm [private]
```

Some APIs can control or use this value.

#### 13.60.4.9 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.

#### 13.60.4.10 m\_display\_name

```
std::string seq64::midibase::m_display_name [private]
```

Assembled by the [set\\_name\(\)](#) function.

#### 13.60.4.11 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.

#### 13.60.4.12 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.

#### 13.60.4.13 m\_lasttick

```
midipulse seq64::midibase::m_lasttick [private]
```

#### 13.60.4.14 m\_is\_virtual\_port

```
bool seq64::midibase::m_is_virtual_port [private]
```

The default is to create a system port (true).

#### 13.60.4.15 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.

#### 13.60.4.16 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.

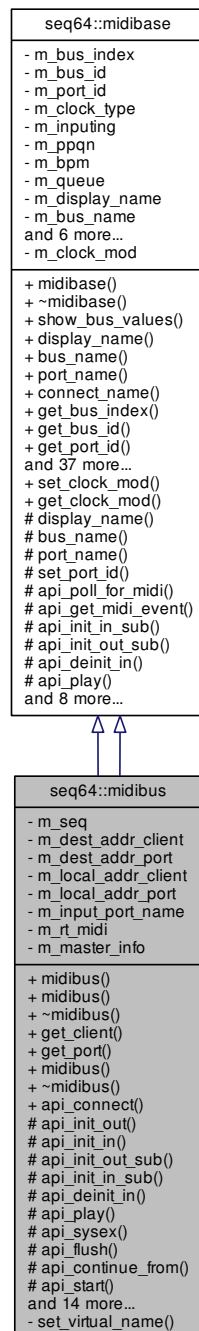
#### 13.60.4.17 m\_mutex

```
mutex seq64::midibase::m_mutex [private]
```

## 13.61 seq64::midibus Class Reference

This class implements with ALSA version of the midibus object.

Inheritance diagram for seq64::midibus:



## Public Member Functions

- `midibus` (int localclient, int destclient, int destport, snd\_seq\_t \*seq, const std::string &client\_name, const std::string &port\_name, int index, int queue, int ppqn=SEQ64\_USE\_DEFAULT\_PPQN, midibpm bpm=SEQ64\_DEFAULT\_BPM)
- `midibus` (int localclient, snd\_seq\_t \*seq, int index, int bus\_id, int queue, int ppqn=SEQ64\_USE\_DEFAULT\_PPQN, midibpm bpm=SEQ64\_DEFAULT\_BPM)

- virtual `~midibus ()`  
*The destructor closes out the RtMidi MIDI infrastructure.*
- 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*
- `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)`  
*Normal-port and virtual-port constructor.*
- virtual `~midibus ()`
- virtual bool `api_connect ()`  
*Connects to another port.*

## Protected Member Functions

- virtual bool `api_init_out ()`  
*Initializes the MIDI output port.*
- virtual bool `api_init_in ()`  
*Initializes the MIDI input port.*
- virtual bool `api_init_out_sub ()`  
*Initializes the MIDI virtual output port.*
- virtual bool `api_init_in_sub ()`  
*Initializes the MIDI virtual input port.*
- virtual bool `api_deinit_in ()`  
*Forwards the de-initialization call to the API object that implements it.*
- virtual void `api_play (event *e24, midibyte channel)`  
*Takes a native event, and encodes to a Windows message, and writes it to the queue.*
- virtual void `api_sysex (event *e24)`
- virtual void `api_flush ()`
- virtual void `api_continue_from (midipulse tick, midipulse beats)`  
*Continue from the given tick.*
- virtual void `api_start ()`  
*Sets the MIDI clock a-runnin', if the clock type is not e\_clock\_off.*
- virtual void `api_stop ()`  
*Stops the MIDI clock, if the clock-type is not e\_clock\_off.*
- virtual void `api_clock (midipulse tick)`  
*Generates MIDI clock.*
- 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)`  
*Gets a MIDI event.*
- virtual int `api_poll_for_midi ()`  
*Polls for MIDI events.*
- 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 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.*
- [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 ALSA sequencer "handle".*

## Friends

- class [mastermidibus](#)  
*The master MIDI bus sets up the buss.*

## Additional Inherited Members

### 13.61.1 Detailed Description

This class implements with rtmidi version of the midibus object.

### 13.61.2 Constructor & Destructor Documentation

#### 13.61.2.1 midibus() [1/3]

```
seq64::midibus::midibus (
    int localclient,
    int destclient,
    int destport,
    snd_seq_t * seq,
    const std::string & client_name,
    const std::string & port_name,
    int index,
    int queue,
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

**13.61.2.2** `midibus()` [2/3]

```
seq64::midibus::midibus (
    int localclient,
    snd_seq_t * seq,
    int index,
    int bus_id,
    int queue,
    int ppqn = SEQ64_USE_DEFAULT_PPQN,
    midibpm bpm = SEQ64_DEFAULT_BPM )
```

**13.61.2.3** `~midibus()` [1/2]

```
seq64::midibus::~~midibus ( ) [virtual]
```

**13.61.2.4** `midibus()` [3/3]

```
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 )
```

**Parameters**

<i>rt</i>	Provides the <code>rtmidi_info</code> object to use to obtain the client ID (buss ID), port ID, and port name, as obtained via calls to the ALSA, JACK, Core MIDI, or Windows MM subsystems. We need it to provide the single ALSA "handle" needed in the in Sequencer64 buss model, where the master MIDI buss provides it to be used by all the MIDI buss objects.
<i>index</i>	This is the index into the <code>rtmidi</code> object, and is used to get the desired client and port information. It is an index into the info container held by the <code>rtmidi</code> object.
<i>makevirtual</i>	Indicates that the port is virtual, as opposed to normal.
<i>isinput</i>	Indicates that the port is an input port, as opposed to an output port.
<i>bussoverride</i>	Optional buss ID, if not equal to the index parameter.
<i>makesystem</i>	Indicates that the port is also a system port (i.e. always present).

**13.61.2.5** `~midibus()` [2/2]

```
virtual seq64::midibus::~~midibus ( ) [virtual]
```



### 13.61.3 Member Function Documentation

#### 13.61.3.1 get\_client()

```
virtual int seq64::midibus::get_client ( ) const [inline], [virtual]
```

#### 13.61.3.2 get\_port()

```
virtual int seq64::midibus::get_port ( ) const [inline], [virtual]
```

#### 13.61.3.3 api\_init\_out() [1/2]

```
bool seq64::midibus::api_init_out ( ) [protected], [virtual]
```

Currently, we use the default values for the rtmidi API, the queue number, and the queue size.

##### Returns

Returns true if the output port was successfully opened.

Implements [seq64::midibase](#).

#### 13.61.3.4 api\_init\_in() [1/2]

```
bool seq64::midibus::api_init_in ( ) [protected], [virtual]
```

##### Returns

Returns true if the input port was successfully opened.

Implements [seq64::midibase](#).

### 13.61.3.5 `api_init_out_sub()` [1/2]

```
bool seq64::midibus::api_init_out_sub ( ) [protected], [virtual]
```

#### Returns

Returns true if the output port was successfully opened.

Reimplemented from [seq64::midibase](#).

### 13.61.3.6 `api_init_in_sub()` [1/2]

```
bool seq64::midibus::api_init_in_sub ( ) [protected], [virtual]
```

#### Returns

Returns true if the input port was successfully opened.

Reimplemented from [seq64::midibase](#).

### 13.61.3.7 `api_deinit_in()` [1/2]

```
bool seq64::midibus::api_deinit_in ( ) [protected], [virtual]
```

We don't bother checking the `m_rt_midi` pointer. If it is null, it is the programmer's fault.

#### Returns

Returns the result of the `m_rt_midi->api_deinit_in()` call.

Reimplemented from [seq64::midibase](#).

### 13.61.3.8 `api_play()` [1/2]

```
void seq64::midibus::api_play (
    event * e24,
    midibyte channel ) [protected], [virtual]
```

It fills a small byte buffer, sets the MIDI channel, make a message of it, and writes the message.

Again, DO WE NEED to distinguish between input and output here?

Note that we're doing a double-forwarding here, which may lower throughput.

## Parameters

<i>e24</i>	The MIDI event to play.
<i>channel</i>	The channel on which to play the event.

Implements [seq64::midibase](#).

### 13.61.3.9 api\_sysex()

```
virtual void seq64::midibus::api_sysex (
    event * e24 ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

### 13.61.3.10 api\_flush()

```
virtual void seq64::midibus::api_flush ( ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

### 13.61.3.11 api\_continue\_from() [1/2]

```
void seq64::midibus::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [virtual]
```

This function implements only the RtMidi-specific code.

Note that, unlike in PortMidi, here we do not deal with zeroing the event timestamp.

## Parameters

<i>tick</i>	The tick to continue from; unused in the RtMidi API implementation.
<i>beats</i>	The calculated beats. This calculation is made in the <a href="#">midibase::continue_from()</a> function.

Implements [seq64::midibase](#).

### 13.61.3.12 api\_start() [1/2]

```
void seq64::midibus::api_start ( ) [protected], [virtual]
```

This function is called by [midibase::start\(\)](#). No timestamp handling.

Implements [seq64::midibase](#).

### 13.61.3.13 `api_stop()` [1/2]

```
void seq64::midibus::api_stop ( ) [protected], [virtual]
```

This function is called by [midibase::stop\(\)](#). No timestamp handling.

Implements [seq64::midibase](#).

### 13.61.3.14 `api_clock()` [1/2]

```
void seq64::midibus::api_clock (
    midipulse tick ) [protected], [virtual]
```

This function is called by [midibase::clock\(\)](#). No timestamp handling.

#### Parameters

<i>tick</i>	The clock tick value, not used in the API implementation of this function for RtMidi.
-------------	---

Implements [seq64::midibase](#).

### 13.61.3.15 `set_virtual_name()`

```
bool seq64::midibus::set_virtual_name (
    int portid,
    const std::string & portname ) [private]
```

### 13.61.3.16 `api_connect()`

```
bool seq64::midibus::api_connect ( ) [virtual]
```

If the port is an input port, but is not configured (by the user or the "rc" configuration file), then it is not connected, and this is not an error. Output ports are always connected.

Note that the [api\\_connect\(\)](#) function will errprint its own errors. But if we expected to be able to connect, and have a null `rt_midi` pointer, then this is a reported error.

#### Returns

Returns true if the connection was made successfully, or there was no need to make the connection.

**13.61.3.17** `api_init_in()` [2/2]

```
virtual bool seq64::midibus::api_init_in ( ) [protected], [virtual]
```

Implements [seq64::midibase](#).

**13.61.3.18** `api_init_in_sub()` [2/2]

```
virtual bool seq64::midibus::api_init_in_sub ( ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

**13.61.3.19** `api_init_out()` [2/2]

```
virtual bool seq64::midibus::api_init_out ( ) [protected], [virtual]
```

Implements [seq64::midibase](#).

**13.61.3.20** `api_init_out_sub()` [2/2]

```
virtual bool seq64::midibus::api_init_out_sub ( ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

**13.61.3.21** `api_deinit_in()` [2/2]

```
virtual bool seq64::midibus::api_deinit_in ( ) [protected], [virtual]
```

Reimplemented from [seq64::midibase](#).

**13.61.3.22** `api_get_midi_event()`

```
bool seq64::midibus::api_get_midi_event (
    event * inev ) [protected], [virtual]
```

**Parameters**

<i>inev</i>	The location to deposit the MIDI event data.
-------------	--

**Returns**

Returns true if a MIDI event was obtained.

Reimplemented from [seq64::midibase](#).

**13.61.3.23 api\_poll\_for\_midi()**

```
int seq64::midibus::api_poll_for_midi ( ) [protected], [virtual]
```

This is the API implementation for RtMidi.

This should work only for input busses, so we need to insure this at some point. Currently, this is the domain of the master bus. We also should make this routine just check the input queue size and then read the queue. Note that the ALSA handle checks incoming MIDI events and either passes them to the callback function or pushes them onto the input queue.

**Returns**

Returns 0 if the polling succeeded, and 1 if it failed. If the buss hasn't been initialized, it has a null m\_rt\_midi pointer, and will return 0. This can happen normally when a MIDI input port is configured to be disabled.

Reimplemented from [seq64::midibase](#).

**13.61.3.24 api\_continue\_from()** [2/2]

```
virtual void seq64::midibus::api_continue_from (
    midipulse tick,
    midipulse beats ) [protected], [virtual]
```

Implements [seq64::midibase](#).

**13.61.3.25 api\_start()** [2/2]

```
virtual void seq64::midibus::api_start ( ) [protected], [virtual]
```

Implements [seq64::midibase](#).

**13.61.3.26 api\_stop()** [2/2]

```
virtual void seq64::midibus::api_stop ( ) [protected], [virtual]
```

Implements [seq64::midibase](#).

### 13.61.3.27 api\_clock() [2/2]

```
virtual void seq64::midibus::api_clock (  
    midipulse tick ) [protected], [virtual]
```

Implements [seq64::midibase](#).

### 13.61.3.28 api\_play() [2/2]

```
virtual void seq64::midibus::api_play (  
    event * e24,  
    midibyte channel ) [protected], [virtual]
```

Implements [seq64::midibase](#).

## 13.61.4 Friends And Related Function Documentation

### 13.61.4.1 mastermidibus

```
mastermidibus [friend]
```

The master MIDI bus sets up the buss, so it gets access to private details.

## 13.61.5 Field Documentation

### 13.61.5.1 m\_seq

```
snd_seq_t* const seq64::midibus::m_seq [private]
```

### 13.61.5.2 m\_dest\_addr\_client

```
const int seq64::midibus::m_dest_addr_client [private]
```

Could potentially be replaced by [midibase::m\\_bus\\_id](#).

### 13.61.5.3 m\_dest\_addr\_port

```
const int seq64::midibus::m_dest_addr_port [private]
```

Could potentially be replaced by [midibase::m\\_port\\_id](#).

### 13.61.5.4 m\_local\_addr\_client

```
const int seq64::midibus::m_local_addr_client [private]
```

### 13.61.5.5 m\_local\_addr\_port

```
int seq64::midibus::m_local_addr_port [private]
```

### 13.61.5.6 m\_input\_port\_name

```
const std::string seq64::midibus::m_input_port_name [private]
```

It is derived from the (optionally configured) official client name for the application with the word "in" appended.

### 13.61.5.7 m\_rt\_midi

```
rtmidi* seq64::midibus::m_rt_midi [private]
```

### 13.61.5.8 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.

## 13.62 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.*
- [~midifile](#) ()  
*A rote destructor.*
- bool [parse](#) ([perform](#) &p, int a\_screen\_set=0)  
*This function opens a binary MIDI file and parses it into sequences and other application objects.*
- bool [write](#) ([perform](#) &p)  
*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 sequences when [parse\(\)](#) was called.*

## Private Member Functions

- 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.*
- int [pow2](#) (int logbase2)  
*Internal function for simple calculation of a power of 2 without a lot of math.*
- 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.*
- [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.*
- 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\\_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 [read\\_byte\\_array](#) (midibyte \*b, int len)

*A helper function to simplify reading [midi\\_control](#) data from the MIDI file.*

- 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.*

- void [errdump](#) (const std::string &msg)

*Helper function to emit more useful error messages.*

- void [errdump](#) (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.*
- [int 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.*
- [int 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 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.*

### 13.62.1 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.

### 13.62.2 Constructor & Destructor Documentation

#### 13.62.2.1 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"> <li>• Reading. <ul style="list-style-type: none"> <li>– If set to SEQ64_USE_DEFAULT_PPQN, 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>.</li> <li>– 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.</li> </ul> </li> <li>• 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.</li> </ul>
<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 <code>globals</code> 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.

## 13.62.2.2 ~midifile()

```
seq64::midifile::~~midifile ( )
```

## 13.62.3 Member Function Documentation

## 13.62.3.1 parse()

```
bool seq64::midifile::parse (
    perform & p,
    int screenset = 0 )
```

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 Seq24, handles SysEx message only to the extend of passing them via MIDI Thru. We hope to improve on that capability.

### 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.

### 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`.

#### 13.62.3.2 write()

```
bool seq64::midifile::write (
    perform & p )
```

Also see the [write\\_song\(\)](#) function, for exporting to standard MIDI.

Seq24 reverses the order of some events, due to popping from its container. Not an issue here.

**Parameters**

<i>p</i>	Provides the object that will contain and manage the entire performance.
----------	--

**Returns**

Returns true if the write operations succeeded.

**13.62.3.3 write\_song()**

```
bool seq64::midifile::write_song (
    perform & p )
```

**13.62.3.4 error\_message()**

```
const std::string& seq64::midifile::error_message ( ) const [inline]
```

**13.62.3.5 error\_is\_fatal()**

```
bool seq64::midifile::error_is_fatal ( ) const [inline]
```

**13.62.3.6 ppqn()**

```
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.

**13.62.3.7 parse\_smf\_0()**

```
bool seq64::midifile::parse_smf_0 (
    perform & p,
    int screenset ) [private]
```

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.

## 13.62.3.8 parse\_smf\_1()

```
bool seq64::midifile::parse_smf_1 (
    perform & p,
    int screenset,
    bool is_smf0 = false ) [private]
```

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.

## 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.

## 13.62.3.9 parse\_prop\_header()

```
midilong seq64::midifile::parse_prop_header (
    int file_size ) [private]
```

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 (0x242400nn), 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 <code>m_pos</code> (the position inside <code>m_data[]</code> ), 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.

#### 13.62.3.10 `parse_proprietary_track()`

```
bool seq64::midifile::parse_proprietary_track (
    perform & p,
    int file_size ) [private]
```

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 <a href="#">parse()</a> function.

There are also implicit parameters, with the `m_pos` and `m_new_format` member variables.

#### 13.62.3.11 pow2()

```
int seq64::midifile::pow2 (
    int logbase2 ) [private]
```

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.

#### 13.62.3.12 checklen()

```
bool seq64::midifile::checklen (
    midilong len,
    midibyte type ) [private]
```

A length of zero is now considered legal, but a "warning" message is shown. The largest value allowed within a MIDI file is 0xFFFFFFFF. 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.

**13.62.3.13 add\_trigger()**

```
void seq64::midifile::add_trigger (
    sequence & seq,
    midishort ppqn ) [private]
```

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 m_use_default_ppqn is true. If 0, the ppqn value is not used.

**13.62.3.14 read\_long()**

```
midilong seq64::midifile::read_long ( ) [private]
```

**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.

**13.62.3.15 read\_short()**

```
midishort seq64::midifile::read_short ( ) [private]
```

**Returns**

Returns the two bytes, shifted appropriately and added together, most-significant byte first, to sum to a short value.

### 13.62.3.16 read\_byte()

```
midibyte seq64::midifile::read_byte ( ) [private]
```

#### 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.

### 13.62.3.17 read\_varinum()

```
midilong seq64::midifile::read_varinum ( ) [private]
```

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.

### 13.62.3.18 write\_long()

```
void seq64::midifile::write_long (
    midilong x ) [private]
```

#### Warning

This code looks endian-dependent.

#### Parameters

x	The long value to be written to the MIDI file.
---	--

### 13.62.3.19 write\_short()

```
void seq64::midifile::write_short (
    midishort x ) [private]
```

#### Warning

This code looks endian-dependent.

## Parameters

<i>x</i>	The short value to be written to the MIDI file.
----------	---

## 13.62.3.20 read\_byte\_array()

```
void seq64::midifile::read_byte_array (
    midibyte * b,
    int len ) [inline], [private]
```

## 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.

## 13.62.3.21 write\_byte()

```
void seq64::midifile::write_byte (
    midibyte c ) [inline], [private]
```

The byte is written to the `m_char_list` member, using a call to `push_back()`.

## Parameters

<i>c</i>	The MIDI byte to be "written".
----------	--------------------------------

## 13.62.3.22 write\_varinum()

```
void seq64::midifile::write_varinum (
    midilong value ) [private]
```

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 (0x7F) are represented by a single byte.
- 0x80 is represented as "0x81 0x00".
- 0x0FFFFFFF (the largest number) is represented as "0xFF 0xFF 0xFF 0x7F".

Also see the [varinum\\_size\(\)](#) function.

#### Parameters

<i>value</i>	The long value to be encoded as a MIDI varinum, and written to the MIDI file.
--------------	---

#### 13.62.3.23 write\_track\_name()

```
void seq64::midifile::write_track_name (
    const std::string & trackname ) [private]
```

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.
------------------	--

#### 13.62.3.24 read\_track\_name()

```
std::string seq64::midifile::read_track_name ( ) [private]
```

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.

#### 13.62.3.25 write\_seq\_number()

```
void seq64::midifile::write_seq_number (
    midishort seqnum ) [private]
```

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.
---------------	-------------------------------

13.62.3.26 `read_seq_number()`

```
int seq64::midifile::read_seq_number ( ) [private]
```

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.

13.62.3.27 `write_track_end()`

```
void seq64::midifile::write_track_end ( ) [private]
```

13.62.3.28 `write_header()`

```
bool seq64::midifile::write_header (
    int numtracks ) [private]
```

- 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.
```

## 13.62.3.29 write\_prop\_header()

```
void seq64::midifile::write_prop_header (
    midilong control_tag,
    long data_length ) [private]
```

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.

## 13.62.3.30 write\_proprietary\_track()

```
bool seq64::midifile::write_proprietary_track (
    perform & p ) [private]
```

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.
```

**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.

**13.62.3.31 varinum\_size()**

```
long seq64::midifile::varinum_size (
    long len ) const [private]
```

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](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.

**13.62.3.32 prop\_item\_size()**

```
long seq64::midifile::prop_item_size (
    long data_length ) const [private]
```

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.

## 13.62.3.33 track\_name\_size()

```
long seq64::midifile::track_name_size (
    const std::string & trackname ) const [private]
```

## 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".

## 13.62.3.34 errdump() [1/2]

```
void seq64::midifile::errdump (
    const std::string & msg ) [private]
```

It adds the file offset to the message.

## Parameters

<i>msg</i>	The main error message string, without an ending newline character.
------------	---

## Returns

The constructed string is returned as a side-effect, in case we want to pass it along to the externally-accessible error-message buffer.

## 13.62.3.35 errdump() [2/2]

```
void seq64::midifile::errdump (
    const std::string & msg,
    unsigned long value ) [private]
```

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

The constructed string is returned as a side-effect, in case we want to pass it along to the externally-accessible error-message buffer.

#### 13.62.3.36 write\_track()

```
void seq64::midifile::write_track (
    const midi_vector & lst ) [private]
```

#### 13.62.3.37 seq\_number\_size()

```
long seq64::midifile::seq_number_size ( ) const [inline], [private]
```

#### 13.62.3.38 track\_end\_size()

```
long seq64::midifile::track_end_size ( ) const [inline], [private]
```

#### 13.62.3.39 is\_sysex\_special\_id()

```
bool seq64::midifile::is_sysex_special_id (
    midibyte ch ) [inline], [private]
```

#### Parameters

<i>ch</i>	Provides the byte to be checked against 0x7D through 0x7F.
-----------	--

#### Returns

Returns true if the byte is SysEx special ID.

## 13.62.4 Field Documentation

### 13.62.4.1 m\_mutex

```
mutex seq64::midifile::m_mutex [mutable], [private]
```

Made mutable for use in certain locked getter functions.

### 13.62.4.2 m\_file\_size

```
int 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.

### 13.62.4.3 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.

### 13.62.4.4 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\(\)](#).

### 13.62.4.5 m\_disable\_reported

```
bool seq64::midifile::m_disable_reported [private]
```

Once is enough.

### 13.62.4.6 m\_pos

```
int 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.

### 13.62.4.7 m\_name

```
const std::string seq64::midifile::m_name [private]
```

#### 13.62.4.8 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.

#### 13.62.4.9 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.

#### 13.62.4.10 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.

#### 13.62.4.11 m\_global\_bgsequence

```
bool seq64::midifile::m_global_bgsequence [private]
```

#### 13.62.4.12 m\_ppqn

```
int seq64::midifile::m_ppqn [private]
```

#### 13.62.4.13 m\_use\_default\_ppqn

```
bool seq64::midifile::m_use_default_ppqn [private]
```

#### 13.62.4.14 m\_smf0\_splitter

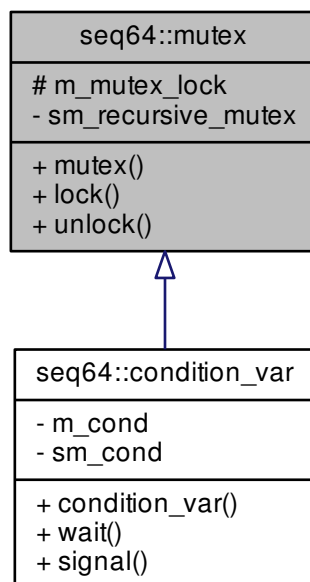
```
midi_splitter seq64::midifile::m_smf0_splitter [private]
```

This object holds all of the information needed to split a multi-channel sequence.

## 13.63 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.*

### 13.63.1 Constructor & Destructor Documentation

#### 13.63.1.1 mutex()

```
seq64::mutex::mutex ( )
```

### 13.63.2 Member Function Documentation

#### 13.63.2.1 lock()

```
void seq64::mutex::lock ( ) const
```

#### 13.63.2.2 unlock()

```
void seq64::mutex::unlock ( ) const
```

### 13.63.3 Field Documentation

#### 13.63.3.1 sm\_recursive\_mutex

```
const pthread_mutex_t seq64::mutex::sm_recursive_mutex [static], [private]
```

Define the static recursive mutex and its condition variable.

#### 13.63.3.2 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").

## 13.64 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.*

### 13.64.1 Field Documentation

#### 13.64.1.1 event\_value

```
unsigned short seq64::editable_event::name_value_t::event_value
```

#### 13.64.1.2 event\_name

```
std::string seq64::editable_event::name_value_t::event_name
```

## 13.65 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\\_mod\\_callback](#) (Gtk::Adjustment \*adj)
- 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.*

## 13.65.1 Member Enumeration Documentation



## 13.65.1.1 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, <a href="#">rc_settings :: with_jack_transport()</a> .
e_jack_master	Turns on the "with JACK Master" option, <a href="#">rc_settings :: with_jack_master()</a> . If another application is already JACK Master, this will fail.
e_jack_master_cond	Turns on the "with JACK Master" option <a href="#">rc_settings :: with_jack_master_cond()</a> . 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 <a href="#">rc_settings :: with_jack_midi()</a> . This is a setting independent of the JACK Transport settings. This is use only in the "rtmidi" implementation os Sequencer64, <a href="#">seq64</a> .
e_jack_start_mode_live	Doesn't directly do anything; the live mode versus song mode is set by the <a href="#">e_jack_start_mode_song</a> 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 <a href="#">rc_settings :: song_start_mode()</a> . 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, <a href="#">perform::init_jack()</a> , to be called.
e_jack_disconnect	Causes the perform object's JACK deinitialization function, <a href="#">perform::deinit_jack()</a> , to be called.

## 13.65.2 Constructor &amp; Destructor Documentation

## 13.65.2.1 options()

```
seq64::options::options (
    Gtk::Window & parent,
    perform & p,
    bool showjack = false )
```

## 13.65.3 Member Function Documentation

## 13.65.3.1 perf()

```
perform& seq64::options::perf ( ) [inline], [private]
```

#### 13.65.3.2 clock\_callback\_off()

```
void seq64::options::clock_callback_off (
    int bus,
    Gtk::RadioButton * button ) [private]
```

#### 13.65.3.3 clock\_callback\_on()

```
void seq64::options::clock_callback_on (
    int bus,
    Gtk::RadioButton * button ) [private]
```

#### 13.65.3.4 clock\_callback\_mod()

```
void seq64::options::clock_callback_mod (
    int bus,
    Gtk::RadioButton * button ) [private]
```

#### 13.65.3.5 clock\_mod\_callback()

```
void seq64::options::clock_mod_callback (
    Gtk::Adjustment * adj ) [private]
```

#### 13.65.3.6 input\_callback()

```
void seq64::options::input_callback (
    int bus,
    Gtk::Button * button ) [private]
```

#### 13.65.3.7 filter\_callback()

```
void seq64::options::filter_callback (
    Gtk::Button * button ) [private]
```

#### 13.65.3.8 transport\_callback()

```
void seq64::options::transport_callback (
    button type,
    Gtk::Button * button ) [private]
```

#### 13.65.3.9 mouse\_seq24\_callback()

```
void seq64::options::mouse_seq24_callback (
    Gtk::RadioButton * ) [private]
```

#### 13.65.3.10 mouse\_fruity\_callback()

```
void seq64::options::mouse_fruity_callback (
    Gtk::RadioButton * ) [private]
```

#### 13.65.3.11 mouse\_mod4\_callback()

```
void seq64::options::mouse_mod4_callback (
    Gtk::CheckButton * ) [private]
```

#### 13.65.3.12 mouse\_snap\_split\_callback()

```
void seq64::options::mouse_snap_split_callback (
    Gtk::CheckButton * ) [private]
```

#### 13.65.3.13 mouse\_click\_edit\_callback()

```
void seq64::options::mouse_click_edit_callback (
    Gtk::CheckButton * ) [private]
```

#### 13.65.3.14 lash\_support\_callback()

```
void seq64::options::lash_support_callback (
    Gtk::CheckButton * ) [private]
```

#### 13.65.3.15 add\_midi\_clock\_page()

```
void seq64::options::add_midi_clock_page ( ) [private]
```

#### 13.65.3.16 add\_midi\_input\_page()

```
void seq64::options::add_midi_input_page ( ) [private]
```

#### 13.65.3.17 add\_keyboard\_page()

```
void seq64::options::add_keyboard_page ( ) [private]
```

#### 13.65.3.18 add\_extended\_keys\_page()

```
void seq64::options::add_extended_keys_page ( ) [private]
```

#### 13.65.3.19 add\_mouse\_page()

```
void seq64::options::add_mouse_page ( ) [private]
```

#### 13.65.3.20 add\_jack\_sync\_page()

```
void seq64::options::add_jack_sync_page ( ) [private]
```

### 13.65.4 Field Documentation

#### 13.65.4.1 m\_tooltips

```
Gtk::Tooltips* seq64::options::m_tooltips [private]
```

#### 13.65.4.2 m\_mainperf

`perform& seq64::options::m_mainperf [private]`

#### 13.65.4.3 m\_button\_ok

`Gtk::Button* seq64::options::m_button_ok [private]`

#### 13.65.4.4 m\_button\_jack\_transport

`Gtk::CheckButton* seq64::options::m_button_jack_transport [private]`

#### 13.65.4.5 m\_button\_jack\_master

`Gtk::CheckButton* seq64::options::m_button_jack_master [private]`

#### 13.65.4.6 m\_button\_jack\_master\_cond

`Gtk::CheckButton* seq64::options::m_button_jack_master_cond [private]`

#### 13.65.4.7 m\_button\_jack\_connect

`Gtk::Button* seq64::options::m_button_jack_connect [private]`

#### 13.65.4.8 m\_button\_jack\_disconnect

`Gtk::Button* seq64::options::m_button_jack_disconnect [private]`

#### 13.65.4.9 m\_notebook

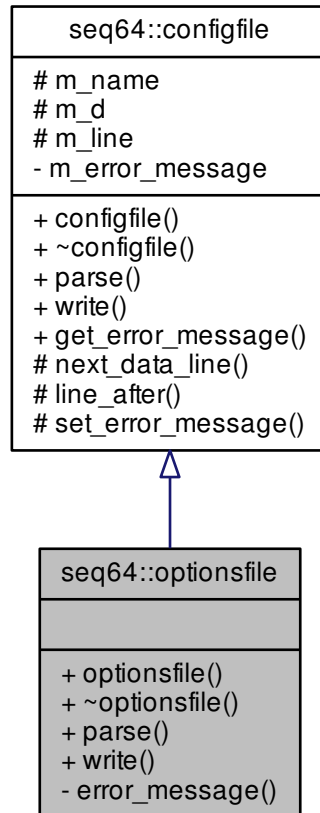
`Gtk::Notebook* seq64::options::m_notebook [private]`

Must be a GTK thang.

## 13.66 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 &perf)  
*Parse the ~/.seq24rc or ~/.config/sequencer64/sequencer64.rc file.*
- bool `write` (const perform &perf)  
*This options-writing function is just about as complex as the options-reading function.*

### Private Member Functions

- bool `error_message` (const std::string &sectionname, const std::string &additional="")  
*Helper function for error-handling.*

## Additional Inherited Members

### 13.66.1 Detailed Description

The settings that are passed around are provided or used by the perform class.

### 13.66.2 Constructor & Destructor Documentation

#### 13.66.2.1 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.
-------------	--

#### 13.66.2.2 ~optionsfile()

```
seq64::optionsfile::~~optionsfile ( )
```

### 13.66.3 Member Function Documentation

#### 13.66.3.1 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.

Implements [seq64::configfile](#).

#### 13.66.3.2 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](#).

#### 13.66.3.3 error\_message()

```
bool seq64::optionsfile::error_message (
    const std::string & sectionname,
    const std::string & additional = "" ) [private]
```

#### 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.

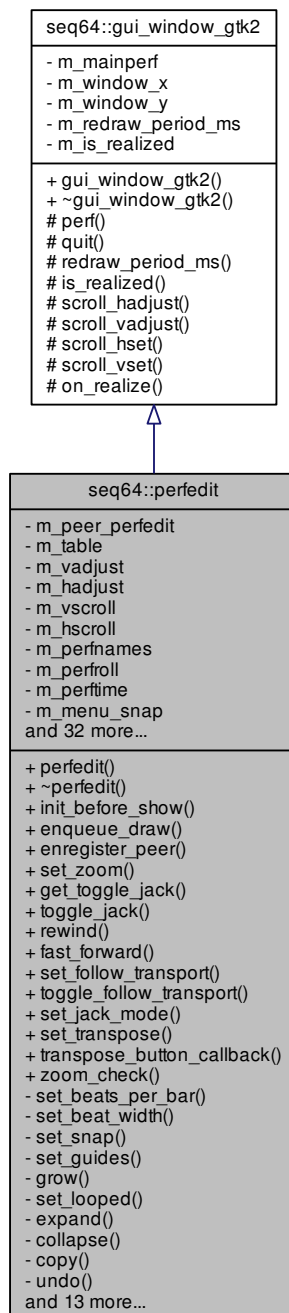
#### Returns

Always returns false.

## 13.67 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 `~perfeddit ()`

*This rote destructor does nothing.*
  - void `init_before_show ()`

*This function forwards its call to the `perffroll` function of the same name.*
  - void `enqueue_draw (bool forward=true)`

*Helper wrapper for calling `perffroll::queue_draw()` for one or both `perfedits`.*
  - void `enregister_peer (perfeddit *peer)`

*Register the peer `perfeddit` 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 `perffroll` 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 `perffroll` 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::Tooltips \* [m\\_tooltips](#)  
*Container for tool-tips.*
- 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

### 13.67.1 Detailed Description

It has a seqroll and piano roll? No, it has a perform, a perfnames, a perroll, and a perftime.

### 13.67.2 Constructor & Destructor Documentation

#### 13.67.2.1 `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.

## 13.67.2.2 ~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.

## 13.67.3 Member Function Documentation

## 13.67.3.1 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.

## 13.67.3.2 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.
----------------	--

## 13.67.3.3 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.
----------	-------------------------------------

**13.67.3.4 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.
-------------	--

**13.67.3.5 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.
----------	--

**13.67.3.6 get\_toggle\_jack()**

```
bool seq64::perfedit::get_toggle_jack ( )
```

**Returns**

Returns the JACK button's `get_active()` status.

**13.67.3.7 toggle\_jack()**

```
void seq64::perfedit::toggle_jack ( )
```

Note that this will trigger the button signal callback.



### 13.67.3.8 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.
--------------	--

### 13.67.3.9 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.
--------------	--

### 13.67.3.10 set\_follow\_transport()

```
void seq64::perfedit::set_follow_transport ( )
```

Note that this will trigger the button signal callback.

### 13.67.3.11 toggle\_follow\_transport()

```
void seq64::perfedit::toggle_follow_transport ( )
```

Note that this will trigger the button signal callback.

### 13.67.3.12 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.

### 13.67.3.13 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.
------------------	---

## 13.67.3.14 transpose\_button\_callback()

```
void seq64::perfedit::transpose_button_callback (
    int transpose )
```

## Parameters

<i>transpose</i>	The amount to transpose the transposable sequences.
------------------	---

## 13.67.3.15 set\_beats\_per\_bar()

```
void seq64::perfedit::set_beats_per_bar (
    int bpm ) [private]
```

The usage of `isModified` 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.
------------	---

## 13.67.3.16 set\_beat\_width()

```
void seq64::perfedit::set_beat_width (
    int bw ) [private]
```

The usage of `isModified` 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.
-----------	---

**13.67.3.17 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".
-------------	---

**13.67.3.18 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.

**13.67.3.19 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.

**13.67.3.20 set\_looped()**

```
void seq64::perfedit::set_looped ( ) [private]
```

**13.67.3.21 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 perfroll to redraw.

**13.67.3.22 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 perfroll to redraw.

### 13.67.3.23 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.

### 13.67.3.24 undo()

```
void seq64::perfedit::undo ( ) [private]
```

We pop an Undo trigger, and then ask the perffroll to queue up a (re)drawing action.

### 13.67.3.25 redo()

```
void seq64::perfedit::redo ( ) [private]
```

We pop an Redo trigger, and then ask the perffroll to queue up a (re)drawing action.

### 13.67.3.26 popup\_menu()

```
void seq64::perfedit::popup_menu (
    Gtk::Menu * menu ) [private]
```

### 13.67.3.27 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.

### 13.67.3.28 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.

### 13.67.3.29 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.
------------------	---

**13.67.3.30 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.

**13.67.3.31 pause\_playing()**

```
void seq64::perfedit::pause_playing ( ) [private]
```

Keeps the stop button enabled as a kind of rewind for ALSA. Stop in place!

**13.67.3.32 stop\_playing()**

```
void seq64::perfedit::stop_playing ( ) [private]
```

We need to make the progress line move back to the beginning right away here.

**13.67.3.33 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.

**13.67.3.34 on\_realize()**

```
void seq64::perfedit::on_realize ( ) [private]
```

**13.67.3.35 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.
-----------	--------------------------------------

**13.67.3.36 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.
-----------	--------------------------------------

**13.67.3.37 on\_delete\_event()**

```
bool seq64::perfedit::on_delete_event (
    GdkEventAny * ) [inline], [private]
```

**13.67.4 Friends And Related Function Documentation****13.67.4.1 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).

**13.67.5 Field Documentation****13.67.5.1 m\_peer\_perfedit**

```
perfedit* seq64::perfedit::m_peer_perfedit [private]
```

### 13.67.5.2 m\_table

Gtk::Table\* seq64::perfedit::m\_table [private]

Layout table for song editor.

### 13.67.5.3 m\_vadjust

Gtk::Adjustment\* seq64::perfedit::m\_vadjust [private]

### 13.67.5.4 m\_hadjust

Gtk::Adjustment\* seq64::perfedit::m\_hadjust [private]

### 13.67.5.5 m\_vscroll

Gtk::VScrollbar\* seq64::perfedit::m\_vscroll [private]

### 13.67.5.6 m\_hscroll

Gtk::HScrollbar\* seq64::perfedit::m\_hscroll [private]

### 13.67.5.7 m\_perfnames

perfnames\* seq64::perfedit::m\_perfnames [private]

### 13.67.5.8 m\_perfroll

perfroll\* seq64::perfedit::m\_perfroll [private]

### 13.67.5.9 m\_perftime

perftime\* seq64::perfedit::m\_perftime [private]

#### 13.67.5.10 m\_menu\_snap

Gtk::Menu\* seq64::perfedit::m\_menu\_snap [private]

#### 13.67.5.11 m\_menu\_xpose

Gtk::Menu\* seq64::perfedit::m\_menu\_xpose [private]

#### 13.67.5.12 m\_button\_xpose

Gtk::Button\* seq64::perfedit::m\_button\_xpose [private]

#### 13.67.5.13 m\_entry\_xpose

Gtk::Entry\* seq64::perfedit::m\_entry\_xpose [private]

#### 13.67.5.14 m\_image\_play

Gtk::Image\* seq64::perfedit::m\_image\_play [private]

#### 13.67.5.15 m\_button\_snap

Gtk::Button\* seq64::perfedit::m\_button\_snap [private]

#### 13.67.5.16 m\_entry\_snap

Gtk::Entry\* seq64::perfedit::m\_entry\_snap [private]

#### 13.67.5.17 m\_button\_stop

Gtk::Button\* seq64::perfedit::m\_button\_stop [private]



#### 13.67.5.18 m\_button\_play

Gtk::Button\* seq64::perfedit::m\_button\_play [private]

The Play button object.

#### 13.67.5.19 m\_button\_loop

Gtk::ToggleButton\* seq64::perfedit::m\_button\_loop [private]

#### 13.67.5.20 m\_button\_expand

Gtk::Button\* seq64::perfedit::m\_button\_expand [private]

#### 13.67.5.21 m\_button\_collapse

Gtk::Button\* seq64::perfedit::m\_button\_collapse [private]

#### 13.67.5.22 m\_button\_copy

Gtk::Button\* seq64::perfedit::m\_button\_copy [private]

#### 13.67.5.23 m\_button\_grow

Gtk::Button\* seq64::perfedit::m\_button\_grow [private]

#### 13.67.5.24 m\_button\_undo

Gtk::Button\* seq64::perfedit::m\_button\_undo [private]

#### 13.67.5.25 m\_button\_redo

Gtk::Button\* seq64::perfedit::m\_button\_redo [private]

**13.67.5.26 m\_button\_jack**

Gtk::ToggleButton\* seq64::perfedit::m\_button\_jack [private]

**13.67.5.27 m\_button\_follow**

Gtk::ToggleButton\* seq64::perfedit::m\_button\_follow [private]

**13.67.5.28 m\_button\_bpm**

Gtk::Button\* seq64::perfedit::m\_button\_bpm [private]

**13.67.5.29 m\_entry\_bpm**

Gtk::Entry\* seq64::perfedit::m\_entry\_bpm [private]

**13.67.5.30 m\_button\_bw**

Gtk::Button\* seq64::perfedit::m\_button\_bw [private]

**13.67.5.31 m\_entry\_bw**

Gtk::Entry\* seq64::perfedit::m\_entry\_bw [private]

**13.67.5.32 m\_hbox**

Gtk::HBox\* seq64::perfedit::m\_hbox [private]

**13.67.5.33 m\_hlbox**

Gtk::HBox\* seq64::perfedit::m\_hlbox [private]

**13.67.5.34 m\_tooltips**

```
Gtk::Tooltips* seq64::perfedit::m_tooltips [private]
```

**13.67.5.35 m\_menu\_bpm**

```
Gtk::Menu* seq64::perfedit::m_menu_bpm [private]
```

Drop-down menu for beats/minute.

**13.67.5.36 m\_menu\_bw**

```
Gtk::Menu* seq64::perfedit::m_menu_bw [private]
```

**13.67.5.37 m\_snap**

```
int seq64::perfedit::m_snap [private]
```

**13.67.5.38 m\_bpm**

```
int seq64::perfedit::m_bpm [private]
```

Do not confuse it with BPM (beats per minute). The numerator of the time signature.

**13.67.5.39 m\_bw**

```
int seq64::perfedit::m_bw [private]
```

The denominator of the time signature.

**13.67.5.40 m\_ppqn**

```
int seq64::perfedit::m_ppqn [private]
```

**13.67.5.41 m\_is\_running**

```
bool seq64::perfedit::m_is_running [private]
```

## 13.67.5.42 m\_standard\_bpm

```
int seq64::perfedit::m_standard_bpm [private]
```

## 13.68 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 [perfedit](#), so that it can forward it to any other [perfedit](#) 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

- [perfedit](#) & [m\\_parent](#)

*Provides a link to the [perfedit](#) 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

### 13.68.1 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).

### 13.68.2 Constructor & Destructor Documentation

#### 13.68.2.1 [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.

**13.68.2.2** ~perfnames()

```
virtual seq64::perfnames::~~perfnames ( ) [inline], [virtual]
```

**13.68.3 Member Function Documentation****13.68.3.1** redraw\_dirty\_sequences()

```
void seq64::perfnames::redraw_dirty_sequences ( )
```

**13.68.3.2** 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.

**13.68.3.3** convert\_y()

```
int seq64::perfnames::convert_y (
    int y ) [private]
```

Used in figuring out which sequence to mute/unmute in the performance editor.

**Parameters**

<b>y</b>	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.

**13.68.3.4** draw\_sequences()

```
void seq64::perfnames::draw_sequences ( ) [private]
```

### 13.68.3.5 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.
---------------	--

### 13.68.3.6 change\_vert()

```
void seq64::perfnames::change_vert ( ) [private]
```

### 13.68.3.7 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](#).

### 13.68.3.8 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.



### 13.68.3.9 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.

### 13.68.3.10 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.

### 13.68.3.11 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.

**13.68.3.12 on\_size\_allocate()**

```
void seq64::perfnames::on_size_allocate (
    Gtk::Allocation & a ) [private]
```

It first calls the base-class version of this function.

**Parameters**

<b>a</b>	The allocation event. It is passed to the base-class <a href="#">on_size_allocate()</a> function, and then m_window_x and m_window_y are set to the width and height, respectively, of the allocation.
----------	--

**13.68.3.13 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**

<b>ev</b>	The scrolling event.
-----------	----------------------

**Returns**

Always returns true.

**13.68.4 Friends And Related Function Documentation****13.68.4.1 perfedit**

```
friend class perfedit [friend]
```

**13.68.5 Field Documentation**

#### 13.68.5.1 m\_parent

```
perfedited 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.

#### 13.68.5.2 m\_names\_chars

```
int seq64::perfnames::m_names_chars [private]
```

Pretty much hardwired to 24 at present.

#### 13.68.5.3 m\_char\_w

```
int seq64::perfnames::m_char_w [private]
```

This value is obtained from a font-renderer accessor function.

#### 13.68.5.4 m\_setbox\_w

```
int seq64::perfnames::m_setbox_w [private]
```

This used to be hardwired to  $6 * 2$  (character-width times two).

#### 13.68.5.5 m\_namebox\_w

```
int seq64::perfnames::m_namebox_w [private]
```

This used to be a weird calculation based on character width.

#### 13.68.5.6 m\_names\_x

```
int seq64::perfnames::m_names_x [private]
```

#### 13.68.5.7 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.

**13.68.5.8 m\_xy\_offset**

```
int seq64::perfnames::m_xy_offset [private]
```

Currently hardwired.

**13.68.5.9 m\_seqs\_in\_set**

```
const int seq64::perfnames::m_seqs_in_set [private]
```

**13.68.5.10 m\_sequence\_max**

```
const int seq64::perfnames::m_sequence_max [private]
```

**13.68.5.11 m\_sequence\_offset**

```
int seq64::perfnames::m_sequence_offset [private]
```

**13.68.5.12 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.

**13.69 seq64::perform Class Reference**

This class supports the performance mode.

**Public Types**

- 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*
- 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\\_max](#) () const  
*'Getter' function for member m\_sequence\_max*
- bool [is\\_control\\_status](#) () const  
*'Getter' function for member m\_control\_status*
- 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](#) (midibpm 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*
- 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_running` Could also be called "`is_playing()`".*
- bool `is_pattern_playing` () const

*'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` \*pfcb)

*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()`*
- 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 perfroll.*
- 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 `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 `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 `finish` ()
  - *The rough opposite of `launch()`; it doesn't stop the threads.*
- `midipulse` `get_tick` () const
  - *'Getter' function for member m\_tick*
- void `set_tick` (`midipulse` tick)
  - *'Setter' function for member m\_tick*
- `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.*
- 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 [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, if it is active.*
- 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](#) ()
- 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.*
- void [set\\_offset](#) (int offset)  
*Calculates the offset into the screen sets.*
- int [get\\_offset](#) () const  
*'Getter' function for member m\_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.*
- std::string [key\\_name](#) (unsigned int 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.*



- [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 int [lookup\\_keyevent\\_key](#) (int seqnum)  
*Gets the event key for the given sequence.*
- long [lookup\\_keyevent\\_seq](#) (unsigned int keycode)  
*Gets the sequence number for the given event key.*
- unsigned int [lookup\\_keygroup\\_key](#) (long groupnum)  
*Gets the group key for the given sequence.*
- long [lookup\\_keygroup\\_group](#) (unsigned int 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](#) ()  
*Encapsulates some calls used in mainwnd.*
- int [increment\\_screenset](#) ()  
*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.*
- `void set_input_bus` (int 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` (int 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 perfroll_key_event` (const `keystroke` &k, int drop\_sequence)  
*Provided for perfroll :: on\_key\_press\_event() and perfroll :: 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 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 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.*
- `void set_screenset` (int ss)  
*Sets the `m_screenset` value (the index or ID of the current screen set).*
- `int get_screenset` () const  
*'Getter' function for member `m_screenset`*
- `int get_playing_screenset` () const

- *'Getter' function for member m\_playing\_screen*
- 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.*

## Private Member Functions

- 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*
- void [split\\_trigger](#) (int seqnum, [midipulse](#) tick)  
*Convenience function for perfroll's split-trigger functionality.*
- [midipulse](#) [get\\_max\\_trigger](#) ()  
*Locates the largest trigger value among the active sequences.*
- 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[].*
- void [midi\\_control\\_event](#) (const [event](#) &ev)  
*This function encapsulates code in [input\\_func\(\)](#) to make it easier to read and understand.*
- void [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)  
*Provides operation of the new MIDI controls.*
- const std::string & [get\\_screen\\_set\\_notepad](#) (int screen\_set) const  
*Retrieves the given string from m\_screen\_set\_notepad[].*
- const std::string & [current\\_screen\\_set\\_notepad](#) () const  
*Returns the notepad text for the current screen-set.*
- void [set\\_screen\\_set\\_notepad](#) (int screenset, const std::string &note)  
*Copies the given string into m\_screen\_set\_notepad[].*
- void [set\\_screen\\_set\\_notepad](#) (const std::string &note)  
*Sets the notepad text for the current screen-set.*
- void [set\\_playing\\_screenset](#) ()  
*Sets the screen set that is active, based on the value of m\_screenset.*
- bool [any\\_group\\_unmutes](#) () const

- 'Getter' function for member `m_mute_group[]`
- 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_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 `g_mute`)
  - 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 `is_group_learning` ()
  - 'Getter' function for member `m_mode_group_learn`
- 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 `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 `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 `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.*
- 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 [set\\_looping](#) (bool looping)

*'Setter' function for member m\_looping*
- 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.*
- bool [is\\_screenset\\_valid](#) (int screenset) const

*Checks the screenset against m\_max\_sets.*
- void [set\\_running](#) (bool running)

*'Setter' function for member m\_running*
- void [is\\_pattern\\_playing](#) (bool flag)

*'Setter' function for member m\_is\_pattern\_playing*
- void [set\\_playback\\_mode](#) (bool playbackmode)

*'Setter' function for member m\_playback\_mode*
- int [mute\\_group\\_offset](#) (int track)

*A helper function to calculate the index into the mute-group array, based on the desired track.*
- 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, resets the sequences, and sets m\_usemidiclock false.*
- int [clamp\\_track](#) (int track) const

*Provides common code to keep the track value valid.*
- void [set\\_key\\_event](#) (unsigned int keycode, long sequence\_slot)

*At construction time, this function sets up one keycode and one event slot.*
- void [set\\_key\\_group](#) (unsigned int keycode, long group\_slot)

*At construction time, this function sets up one keycode and one group slot.*
- bool [create\\_master\\_bus](#) ()

*Creates the mastermidibus.*
- 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` (int bus, `clock_e` clocktype)  
*Sets 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` (int bus, bool inputing)  
*Sets a single input item, if in the currently existing range.*
- bool `get_input` (int bus)
- bool `is_input_system_port` (int bus)

## 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_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.*
- bool `m_tracks_mute_state` [c\_seqs\_in\_set]  
*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.*
- int `m_playing_screen`  
*Playing screen support.*
- int `m_playscreen_offset`  
*Playing screen sequence number offset.*
- `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.*
- 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\\_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\\_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\\_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.*
- 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\\_screen\\_set\\_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\\_offset](#)  
*Holds the current offset into the screen-sets.*
- [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\\_seqs\\_in\\_set](#)  
*We will eventually replace `c_seqs_in_set` with this member, which defaults to the value of `c_seqs_in_set`.*
- [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](#)
- `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 [optionsfile](#)
- class [options](#)
- class [perfedit](#)
- class [perfroll](#)
- 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)  
*Global functions for JACK support and JACK sessions.*
- int [jack\\_transport\\_callback](#) (jack\_nframes\_t nframes, void \*arg)
- 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)

### 13.69.1 Detailed Description

It has way too many data members, one of them public. Might be ripe for refactoring. That has its own dangers, of course.

## 13.69.2 Member Enumeration Documentation

### 13.69.2.1 mute\_op\_t

enum seq64::perform::mute\_op\_t

## Enumerator

MUTE_TOGGLE	
MUTE_OFF	
MUTE_ON	

## 13.69.2.2 ff\_rw\_button\_t

```
enum seq64::perform::ff_rw_button_t
```

## Enumerator

FF_RW_REWIND	
FF_RW_NONE	
FF_RW_FORWARD	

## 13.69.3 Constructor &amp; Destructor Documentation

## 13.69.3.1 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 <a href="#">gui_assistant_gtk2</a> class in the seq_gtkmm2 GUI-specific library. Note that we access the m_gui_support member using the <a href="#">gui()</a> accessor function.
<i>ppqn</i>	The default, choosable, or actual PPQN value.

### 13.69.3.2 ~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!

## 13.69.4 Member Function Documentation

### 13.69.4.1 is\_modified() [1/2]

```
bool seq64::perform::is_modified ( ) const [inline]
```

### 13.69.4.2 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.

### 13.69.4.3 ppqn()

```
int seq64::perform::ppqn ( ) const [inline]
```

### 13.69.4.4 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.

### 13.69.4.5 sequence\_max()

```
int seq64::perform::sequence_max ( ) const [inline]
```

### 13.69.4.6 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.

### 13.69.4.7 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 <a href="#">unset_edit_sequence()</a> to disable it if it matches the current edit-sequence number.
---------------	---

## 13.69.4.8 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.
---------------	---

## 13.69.4.9 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.
---------------	---

## 13.69.4.10 get\_beats\_per\_bar()

```
int seq64::perform::get_beats_per_bar ( ) const [inline]
```

## 13.69.4.11 set\_beats\_per\_bar()

```
void seq64::perform::set_beats_per_bar (
    midibpm bpm ) [inline]
```

## Parameters

<i>bpm</i>	Provides the value for beats/measure. Also used to set the beats/measure in the JACK assistant object.
------------	--

**13.69.4.12 get\_beat\_width()**

```
int seq64::perform::get_beat_width ( ) const [inline]
```

**13.69.4.13 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.
-----------	--

**13.69.4.14 clocks\_per\_metronome()** [1/2]

```
void seq64::perform::clocks_per_metronome (
    int cpm ) [inline]
```

**13.69.4.15 clocks\_per\_metronome()** [2/2]

```
int seq64::perform::clocks_per_metronome ( ) const [inline]
```

**13.69.4.16 set\_32nds\_per\_quarter()**

```
void seq64::perform::set_32nds_per_quarter (
    int tpq ) [inline]
```

**13.69.4.17 get\_32nds\_per\_quarter()**

```
int seq64::perform::get_32nds_per_quarter ( ) const [inline]
```

**13.69.4.18 us\_per\_quarter\_note()** [1/2]

```
void seq64::perform::us_per_quarter_note (
    long upqn ) [inline]
```

**13.69.4.19 us\_per\_quarter\_note()** [2/2]

```
long seq64::perform::us_per_quarter_note ( ) const [inline]
```

**13.69.4.20 gui()** [1/2]

```
const gui_assistant& seq64::perform::gui ( ) const [inline]
```

**13.69.4.21 gui()** [2/2]

```
gui_assistant& seq64::perform::gui ( ) [inline]
```

**13.69.4.22 keys()** [1/2]

```
const keys_perform& seq64::perform::keys ( ) const [inline]
```

**13.69.4.23 keys()** [2/2]

```
keys_perform& seq64::perform::keys ( ) [inline]
```

**13.69.4.24 master\_bus()**

```
mastermidibus& seq64::perform::master_bus ( ) [inline]
```

**13.69.4.25 filter\_by\_channel()**

```
void seq64::perform::filter_by_channel (
    bool flag ) [inline]
```

**13.69.4.26 is\_running()**

```
bool seq64::perform::is_running ( ) const [inline]
```

**13.69.4.27 is\_pattern\_playing()** [1/2]

```
bool seq64::perform::is_pattern_playing ( ) const [inline]
```

**13.69.4.28 toggle\_song\_start\_mode()**

```
bool seq64::perform::toggle_song_start_mode ( ) [inline]
```

**13.69.4.29 song\_start\_mode()** [1/2]

```
void seq64::perform::song_start_mode (
    bool flag ) [inline]
```

**13.69.4.30 song\_start\_mode()** [2/2]

```
bool seq64::perform::song_start_mode ( ) const [inline]
```

**13.69.4.31 is\_jack\_running()**

```
bool seq64::perform::is_jack_running ( ) const [inline]
```

**13.69.4.32 is\_jack\_master()**

```
bool seq64::perform::is_jack_master ( ) const [inline]
```

**13.69.4.33 enregister()**

```
void seq64::perform::enregister (
    performcallback * pfc ) [inline]
```



## Parameters

<i>pcb</i>	Provides the pointer to the performance callback.
------------	---

## 13.69.4.34 toggle\_jack\_mode()

```
void seq64::perform::toggle_jack_mode ( ) [inline]
```

## 13.69.4.35 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.

## 13.69.4.36 get\_toggle\_jack()

```
bool seq64::perform::get_toggle_jack ( ) const [inline]
```

## 13.69.4.37 set\_jack\_stop\_tick()

```
void seq64::perform::set_jack_stop_tick (
    midipulse tick ) [inline]
```

**13.69.4.38 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).

**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.

**13.69.4.39 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.

**13.69.4.40 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.

**13.69.4.41 start\_from\_perfedit()** [1/2]

```
void seq64::perform::start_from_perfedit (
    bool flag ) [inline]
```

**13.69.4.42 start\_from\_perfedit()** [2/2]

```
bool seq64::perform::start_from_perfedit ( ) const [inline]
```

**13.69.4.43 set\_follow\_transport()**

```
void seq64::perform::set_follow_transport (
    bool flag ) [inline]
```

**13.69.4.44 get\_follow\_transport()**

```
bool seq64::perform::get_follow_transport ( ) const [inline]
```

**13.69.4.45 toggle\_follow\_transport()**

```
void seq64::perform::toggle_follow_transport ( ) [inline]
```

**13.69.4.46 set\_reposition()**

```
void seq64::perform::set_reposition (
    bool postype = true ) [inline]
```

**13.69.4.47 ff\_rw\_type()** [1/2]

```
ff_rw_button_t seq64::perform::ff_rw_type ( ) [inline]
```

**13.69.4.48 ff\_rw\_type()** [2/2]

```
void seq64::perform::ff_rw_type (
    ff_rw_button_t button_type ) [inline]
```

**13.69.4.49 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.
--------------	--

**13.69.4.50 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.
--------------	---

**13.69.4.51 reposition()**

```
void seq64::perform::reposition (
    midipulse tick )
```

Used only in perfroll :: on\_key\_press\_event() to implement the Seq32 pointer-position feature.

**Parameters**

<i>tick</i>	Provides the position value to be set.
-------------	--

**13.69.4.52 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.

**13.69.4.53 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.

**13.69.4.54 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.
------------	--

**13.69.4.55 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.

#### 13.69.4.56 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.
------------	---

#### 13.69.4.57 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.

**13.69.4.58 clear\_sequence\_triggers()**

```
void seq64::perform::clear_sequence_triggers (
    int seq )
```

**Parameters**

<code>seq</code>	Provides the desired sequence. The <a href="#">is_active()</a> function validates this value.
------------------	---

**13.69.4.59 print\_triggers()**

```
void seq64::perform::print_triggers ( ) const
```

**13.69.4.60 finish()**

```
void seq64::perform::finish ( ) [inline]
```

A minor simplification for the `main()` routine, hides the JACK support macro.

**13.69.4.61 get\_tick()**

```
midipulse seq64::perform::get_tick ( ) const [inline]
```

**13.69.4.62 set\_tick()**

```
void seq64::perform::set_tick (
    midipulse tick ) [inline]
```

**13.69.4.63 get\_jack\_tick()**

```
midipulse seq64::perform::get_jack_tick ( ) const [inline]
```

**13.69.4.64 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.
-------------	--

13.69.4.65 `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 `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.

13.69.4.66 `get_left_tick()`

```
midipulse seq64::perform::get_left_tick ( ) const [inline]
```

13.69.4.67 `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.
-------------	---

13.69.4.68 `get_start_tick()`

```
midipulse seq64::perform::get_start_tick ( ) const [inline]
```



**13.69.4.69 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.

**13.69.4.70 get\_right\_tick()**

```
midipulse seq64::perform::get_right_tick ( ) const [inline]
```

**13.69.4.71 left\_right\_size()**

```
double seq64::perform::left_right_size ( ) const [inline]
```

**Returns**

Returns the difference between the right and left tick, cast to double.

**13.69.4.72 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.

**13.69.4.73 apply\_song\_transpose()**

```
void seq64::perform::apply_song_transpose ( )
```

**13.69.4.74 set\_transpose()**

```
void seq64::perform::set_transpose (
    int t ) [inline]
```

**13.69.4.75 get\_transpose()**

```
int seq64::perform::get_transpose ( ) const [inline]
```

**13.69.4.76 get\_beats\_per\_minute()**

```
midibpm seq64::perform::get_beats_per_minute ( ) [inline]
```

**Returns**

Returns the value of beats/minute from the master buss.

**13.69.4.77 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.
---------------	------------------------

**13.69.4.78 unset\_sequence\_control\_status()**

```
void seq64::perform::unset_sequence_control_status (
    int status )
```

Then the given status is reversed in `m_control_status`.

## Parameters

<i>status</i>	The status to be used.
---------------	------------------------

13.69.4.79 `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 is offset by the current screen-set number, `m_offset` before passing it to this function.

## Parameters

<i>seq</i>	The sequence number of the sequence to be potentially toggled. This value must be a valid and active sequence number.
------------	---

13.69.4.80 `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\(\)](#).

## 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.

**13.69.4.81 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.
------------	---

**13.69.4.82 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.
------------	--

**13.69.4.83 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. HMMMMMM.

**Parameters**

<i>flag</i>	If true (the default), the song-mute of the sequence is turned on. Otherwise, it is turned off.
-------------	---

**13.69.4.84 toggle\_all\_tracks()**

```
void seq64::perform::toggle_all_tracks ( )
```

Covers tracks from 0 to m\_sequence\_max.

**13.69.4.85 armed\_saved()**

```
bool seq64::perform::armed_saved ( ) const [inline]
```

**13.69.4.86 toggle\_playing\_tracks()**

```
void seq64::perform::toggle_playing_tracks ( )
```

**13.69.4.87 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.

**13.69.4.88 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.

**13.69.4.89 input\_func()**

```
void seq64::perform::input_func ( )
```

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.

**13.69.4.90 set\_group\_mute\_state()**

```
void seq64::perform::set_group_mute_state (
    int gtrack,
    bool muted ) [inline]
```

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 midifile and optionsfile when parsing the file to get the initial mute-groups.

**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.

**13.69.4.91 get\_group\_mute\_state()**

```
bool seq64::perform::get_group_mute_state (
    int gtrack ) [inline]
```

Uses the [mute\\_group\\_offset\(\)](#) function. This function is used in midifile and optionsfile when writing the file to get the initial mute-groups.

**Parameters**

<i>gtrack</i>	The number of the track for which the state is to be obtained. Like <a href="#">set_group_mute_state()</a> , this value is offset by adding <code>m_mute_group_selected * m_seqs_in_set</code> .
---------------	--

**Returns**

Returns the desired `m_mute_group[]` value.

**13.69.4.92 set\_offset()**

```
void seq64::perform::set_offset (
    int offset ) [inline]
```

Sets `m_offset = offset * c_mainwnd_rows * c_mainwnd_cols`.

**Parameters**

<i>offset</i>	The desired offset.
---------------	---------------------

**13.69.4.93 get\_offset()**

```
int seq64::perform::get_offset ( ) const [inline]
```

**13.69.4.94 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`).

**13.69.4.95 restore\_playing\_state()**

```
void seq64::perform::restore_playing_state ( )
```

Used in unsetting the snapshot status (`c_status_snapshot`).

**13.69.4.96 key\_name()**

```
std::string seq64::perform::key_name (
    unsigned int k ) const [inline]
```

**Parameters**

<i>k</i>	The key number for which to return the string name of the key.
----------	--



**13.69.4.97 get\_key\_events()**

```
keys_perform::SlotMap& seq64::perform::get_key_events ( ) [inline]
```

**13.69.4.98 get\_key\_groups()**

```
keys_perform::SlotMap& seq64::perform::get_key_groups ( ) [inline]
```

**13.69.4.99 get\_key\_events\_rev()**

```
keys_perform::RevSlotMap& seq64::perform::get_key_events_rev ( ) [inline]
```

**13.69.4.100 get\_key\_groups\_rev()**

```
keys_perform::RevSlotMap& seq64::perform::get_key_groups_rev ( ) [inline]
```

**13.69.4.101 show\_ui\_sequence\_key()** [1/2]

```
bool seq64::perform::show_ui_sequence_key ( ) const [inline]
```

Used in mainwid, options, optionsfile, userfile, and perform.

**13.69.4.102 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 <a href="#">keys().show_ui_sequence_key()</a> .
-------------	---

**13.69.4.103 show\_ui\_sequence\_number()** [1/2]

```
bool seq64::perform::show_ui_sequence_number ( ) const [inline]
```

Used in mainwid, optionsfile, and perform.

**13.69.4.104 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 <a href="#">keys().show_ui_sequence_number()</a> .
-------------	---

**13.69.4.105 lookup\_keyevent\_key()**

```
unsigned int 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.

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.

**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 period ('?') character is returned.

**13.69.4.106 lookup\_keyevent\_seq()**

```
long seq64::perform::lookup_keyevent_seq (
    unsigned int 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.

**13.69.4.107 lookup\_keygroup\_key()**

```
unsigned int seq64::perform::lookup_keygroup_key (  
    long groupnum ) [inline]
```

**Parameters**

<i>groupnum</i>	The number of the sequence for which to return the group key.
-----------------	---

**Returns**

Returns the desired key. If there is no such value, then the period ('.') character is returned.

**13.69.4.108 lookup\_keygroup\_group()**

```
long seq64::perform::lookup_keygroup_group (  
    unsigned int 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.

**13.69.4.109 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).
-----------------	--

#### 13.69.4.110 `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);
```

#### 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 <code>start_playing()</code> function.
-----------------	--

**13.69.4.111 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).

**13.69.4.112 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).
-----------------	--

**13.69.4.113 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).
-----------------	--

**13.69.4.114 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.

**13.69.4.115 learn\_toggle()**

```
void seq64::perform::learn_toggle ( ) [inline]
```

**13.69.4.116 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.

**13.69.4.117 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.

**13.69.4.118 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.

**13.69.4.119 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.

**13.69.4.120 decrement\_screenset()**

```
int seq64::perform::decrement_screenset ( ) [inline]
```

**13.69.4.121 increment\_screenset()**

```
int seq64::perform::increment_screenset ( ) [inline]
```

**13.69.4.122 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**

<i>seq</i>	Provides a reference to the desired sequence.
------------	---

**13.69.4.123 is\_smf\_0()**

```
bool seq64::perform::is_smf_0 (
    const sequence & seq ) const [inline]
```

**Parameters**

<i>seq</i>	Provides a reference to the desired sequence.
------------	---

**13.69.4.124 get\_sequence()** [1/2]

```
const sequence* seq64::perform::get_sequence (
    int seq ) const [inline]
```

This is the const version.

**Parameters**

<i>seq</i>	The prospective sequence number.
------------	----------------------------------

**Returns**

Returns the value of `m_seqs[seq]` if `seq` is valid. Otherwise, a null pointer is returned.

**13.69.4.125 `get_sequence()` [2/2]**

```
sequence* seq64::perform::get_sequence (
    int seq ) [inline]
```

**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.

**13.69.4.126 `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.
------------------	---

**13.69.4.127 `sequence_label()`**

```
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.



**Note**

Later, we could add the sequence hot-key to this string, though showing that is not much use in perfnames. 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.

**13.69.4.128 set\_input\_bus()**

```
void seq64::perform::set_input_bus (
    int 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.

**13.69.4.129 set\_clock\_bus()**

```
void seq64::perform::set_clock_bus (
    int 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, and e_clock_mod.

**13.69.4.130 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`: bpm up & down; screenset up & down.

**Parameters**

<i>k</i>	The keystroke object to be handled.
----------	-------------------------------------

**Returns**

Returns true if the key was handled.

**13.69.4.131 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.

**13.69.4.132 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.

#### 13.69.4.133 move\_triggers()

```
void seq64::perform::move_triggers (
    bool direction )
```

#### Parameters

<i>direction</i>	Specifies the desired direction; false = left, true = right.
------------------	--

#### 13.69.4.134 copy\_triggers()

```
void seq64::perform::copy_triggers ( )
```

This copies the triggers between the L marker and R marker to the R marker.

#### 13.69.4.135 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.
--------------	--

13.69.4.136 `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.

13.69.4.137 `pop_trigger_redo()`

```
void seq64::perform::pop_trigger_redo ( )
```

13.69.4.138 `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.

13.69.4.139 `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.

**13.69.4.140 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.

**13.69.4.141 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.

**13.69.4.142 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.

**13.69.4.143 set\_screenset()**

```
void 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.

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.

**Parameters**

<i>ss</i>	The index of the desired new screen set. It is forced to range from 0 to <code>m_max_sets - 1</code> . The clamping seems weird, but hews to <code>seq24</code> . What it does is let the user wrap around the screen-sets in the user interface.
-----------	---

**13.69.4.144 get\_screenset()**

```
int seq64::perform::get_screenset ( ) const [inline]
```

**13.69.4.145 get\_playing\_screenset()**

```
int seq64::perform::get_playing_screenset ( ) const [inline]
```

**13.69.4.146 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>isshiftkey</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.

**13.69.4.147 toggle\_other\_names()**

```
bool seq64::perform::toggle_other_names (
    int seqnum,
    bool isshiftkey )
```

See [perfnames::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>isshiftkey</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.

**13.69.4.148 have\_undo()**

```
bool seq64::perform::have_undo ( ) const [inline], [private]
```

**13.69.4.149 set\_have\_undo()**

```
void seq64::perform::set_have_undo (
    bool undo ) [inline], [private]
```

Once it is set, it remains set, unless cleared by saving the file.

**13.69.4.150 have\_redo()**

```
bool seq64::perform::have_redo ( ) const [inline], [private]
```

**13.69.4.151 set\_have\_redo()**

```
void seq64::perform::set_have_redo (
    bool redo ) [inline], [private]
```

**13.69.4.152 split\_trigger()**

```
void seq64::perform::split_trigger (
    int seqnum,
    midipulse tick ) [private]
```

**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.

**13.69.4.153 get\_max\_trigger()**

```
midipulse seq64::perform::get_max_trigger ( ) [private]
```

**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?

**13.69.4.154 collapse()**

```
void seq64::perform::collapse ( ) [inline], [private]
```

**13.69.4.155 copy()**

```
void seq64::perform::copy ( ) [inline], [private]
```

**13.69.4.156 expand()**

```
void seq64::perform::expand ( ) [inline], [private]
```



**13.69.4.157 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 <a href="#">valid_midi_control_seq()</a> to obtain a control value (such as <code>c_midi_control_bpm_up</code> ) to use to retrieve the desired <a href="#">midi_control</a> object.
------------	--

**Returns**

Returns the "toggle" value if the control value is valid, or a reference to `sm_mc_dummy` otherwise.

**13.69.4.158 midi\_control\_on()**

```
midi_control & seq64::perform::midi_control_on (
    int ctl ) [private]
```

**Parameters**

<i>ctl</i>	Provides the index to pass to <a href="#">valid_midi_control_seq()</a> to obtain a control value (such as <code>c_midi_control_bpm_up</code> ) to use to retrieve the desired <a href="#">midi_control</a> object.
------------	--

**Returns**

Returns the "on" value if the control value is valid, and a reference to `sm_mc_dummy` otherwise.

**13.69.4.159 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 <a href="#">midi_control</a> object.
------------	--

**Returns**

Returns the "off" value if the control value is valid, and a reference to sm\_mc\_dummy otherwise.

**13.69.4.160 midi\_control\_event()**

```
void seq64::perform::midi_control_event (
    const event & ev ) [private]
```

Here is the processing involved in this function ....

Incorporates pull request #24, arnaud-jacquemin, issue #23 "MIDI controller toggles wrong pattern".

*Change Note* ca 2016-10-05 Issue #35. Changed "on" to "off".

**QUESTIONS/TODO:**

1. Why go above the sequence numbers, why not just go up to c\_midi\_track\_ctrl?
2. What about our new extended controls?

**13.69.4.161 handle\_midi\_control()**

```
void 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_record
c_midi_control_solo          (for toggle, on, or off)
c_midi_control_thru
c_midi_control_bpm_page_up
c_midi_control_bpm_page_dn
c_midi_control_16 to _19    (reserved for expansion)
```

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:

## 13.69.4.162 handle\_midi\_control\_ex()

```
bool seq64::perform::handle_midi_control_ex (
    int ctl,
    midi_control::action a ) [private]
```

## Parameters

<i>ctl</i>	The MIDI control value to use to perform an operation.
<i>a</i>	The action of the control.

## Returns

Returns true if the control was an extended control and was acted on.

## 13.69.4.163 get\_screen\_set\_notepad()

```
const std::string & seq64::perform::get_screen_set_notepad (
    int screenset ) const [private]
```

## Parameters

<i>screenset</i>	The ID number of the screen set, an index into the m_screen_set_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.

## 13.69.4.164 current\_screen\_set\_notepad()

```
const std::string& seq64::perform::current_screen_set_notepad ( ) const [inline], [private]
```

**13.69.4.165 set\_screen\_set\_notepad()** [1/2]

```
void seq64::perform::set_screen_set_notepad (
    int screenset,
    const std::string & notepad ) [private]
```

**Parameters**

<i>screenset</i>	The ID number of the screen set, an index into the m_screen_set_notepad[] 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.

**13.69.4.166 set\_screen\_set\_notepad()** [2/2]

```
void seq64::perform::set_screen_set_notepad (
    const std::string & note ) [inline], [private]
```

**Parameters**

<i>note</i>	The string value to set into the notepad text.
-------------	--

**13.69.4.167 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\_playing↵\_screen) 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\_playing\_screen, 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.

**13.69.4.168 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.

## 13.69.4.169 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_playing_screen`), and its `m_track_mute_state[]` is true, then the sequence is turned on, otherwise it is turned off.

*Change Note* tdeagan 2015-12-22 via git pull. Replaced `m_playing_screen` with `m_screenset`.

It seems to us that the for (g) clause should have g range from 0 to `m_max_sets`, not `m_seqs_in_set`.

## 13.69.4.170 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.
--------------	--

## 13.69.4.171 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.

## Parameters

<i>op</i>	Provides the "flag" that indicates if this function is to set mute on, off, or to toggle the mute status.
-----------	---

## 13.69.4.172 set\_mode\_group\_mute()

```
void seq64::perform::set_mode_group_mute ( ) [inline], [private]
```

**13.69.4.173** `unset_mode_group_mute()`

```
void seq64::perform::unset_mode_group_mute ( ) [inline], [private]
```

**13.69.4.174** `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.

**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.
------------------	--

**13.69.4.175** `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.

**13.69.4.176** `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.

**13.69.4.177** `is_group_learning()`

```
bool seq64::perform::is_group_learning ( ) [inline], [private]
```

**13.69.4.178 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>. ↩  
git mute\_groups m\_screenset replaces m\_playscreen\_offset.

**Parameters**

<code>mutegroup</code>	Provides the mute-group to select.
------------------------	------------------------------------

**13.69.4.179 activate()**

```
bool seq64::perform::activate ( ) [private]
```

Currently does work only for JACK; the `activate()` calls for other APIs just return true without doing anything.

**13.69.4.180 start()**

```
void seq64::perform::start (
    bool songmode ) [private]
```

**Parameters**

<code>songmode</code>	If true, playback is to be in Song mode. Otherwise, it is to be in Live mode.
-----------------------	---

**13.69.4.181 stop()**

```
void seq64::perform::stop ( ) [private]
```

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!!!

#### 13.69.4.182 `start_jack()`

```
void seq64::perform::start_jack ( ) [inline], [private]
```

#### 13.69.4.183 `stop_jack()`

```
void seq64::perform::stop_jack ( ) [inline], [private]
```

#### 13.69.4.184 `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.



**13.69.4.185 off\_sequences()**

```
void seq64::perform::off_sequences ( ) [private]
```

Replaces "for (int s = 0; s < m\_sequence\_max; ++s)"

**13.69.4.186 all\_notes\_off()**

```
void seq64::perform::all_notes_off ( ) [private]
```

Then flush the master MIDI buss.

**13.69.4.187 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.

**13.69.4.188 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.
------------	---

**13.69.4.189 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.

## Parameters

<i>pause</i>	Try to prevent notes from lingering on pause if true. By default, it is false.
--------------	--

13.69.4.190 `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 "s" (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 <code>m_tick</code> .
-------------	---

13.69.4.191 `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.
-------------	--

13.69.4.192 `set_beats_per_minute()`

```
void seq64::perform::set_beats_per_minute (
    midibpm bpm ) [private]
```

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.
------------	--

## 13.69.4.193 set\_looping()

```
void seq64::perform::set_looping (
    bool looping ) [inline], [private]
```

## Parameters

<i>looping</i>	The boolean value to set for looping, used in the performance editor.
----------------	---

## 13.69.4.194 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.

## 13.69.4.195 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.

## 13.69.4.196 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.

**13.69.4.197 init\_jack\_transport()**

```
bool seq64::perform::init_jack_transport ( ) [inline], [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.

**13.69.4.198 deinit\_jack\_transport()**

```
bool seq64::perform::deinit_jack_transport ( ) [inline], [private]
```

Called by [launch\(\)](#) and in the options module, when the Disconnect button is pressed.

**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.

**13.69.4.199 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.

**13.69.4.200 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.
-------------	---

## 13.69.4.201 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.

## 13.69.4.202 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.

## 13.69.4.203 set\_running()

```
void seq64::perform::set_running (
    bool running ) [inline], [private]
```

## Parameters

<i>running</i>	The value of the running flag to be set.
----------------	--

13.69.4.204 `is_pattern_playing()` [2/2]

```
void seq64::perform::is_pattern_playing (
    bool flag ) [inline], [private]
```

13.69.4.205 `set_playback_mode()`

```
void seq64::perform::set_playback_mode (
    bool playbackmode ) [inline], [private]
```

## Parameters

<i>playbackmode</i>	The value of the playback mode flag to be set.
---------------------	--

13.69.4.206 `mute_group_offset()`

```
int seq64::perform::mute_group_offset (
    int track ) [inline], [private]
```

## Parameters

<i>track</i>	The number of the desired track.
--------------	----------------------------------

13.69.4.207 `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 [0, m_sequence_max).
------------	--

## Returns

Returns true if the sequence number is valid.

## 13.69.4.208 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 <a href="#">sequence_count()</a> , 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.

## 13.69.4.209 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.

**13.69.4.210 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.
-----------------	---

**13.69.4.211 inner\_stop()**

```
void seq64::perform::inner_stop (
    bool midiclock = false ) [private]
```

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.
------------------	--

**13.69.4.212 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.

#### 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.

#### 13.69.4.213 set\_key\_event()

```
void seq64::perform::set_key_event (
    unsigned int keycode,
    long 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.

#### 13.69.4.214 set\_key\_group()

```
void seq64::perform::set_key_group (
    unsigned int keycode,
    long 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.

#### 13.69.4.215 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::port_settings()` 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.

#### Returns

Returns true if the creation succeeded.

#### 13.69.4.216 `add_clock()`

```
void seq64::perform::add_clock (
    clock_e clocktype ) [inline], [private]
```

#### Parameters

<i>clocktype</i>	The clock value read from the "rc" file.
------------------	--

#### 13.69.4.217 `set_clock()`

```
void seq64::perform::set_clock (
    int bus,
    clock_e clocktype ) [inline], [private]
```

Mostly meant for use by the Options / MIDI Input tab.

#### 13.69.4.218 `add_input()`

```
void seq64::perform::add_input (
    bool flag ) [inline], [private]
```

#### Parameters

<i>flag</i>	The input flag read from the "rc" file.
-------------	---

#### 13.69.4.219 `set_input()`

```
void seq64::perform::set_input (
```

```
int bus,  
bool inputing ) [inline], [private]
```

Mostly meant for use by the Options / MIDI Input tab.

#### 13.69.4.220 get\_input()

```
bool seq64::perform::get_input (  
int bus ) [inline], [private]
```

#### 13.69.4.221 is\_input\_system\_port()

```
bool seq64::perform::is_input_system_port (  
int bus ) [inline], [private]
```

### 13.69.5 Friends And Related Function Documentation

#### 13.69.5.1 jack\_assistant

```
friend class jack_assistant [friend]
```

#### 13.69.5.2 keybindentry

```
friend class keybindentry [friend]
```

#### 13.69.5.3 mainwnd

```
friend class mainwnd [friend]
```

#### 13.69.5.4 midifile

```
friend class midifile [friend]
```

#### 13.69.5.5 optionsfile

```
friend class optionsfile [friend]
```

#### 13.69.5.6 options

```
friend class options [friend]
```

#### 13.69.5.7 perfedit

```
friend class perfedit [friend]
```

#### 13.69.5.8 perfroll

```
friend class perfroll [friend]
```

#### 13.69.5.9 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 <a href="#">output_func()</a> is called. Currently, this parameter is not validated, for speed.
---------------	--

##### Returns

Always returns nullptr.

#### 13.69.5.10 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 <a href="#">output_func()</a> is called. Currently, this parameter is not validated, for speed.
---------------	--

## Returns

Always returns nullptr.

## 13.69.5.11 jack\_sync\_callback

```
int jack_sync_callback (
    jack_transport_state_t state,
    jack_position_t * pos,
    void * arg ) [friend]
```

This JACK synchronization callback informs the specified perform object of the current state and parameters of JACK.

The transport state will be:

- JackTransportStopped when a new position is requested.
- JackTransportStarting when the transport is waiting to start.
- JackTransportRolling when the timeout has expired, and the position is now a moving target.

This is the slow-sync callback, which the stazed code replaces with [jack\\_transport\\_callback\(\)](#).

## Parameters

<i>state</i>	The JACK Transport state.
<i>pos</i>	The JACK position value.
<i>arg</i>	The pointer to the <a href="#">jack_assistant</a> object. Currently not checked for nullity, nor dynamic-casted.

## Returns

Returns 1 if the function works, and 0 if something was wrong.

## 13.69.5.12 jack\_transport\_callback

```
int jack_transport_callback (
    jack_nframes_t nframes,
    void * arg ) [friend]
```

### 13.69.5.13 jack\_shutdown

```
void jack_shutdown (
    void * arg ) [friend]
```

### 13.69.5.14 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 seconds 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 <a href="#">jack_assistant</a> pointer, currently unchecked for nullity.

## 13.69.5.15 get\_current\_jack\_position

```
long get_current_jack_position (
    void * arg ) [friend]
```

## Warning

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

## 13.69.6 Field Documentation

## 13.69.6.1 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?

## 13.69.6.2 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.

#### 13.69.6.3 m\_start\_from\_perfedit

```
bool seq64::perform::m_start_from_perfedit [private]
```

#### 13.69.6.4 m\_reposition

```
bool seq64::perform::m_reposition [private]
```

#### 13.69.6.5 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.

#### 13.69.6.6 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.

#### 13.69.6.7 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.

#### 13.69.6.8 m\_armed\_saved

```
bool seq64::perform::m_armed_saved [private]
```

#### 13.69.6.9 m\_armed\_statuses

```
bool seq64::perform::m_armed_statuses[c_max_sequence] [private]
```



#### 13.69.6.10 m\_tracks\_mute\_state

```
bool seq64::perform::m_tracks_mute_state[c_seqs_in_set] [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.

#### 13.69.6.11 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.

#### 13.69.6.12 m\_mode\_group\_learn

```
bool seq64::perform::m_mode_group_learn [private]
```

#### 13.69.6.13 m\_mute\_group\_selected

```
int seq64::perform::m_mute_group_selected [private]
```

It seems like a "group" is essentially a "set" that is selected for the saving and restoring of the status of all patterns in that set.

#### 13.69.6.14 m\_playing\_screen

```
int seq64::perform::m_playing_screen [private]
```

In seq24, this value is altered by `set_playing_screenset()`, which is called by `handle_midi_control(c_midi_control↵_play_ss, state)`.

#### 13.69.6.15 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`, which we want to be able to change at run-time, as a future enhancement.

#### 13.69.6.16 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.

**13.69.6.17 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.

**13.69.6.18 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?

**13.69.6.19 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.

**13.69.6.20 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.

**13.69.6.21 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?

**13.69.6.22 m\_sequence\_state**

```
bool seq64::perform::m_sequence_state[c_max_sequence] [private]
```

**13.69.6.23 m\_transpose**

```
int seq64::perform::m_transpose [private]
```

#### 13.69.6.24 m\_out\_thread

```
pthread_t seq64::perform::m_out_thread [private]
```

Provides a "handle" to the output thread.

#### 13.69.6.25 m\_in\_thread

```
pthread_t seq64::perform::m_in_thread [private]
```

#### 13.69.6.26 m\_out\_thread\_launched

```
bool seq64::perform::m_out_thread_launched [private]
```

#### 13.69.6.27 m\_in\_thread\_launched

```
bool seq64::perform::m_in_thread_launched [private]
```

#### 13.69.6.28 m\_running

```
bool seq64::perform::m_running [private]
```

However, this flag is conflated with some JACK support, and we have to supplement it with another flag, `m_↔ pattern_playing`.

#### 13.69.6.29 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.

#### 13.69.6.30 m\_inputting

```
bool seq64::perform::m_inputting [private]
```

### 13.69.6.31 m\_outputing

```
bool seq64::perform::m_outputing [private]
```

### 13.69.6.32 m\_looping

```
bool seq64::perform::m_looping [private]
```

If true, the performance will loop between the L and R markers in the performance editor.

### 13.69.6.33 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
```

### 13.69.6.34 m\_ppqn

```
int seq64::perform::m_ppqn [private]
```

### 13.69.6.35 m\_bpm

```
midibpm seq64::perform::m_bpm [private]
```

### 13.69.6.36 m\_beats\_per\_bar

```
int seq64::perform::m_beats_per_bar [private]
```

The default value is SEQ64\_DEFAULT\_BEATS\_PER\_MEASURE (4).

### 13.69.6.37 m\_beat\_width

```
int seq64::perform::m_beat_width [private]
```

The default value is SEQ64\_DEFAULT\_BEAT\_WIDTH (4).

**13.69.6.38 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.

**13.69.6.39 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.

**13.69.6.40 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.

**13.69.6.41 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.

**13.69.6.42 m\_master\_clocks**

```
std::vector<clock_e> seq64::perform::m_master_clocks [private]
```

**13.69.6.43 m\_master\_inputs**

```
std::vector<bool> seq64::perform::m_master_inputs [private]
```

**13.69.6.44 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.

**13.69.6.45 m\_left\_tick**

```
midipulse seq64::perform::m_left_tick [private]
```

Note that "tick" is actually "pulses".

**13.69.6.46 m\_right\_tick**

```
midipulse seq64::perform::m_right_tick [private]
```

Note that "tick" is actually "pulses".

**13.69.6.47 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".

**13.69.6.48 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.

**13.69.6.49 m\_jack\_tick**

```
midipulse seq64::perform::m_jack_tick [private]
```

**13.69.6.50 m\_usemidiclock**

```
bool seq64::perform::m_usemidiclock [private]
```

**13.69.6.51 m\_midiclockrunning**

```
bool seq64::perform::m_midiclockrunning [private]
```

**13.69.6.52 m\_midiclocktick**

```
int seq64::perform::m_midiclocktick [private]
```

**13.69.6.53 m\_midiclockpos**

```
int seq64::perform::m_midiclockpos [private]
```

**13.69.6.54 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.

**13.69.6.55 m\_screen\_set\_notepad**

```
std::string seq64::perform::m_screen_set_notepad[c_max_sets] [private]
```

**13.69.6.56 m\_midi\_cc\_toggle**

```
midi_control seq64::perform::m_midi_cc_toggle[c_midi_controls_extended] [private]
```

**13.69.6.57 m\_midi\_cc\_on**

```
midi_control seq64::perform::m_midi_cc_on[c_midi_controls_extended] [private]
```

**13.69.6.58 m\_midi\_cc\_off**

```
midi_control seq64::perform::m_midi_cc_off[c_midi_controls_extended] [private]
```

**13.69.6.59 m\_offset**

```
int seq64::perform::m_offset [private]
```

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.

#### 13.69.6.60 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.

#### 13.69.6.61 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.

#### 13.69.6.62 m\_seqs\_in\_set

```
int seq64::perform::m_seqs_in_set [private]
```

This change will require some arrays to be dynamically allocated (vectors).

#### 13.69.6.63 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.

#### 13.69.6.64 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.

#### 13.69.6.65 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?

#### 13.69.6.66 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`. Its 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.



**13.69.6.67 m\_edit\_sequence**

```
int seq64::perform::m_edit_sequence [private]
```

Moving this status from seqmenu into perform for better centralized management.

**13.69.6.68 m\_is\_modified**

```
bool seq64::perform::m_is_modified [private]
```

All the GUIs seem to use a perform object.

**13.69.6.69 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\_running and m\_outputing are false. This variable is also signalled in the perform destructor.

**13.69.6.70 m\_jack\_asst**

```
jack_assistant seq64::perform::m_jack_asst [private]
```

It implements most of the JACK stuff.

**13.69.6.71 m\_have\_undo**

```
bool seq64::perform::m_have_undo [private]
```

**13.69.6.72 m\_undo\_vect**

```
std::vector<int> seq64::perform::m_undo_vect [private]
```

See the [push\\_trigger\\_undo\(\)](#) function.

**13.69.6.73 m\_have\_redo**

```
bool seq64::perform::m_have_redo [private]
```

#### 13.69.6.74 m\_redo\_vect

```
std::vector<int> seq64::perform::m_redo_vect [private]
```

See the [pop\\_trigger\\_undo\(\)](#) function.

#### 13.69.6.75 m\_notify

```
std::vector<performcallback *> seq64::perform::m_notify [private]
```

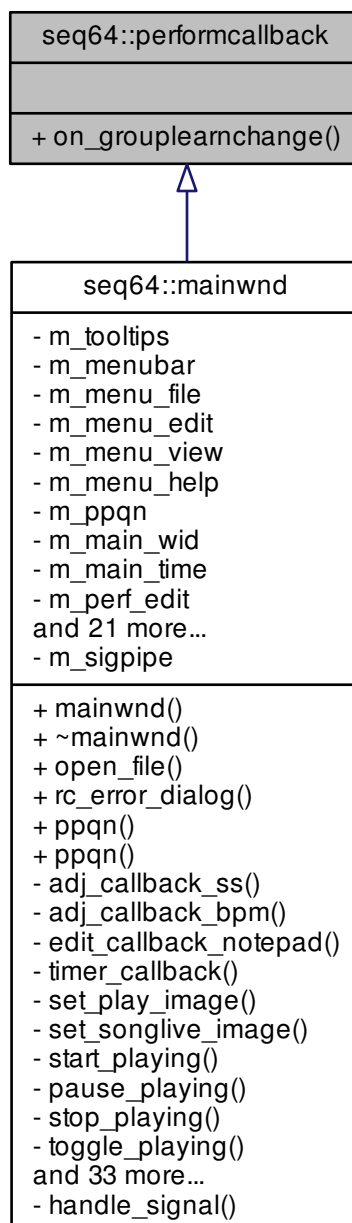
#### 13.69.6.76 m\_gui\_support

```
gui\_assistant& seq64::perform::m_gui_support [private]
```

### 13.70 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.*

### 13.70.1 Detailed Description

Provide a response to a group-learn change event.

## 13.70.2 Member Function Documentation

### 13.70.2.1 `on_grouplearnchange()`

```
virtual void seq64::performcallback::on_grouplearnchange (  
    bool ) [inline], [virtual]
```

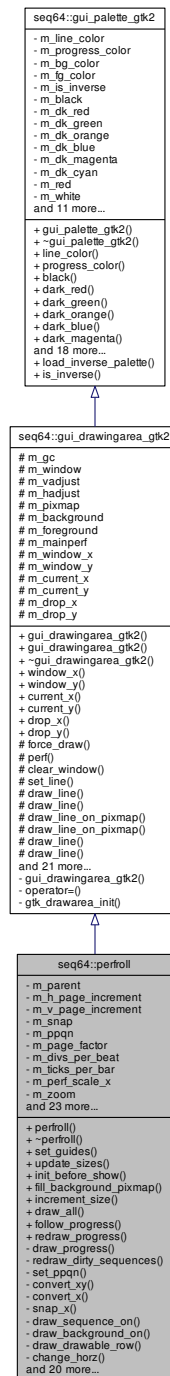
"state" is an Unused parameter.

Reimplemented in [seq64::mainwnd](#).

## 13.71 `seq64::perfroll` Class Reference

This class implements the performance roll user interface.

Inheritance diagram for seq64::perftroll:



## Public Member Functions

- [perftroll](#) ([perform](#) &[perf](#), [perfedit](#) &parent, Gtk::Adjustment &hadjust, Gtk::Adjustment &vadjust, int ppqn=S↵EQ64\_USE\_DEFAULT\_PPQN)

*Principal constructor.*

- virtual [~perftroll](#) ()

*This destructor deletes the interaction object.*

- void [set\\_guides](#) (int snap, int measure, int beat)  
*This function sets the `m_snap`, `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](#) ()
- void [redraw\\_progress](#) ()  
*Helper function to simplify the client call.*

## Private 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](#) &ticks, int &seq)  
*Converts (x, y) coordinates on the piano roll to tick (pulse) and sequence numbers.*
- void [convert\\_x](#) (int x, [midipulse](#) &ticks)  
*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 [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](#) (long y)  
*Not quite sure what this draws yet.*
- 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 `perfeddit`, so that it can forward it to any other `perfeddit` 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.*
- void [on\\_realize](#) ()  
*Provides the on-realization callback.*
- bool [on\\_expose\\_event](#) (GdkEventExpose \*ev)  
*Handles the on-expose event.*
- bool [on\\_button\\_press\\_event](#) (GdkEventButton \*ev)  
*This callback function handles a button press by forwarding it to the interaction object's button-press function.*
- 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.*
- bool [on\\_motion\\_notify\\_event](#) (GdkEventMotion \*ev)  
*Handles motion notification by forwarding it to the interaction object's motion-notification callback function.*
- bool [on\\_scroll\\_event](#) (GdkEventScroll \*ev)  
*Handles horizontal and vertical scrolling.*
- bool [on\\_focus\\_in\\_event](#) (GdkEventFocus \*ev)  
*This callback handles an in-focus event by setting the flag to HAS\_FOCUS.*
- bool [on\\_focus\\_out\\_event](#) (GdkEventFocus \*ev)  
*This callback handles an out-of-focus event by resetting the flag HAS\_FOCUS.*
- 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\(\)](#).*
- bool [on\\_key\\_press\\_event](#) (GdkEventKey \*ev)  
*This callback function handles a key-press event.*
- void [on\\_size\\_request](#) (GtkRequisition \*)  
*This do-nothing callback effectively throws away a size request.*

## Private Attributes

- [perfedit](#) & [m\\_parent](#)  
*Provides a link to the `perfedit` that created this object.*
- 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](#)  
*The amount of horizontal 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\\_zoom](#)  
*New value to attempt a rudimentary time-zoom feature.*
- int [m\\_names\\_y](#)  
*The maximum height of the perfrill names box, in pixes.*
- int [m\\_background\\_x](#)  
*The width of the perfrill background.*
- int [m\\_size\\_box\\_w](#)  
*This is a basically constant value set to `s_perfrill_size_box_w = 3`.*
- int [m\\_measure\\_length](#)  
*The legnth 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.*
- bool [m\\_have\\_button\\_press](#)  
*Used in the fruity and seq24 perfrill input classes to help with trigger push/pop management.*
- bool [m\\_transport\\_follow](#)  
*Indicates that the application should follow JACK transport.*
- bool [m\\_trans\\_button\\_press](#)  
*Indicates if the follow-transport button is pressed.*
- 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\\_trigger\\_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.*
- FruityPerfInput [m\\_fruity\\_interaction](#)  
*We need both styles of interaction object present.*
- Seq24PerfInput [m\\_seq24\\_interaction](#)  
*Provides support for standard Seq24 mouse handling, plus the keystroke handlers.*
- AbstractPerfInput & [m\\_interaction](#)  
*Provides a reference to the selected (at startup time) method of mouse interaction.*
- 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.*



## Friends

- class [FruityPerfInput](#)  
*These friend implement interaction-specific behavior, although only the Seq24 interactions support full keyboard processing, except for some common functionality provided by [perform::perftroll\\_key\\_event\(\)](#).*
- class [Seq24PerfInput](#)
- class [perftedit](#)

## Additional Inherited Members

### 13.71.1 Constructor & Destructor Documentation

#### 13.71.1.1 perftroll()

```
seq64::perftroll::perftroll (
    perform & perf,
    perftedit & parent,
    Gtk::Adjustment & hadjust,
    Gtk::Adjustment & vadjust,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

#### 13.71.1.2 ~perftroll()

```
seq64::perftroll::~~perftroll ( ) [virtual]
```

Well, now there are two objects, so no explicit deletion necessary.

### 13.71.2 Member Function Documentation

#### 13.71.2.1 set\_guides()

```
void seq64::perftroll::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 <a href="#">perfedit::set_guides()</a> . 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 <a href="#">perfedit::set_guides()</a> .
<i>beat</i>	Provides the number of beat-pulses (pulses per beat) as calculated in <a href="#">perfedit::set_guides()</a> .

## 13.71.2.2 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 perfroll 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.

## 13.71.2.3 init\_before\_show()

```
void seq64::perfroll::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.

## 13.71.2.4 fill\_background\_pixmap()

```
void seq64::perfroll::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.

**13.71.2.5 increment\_size()**

```
void seq64::perfroll::increment_size ( )
```

**13.71.2.6 draw\_all()**

```
void seq64::perfroll::draw_all ( )
```

m\_drop\_y is adjusted by [perfroll::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.

**13.71.2.7 follow\_progress()**

```
void seq64::perfroll::follow_progress ( )
```

**13.71.2.8 redraw\_progress()**

```
void seq64::perfroll::redraw_progress ( ) [inline]
```

**13.71.2.9 draw\_progress()**

```
void seq64::perfroll::draw_progress ( ) [private]
```

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. Now we are trying a newer function, [perform::get\\_max\\_tick\(\)](#), which seems to do the trick for resuming (instead of rewinding) the progress bar. It's still a tiny bit laggy, so we have to find a faster way to get the maximum. (Note that the draw\_progress function is called at every timeout, that is, constantly.)

The [perform::get\\_max\\_tick\(\)](#) call doesn't work with JACK: the progress bar rewinds to the beginning when playback is paused, though it does resume where it left off. It also may cause the progress bar to backtrack through any gap. Let's restore the [get\\_tick\(\)](#) call.

**13.71.2.10 redraw\_dirty\_sequences()**

```
void seq64::perfroll::redraw_dirty_sequences ( ) [private]
```

*Change Note* ca 2016-05-30 Lets try not drawing sequences greater than the maximum, at all.

**13.71.2.11 set\_ppqn()**

```
void seq64::perfroll::set_ppqn (
    int ppqn ) [private]
```

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`.

**13.71.2.12 convert\_xy()**

```
void seq64::perfroll::convert_xy (
    int x,
    int y,
    midipulse & d_tick,
    int & d_seq ) [private]
```

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.

**13.71.2.13 convert\_x()**

```
void seq64::perfroll::convert_x (
    int x,
    midipulse & tick ) [private]
```

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.

#### 13.71.2.14 snap\_x()

```
void seq64::perfroll::snap_x (
    int & x ) [private]
```

- m\_snap = number pulses to snap to
- m\_perf\_scale\_x = number of pulses per pixel

Therefore  $\text{mod} = \text{m\_snap} / \text{m\_perf\_scale\_x}$  equals the number pixels to snap to.

#### 13.71.2.15 draw\_sequence\_on()

```
void seq64::perfroll::draw_sequence_on (
    int seqnum ) [private]
```

Statement nesting from hell!

#### 13.71.2.16 draw\_background\_on()

```
void seq64::perfroll::draw_background_on (
    int seqnum ) [private]
```

#### 13.71.2.17 draw\_drawable\_row()

```
void seq64::perfroll::draw_drawable_row (
    long y ) [private]
```

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.

#### 13.71.2.18 change\_horz()

```
void seq64::perfroll::change_horz ( ) [private]
```

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.

**13.71.2.19 change\_vert()**

```
void seq64::perfroll::change_vert ( ) [private]
```

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.

**13.71.2.20 split\_trigger()**

```
void seq64::perfroll::split_trigger (
    int sequence,
    midipulse tick ) [private]
```

**13.71.2.21 enqueue\_draw()**

```
void seq64::perfroll::enqueue_draw ( ) [private]
```

The parent `perfdit` will call `perfroll::queue_draw()` on behalf of this object, and it will pass a [perfroll::enqueue\\_draw\(\)](#) to the peer `perfdit`'s `perfroll`, if the peer exists.

**13.71.2.22 set\_zoom()**

```
void seq64::perfroll::set_zoom (
    int z ) [private]
```

*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.

**13.71.2.23 convert\_drop\_xy()**

```
void seq64::perfroll::convert_drop_xy ( ) [inline], [private]
```

**13.71.2.24 horizontal\_adjust()**

```
void seq64::perfroll::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 <a href="#">gui_drawingarea_gtk2::scroll_hadjust()</a> for more information.
-------------	---

## 13.71.2.25 vertical\_adjust()

```
void seq64::perfroll::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 <a href="#">gui_drawingarea_gtk2::scroll_vadjust()</a> for more information.
-------------	---

## 13.71.2.26 horizontal\_set()

```
void seq64::perfroll::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).
--------------	---

## 13.71.2.27 vertical\_set()

```
void seq64::perfroll::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).
--------------	---

**13.71.2.28 on\_realize()**

```
void seq64::perfroll::on_realize ( ) [private]
```

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...
```

**13.71.2.29 on\_expose\_event()**

```
bool seq64::perfroll::on_expose_event (
    GdkEventExpose * ev ) [private]
```

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**

ev	Provides the expose event.
----	----------------------------

**Returns**

Always returns true.

**13.71.2.30 on\_button\_press\_event()**

```
bool seq64::perfroll::on_button_press_event (
    GdkEventButton * ev ) [private]
```

This gives us Seq24 versus Fruity behavior.

One minor issue: Fruity behavior doesn't yet provide the keystroke behavior we now handle for the Seq24 mode of operation.



### 13.71.2.31 on\_button\_release\_event()

```
bool seq64::perfroll::on_button_release_event (
    GdkEventButton * ev ) [private]
```

This gives us Seq24 versus Fruity behavior.

### 13.71.2.32 on\_motion\_notify\_event()

```
bool seq64::perfroll::on_motion_notify_event (
    GdkEventMotion * ev ) [private]
```

### 13.71.2.33 on\_scroll\_event()

```
bool seq64::perfroll::on_scroll_event (
    GdkEventScroll * ev ) [private]
```

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

ev	Provides the scroll event.
----	----------------------------

#### Returns

Currently always returns true.

### 13.71.2.34 on\_focus\_in\_event()

```
bool seq64::perfroll::on_focus_in_event (
    GdkEventFocus * ev ) [private]
```

### 13.71.2.35 on\_focus\_out\_event()

```
bool seq64::perfroll::on_focus_out_event (
    GdkEventFocus * ev ) [private]
```

**13.71.2.36 on\_size\_allocate()**

```
void seq64::perffroll::on_size_allocate (
    Gtk::Allocation & al ) [private]
```

**13.71.2.37 on\_key\_press\_event()**

```
bool seq64::perffroll::on_key_press_event (
    GdkEventKey * ev ) [private]
```

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::perffroll_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 `perffroll_key_event()` function.

The `perffroll_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 `perffroll_key_event()` if the Ctrl key was pressed. Be careful.

**13.71.2.38 on\_size\_request()**

```
void seq64::perffroll::on_size_request (
    GtkRequisition * ) [inline], [private]
```

**13.71.3 Friends And Related Function Documentation****13.71.3.1 FruityPerfInput**

```
friend class FruityPerfInput [friend]
```

The `perfeddit` class needs access to the private `enqueue_draw()` function.

### 13.71.3.2 Seq24PerfInput

```
friend class Seq24PerfInput [friend]
```

### 13.71.3.3 perfedit

```
friend class perfedit [friend]
```

## 13.71.4 Field Documentation

### 13.71.4.1 m\_parent

```
perfedit& seq64::perfroll::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.

### 13.71.4.2 m\_h\_page\_increment

```
int seq64::perfroll::m_h_page_increment [private]
```

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.

### 13.71.4.3 m\_v\_page\_increment

```
int seq64::perfroll::m_v_page_increment [private]
```

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.

### 13.71.4.4 m\_snap

```
int seq64::perfroll::m_snap [private]
```

### 13.71.4.5 m\_ppqn

```
int seq64::perfroll::m_ppqn [private]
```

#### 13.71.4.6 m\_page\_factor

```
int seq64::perfroll::m_page_factor [private]
```

#### 13.71.4.7 m\_divs\_per\_beat

```
int seq64::perfroll::m_divs_per_beat [private]
```

#### 13.71.4.8 m\_ticks\_per\_bar

```
midipulse seq64::perfroll::m_ticks_per_bar [private]
```

#### 13.71.4.9 m\_perf\_scale\_x

```
int seq64::perfroll::m_perf_scale_x [private]
```

#### 13.71.4.10 m\_zoom

```
int seq64::perfroll::m_zoom [private]
```

It seems to work pretty well now.

#### 13.71.4.11 m\_names\_y

```
int seq64::perfroll::m_names_y [private]
```

This is currently semantically a constant set to `c_names_y = 24`.

#### 13.71.4.12 m\_background\_x

```
int seq64::perfroll::m_background_x [private]
```

This is based on the `m_ppqn` value and the value of `c_perf_scale_x` (or is `m_perf_scale_x` preferable?)

**13.71.4.13 m\_size\_box\_w**

```
int seq64::perfroll::m_size_box_w [private]
```

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.

**13.71.4.14 m\_measure\_length**

```
int seq64::perfroll::m_measure_length [private]
```

**13.71.4.15 m\_beat\_length**

```
int seq64::perfroll::m_beat_length [private]
```

**13.71.4.16 m\_old\_progress\_ticks**

```
midipulse seq64::perfroll::m_old_progress_ticks [private]
```

See the [draw\\_progress\(\)](#) function. This could almost be static inside that function.

**13.71.4.17 m\_have\_button\_press**

```
bool seq64::perfroll::m_have_button_press [private]
```

**13.71.4.18 m\_transport\_follow**

```
bool seq64::perfroll::m_transport_follow [private]
```

The alternative is ...?

**13.71.4.19 m\_trans\_button\_press**

```
bool seq64::perfroll::m_trans_button_press [private]
```

#### 13.71.4.20 m\_4bar\_offset

```
midipulse seq64::perfroll::m_4bar_offset [private]
```

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.

#### 13.71.4.21 m\_sequence\_offset

```
int seq64::perfroll::m_sequence_offset [private]
```

It is obtained or changed when the vertical scroll-bar moves. It is used for drawing the correct vertical window in the piano roll.

#### 13.71.4.22 m\_roll\_length\_ticks

```
int seq64::perfroll::m_roll_length_ticks [private]
```

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\(\)](#).

#### 13.71.4.23 m\_drop\_tick

```
midipulse seq64::perfroll::m_drop_tick [private]
```

Used only by the friend modules `perfroll_input` and `fruityperfroll_input`.

#### 13.71.4.24 m\_drop\_tick\_trigger\_offset

```
midipulse seq64::perfroll::m_drop_tick_trigger_offset [private]
```

Used only by the friend modules `perfroll_input` and `fruityperfroll_input`.

#### 13.71.4.25 m\_drop\_sequence

```
int seq64::perfroll::m_drop_sequence [private]
```

Used for redrawing the sequence.

#### 13.71.4.26 m\_sequence\_max

```
int seq64::perfroll::m_sequence_max [private]
```

#### 13.71.4.27 m\_sequence\_active

```
bool seq64::perfroll::m_sequence_active[c_max_sequence] [private]
```

Not sure yet why we can't just use the sequence's member function to access this status boolean.

#### 13.71.4.28 m\_fruity\_interaction

```
FruityPerfInput seq64::perfroll::m_fruity_interaction [private]
```

Even if the user specifies the fruity interaction, the Seq24 interaction is still needed to handle our new keystroke support for the perfroll. We need both objects to exist all the time, similar to the Fruity/Seq24 roles in the seqroll object.

*Obsolete* [AbstractPerfInput](#) \* m\_interaction

#### 13.71.4.29 m\_seq24\_interaction

```
Seq24PerfInput seq64::perfroll::m_seq24_interaction [private]
```

#### 13.71.4.30 m\_interaction

```
AbstractPerfInput& seq64::perfroll::m_interaction [private]
```

#### 13.71.4.31 m\_moving

```
bool seq64::perfroll::m_moving [private]
```

#### 13.71.4.32 m\_growing

```
bool seq64::perfroll::m_growing [private]
```

#### 13.71.4.33 m\_grow\_direction

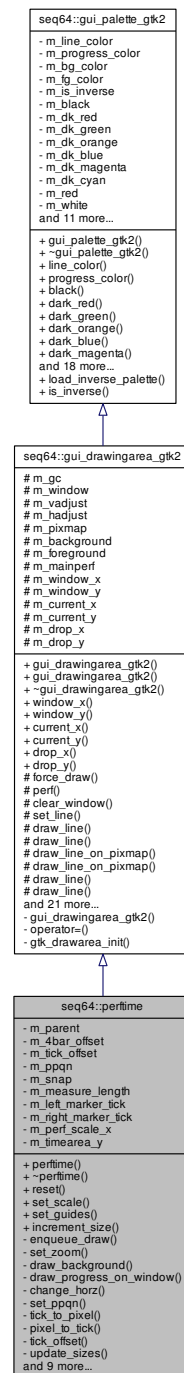
```
bool seq64::perfroll::m_grow_direction [private]
```

Determines whether the section is growing to the left or to the right.

## 13.72 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 `perfeditt`, so that it can forward it to any other `perfeditt` 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

### 13.72.1 Constructor & Destructor Documentation

#### 13.72.1.1 `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.

## 13.72.1.2 ~perftime()

```
virtual seq64::perftime::~~perftime ( ) [inline], [virtual]
```

## 13.72.2 Member Function Documentation

## 13.72.2.1 reset()

```
void seq64::perftime::reset ( )
```

## 13.72.2.2 set\_scale()

```
void seq64::perftime::set_scale (
    int scale )
```

## 13.72.2.3 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 <a href="#">perfedit::set_guides()</a> . 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 <a href="#">perfedit::set_guides()</a> .

**13.72.2.4 increment\_size()**

```
void seq64::perftime::increment_size ( ) [inline]
```

Compare it to [perftroll::increment\\_size\(\)](#).

**13.72.2.5 enqueue\_draw()**

```
void seq64::perftime::enqueue_draw ( ) [private]
```

The parent perftedit will call `perftime::queue_draw()` on behalf of this object, and it will pass a [perftime::enqueue\\_↔draw\(\)](#) to the peer perftedit's `perftime`, if the peer exists.

**13.72.2.6 set\_zoom()**

```
void seq64::perftime::set_zoom (
    int z ) [private]
```

Redraws the background if the new zoom checked out.

**Parameters**

<b>z</b>	Provides the zoom value, which is checked, and then copied into <code>m_perf_scale_x</code> .
----------	---

**13.72.2.7 draw\_background()**

```
void seq64::perftime::draw_background ( ) [private]
```

Note that `m_measure_length == 0` will cause integer overflow.

**13.72.2.8 draw\_progress\_on\_window()**

```
void seq64::perftime::draw_progress_on_window ( ) [private]
```

**13.72.2.9 change\_horz()**

```
void seq64::perftime::change_horz ( ) [private]
```

Again, uses the constant, 16 [now offloaded to the new [tick\\_offset\(\)](#) function.].

### 13.72.2.10 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.
-------------	----------------------------------

### 13.72.2.11 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.

### 13.72.2.12 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.

### 13.72.2.13 tick\_offset()

```
int seq64::perftime::tick_offset ( ) [inline], [private]
```

#### Returns

Returns `m_4bar_offset * 16 * m_ppqn`.

### 13.72.2.14 update\_sizes()

```
void seq64::perftime::update_sizes ( ) [inline], [private]
```

### 13.72.2.15 idle\_progress()

```
int seq64::perftime::idle_progress ( ) [inline], [private]
```

### 13.72.2.16 update\_pixmap()

```
void seq64::perftime::update_pixmap ( ) [inline], [private]
```

### 13.72.2.17 draw\_pixmap\_on\_window()

```
void seq64::perftime::draw_pixmap_on_window ( ) [inline], [private]
```

### 13.72.2.18 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);
```

### 13.72.2.19 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.

**13.72.2.20 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).

**13.72.2.21 on\_size\_allocate()**

```
void seq64::perftime::on_size_allocate (
    Gtk::Allocation & r ) [private]
```

**13.72.2.22 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

### 13.72.2.23 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.

## 13.72.3 Friends And Related Function Documentation

### 13.72.3.1 perfedit

```
friend class perfedit [friend]
```

## 13.72.4 Field Documentation

### 13.72.4.1 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.

### 13.72.4.2 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.

### 13.72.4.3 m\_tick\_offset

```
int seq64::perftime::m_tick_offset [private]
```

Why 16?



#### 13.72.4.4 m\_ppqn

```
int seq64::perftime::m_ppqn [private]
```

#### 13.72.4.5 m\_snap

```
int seq64::perftime::m_snap [private]
```

#### 13.72.4.6 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.

#### 13.72.4.7 m\_left\_marker\_tick

```
int seq64::perftime::m_left_marker_tick [private]
```

Otherwise it is -1. Supports perftime's keystroke processing.

#### 13.72.4.8 m\_right\_marker\_tick

```
int seq64::perftime::m_right_marker_tick [private]
```

Otherwise it is -1. Supports perftime's keystroke processing.

#### 13.72.4.9 m\_perf\_scale\_x

```
int seq64::perftime::m_perf_scale_x [private]
```

#### 13.72.4.10 m\_timearea\_y

```
int seq64::perftime::m_timearea_y [private]
```

### 13.73 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.*

### 13.73.1 Detailed Description

Except for the virtual-vs-normal status, this information is obtained by scanning the system at the startup time of the application.

### 13.73.2 Field Documentation

#### 13.73.2.1 m\_client\_number

```
int seq64::midi_port_info::port_info_t::m_client_number
```

#### 13.73.2.2 m\_client\_name

```
std::string seq64::midi_port_info::port_info_t::m_client_name
```

#### 13.73.2.3 m\_port\_number

```
int seq64::midi_port_info::port_info_t::m_port_number
```

#### 13.73.2.4 m\_port\_name

```
std::string seq64::midi_port_info::port_info_t::m_port_name
```

#### 13.73.2.5 m\_queue\_number

```
int seq64::midi_port_info::port_info_t::m_queue_number
```

#### 13.73.2.6 m\_is\_input

```
bool seq64::midi_port_info::port_info_t::m_is_input
```

#### 13.73.2.7 m\_is\_virtual

```
bool seq64::midi_port_info::port_info_t::m_is_virtual
```

#### 13.73.2.8 m\_is\_system

```
bool seq64::midi_port_info::port_info_t::m_is_system
```

## 13.74 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.*
- 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*
- 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*
- const std::string & [filename](#) () const
- 'Getter' 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*
- const std::string & [config\\_directory](#) () const
- 'Getter' function for member m\_config\_directory*
- 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*

## Protected Member Functions

- 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\\_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*
- 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 [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.*
- void [interaction\\_method](#) ([interaction\\_method\\_t](#) value)  
*'Setter' function for member m\_interaction\_method*
- void [filename](#) (const std::string &value)  
*'Setter' function for member m\_filename*
- void [jack\\_session\\_uuid](#) (const std::string &value)  
*'Setter' function for member m\_jack\_session\_uuid*
- void [last\\_used\\_dir](#) (const std::string &value)  
*'Setter' function for member m\_last\_used\_dir*
- void [config\\_directory](#) (const std::string &value)  
*'Setter' function for member m\_config\_directory*
- void [set\\_config\\_files](#) (const std::string &value)  
*'Setter' function for member m\_config\_filename and m\_user\_filename*
- 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 Member Functions

- std::string [home\\_config\\_directory](#) () const  
*Provides the directory for the configuration file, and also creates the directory if necessary.*

## Private Attributes

- 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]*
- 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.*

## Friends

- class `optionsfile`
- class `options`
- class `mainwnd`
- class `rtmidi_info`
- `int parse_command_line_options (perform &p, int argc, char *argv [])`  
*Parses the command-line options on behalf of the application.*
- `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.*

## 13.74.1 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.

## 13.74.2 Constructor & Destructor Documentation

### 13.74.2.1 `rc_settings()` [1/2]

```
seq64::rc_settings::rc_settings ( )
```

### 13.74.2.2 `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.
------------	--------------------------------------

### 13.74.3 Member Function Documentation

#### 13.74.3.1 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.

#### 13.74.3.2 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.

#### 13.74.3.3 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.

**13.74.3.4 set\_defaults()**

```
void seq64::rc_settings::set_defaults ( )
```

**13.74.3.5 auto\_option\_save()** [1/2]

```
bool seq64::rc_settings::auto_option_save ( ) const [inline]
```

**13.74.3.6 legacy\_format()** [1/2]

```
bool seq64::rc_settings::legacy_format ( ) const [inline]
```

**13.74.3.7 lash\_support()** [1/2]

```
bool seq64::rc_settings::lash_support ( ) const [inline]
```

**13.74.3.8 allow\_mod4\_mode()** [1/2]

```
bool seq64::rc_settings::allow_mod4_mode ( ) const [inline]
```

**13.74.3.9 allow\_snap\_split()** [1/2]

```
bool seq64::rc_settings::allow_snap_split ( ) const [inline]
```

**13.74.3.10 allow\_click\_edit()** [1/2]

```
bool seq64::rc_settings::allow_click_edit ( ) const [inline]
```

**13.74.3.11 show\_midi()** [1/2]

```
bool seq64::rc_settings::show_midi ( ) const [inline]
```

**13.74.3.12 priority()** [1/2]

```
bool seq64::rc_settings::priority ( ) const [inline]
```

**13.74.3.13 stats()** [1/2]

```
bool seq64::rc_settings::stats ( ) const [inline]
```

**13.74.3.14 pass\_sysex()** [1/2]

```
bool seq64::rc_settings::pass_sysex ( ) const [inline]
```

**13.74.3.15 with\_jack\_transport()** [1/2]

```
bool seq64::rc_settings::with_jack_transport ( ) const [inline]
```

**13.74.3.16 with\_jack\_master()** [1/2]

```
bool seq64::rc_settings::with_jack_master ( ) const [inline]
```

**13.74.3.17 with\_jack\_master\_cond()** [1/2]

```
bool seq64::rc_settings::with_jack_master_cond ( ) const [inline]
```

**13.74.3.18 with\_jack\_midi()** [1/2]

```
bool seq64::rc_settings::with_jack_midi ( ) const [inline]
```

**13.74.3.19 with\_jack()**

```
bool seq64::rc_settings::with_jack ( ) const [inline]
```

Do not confuse these original options with the new "no JACK MIDI" option.

**13.74.3.20 filter\_by\_channel()** [1/2]

```
bool seq64::rc_settings::filter_by_channel ( ) const [inline]
```

**13.74.3.21 manual\_alsa\_ports()** [1/2]

```
bool seq64::rc_settings::manual_alsa_ports ( ) const [inline]
```

**13.74.3.22 reveal\_alsa\_ports()** [1/2]

```
bool seq64::rc_settings::reveal_alsa_ports ( ) const [inline]
```

**13.74.3.23 print\_keys()** [1/2]

```
bool seq64::rc_settings::print_keys ( ) const [inline]
```

**13.74.3.24 device\_ignore()** [1/2]

```
bool seq64::rc_settings::device_ignore ( ) const [inline]
```

**13.74.3.25 device\_ignore\_num()** [1/2]

```
int seq64::rc_settings::device_ignore_num ( ) const [inline]
```

**13.74.3.26 interaction\_method()** [1/2]

```
interaction_method_t seq64::rc_settings::interaction_method ( ) const [inline]
```

**13.74.3.27 filename()** [1/2]

```
const std::string& seq64::rc_settings::filename ( ) const [inline]
```

**13.74.3.28 jack\_session\_uuid()** [1/2]

```
const std::string& seq64::rc_settings::jack_session_uuid ( ) const [inline]
```

**13.74.3.29 last\_used\_dir()** [1/2]

```
const std::string& seq64::rc_settings::last_used_dir ( ) const [inline]
```

**13.74.3.30 config\_directory()** [1/2]

```
const std::string& seq64::rc_settings::config_directory ( ) const [inline]
```

**13.74.3.31 config\_filename()** [1/2]

```
const std::string& seq64::rc_settings::config_filename ( ) const [inline]
```

**13.74.3.32 user\_filename()** [1/2]

```
const std::string& seq64::rc_settings::user_filename ( ) const [inline]
```

**13.74.3.33 config\_filename\_alt()** [1/2]

```
const std::string& seq64::rc_settings::config_filename_alt ( ) const [inline]
```

**13.74.3.34 user\_filename\_alt()** [1/2]

```
const std::string& seq64::rc_settings::user_filename_alt ( ) const [inline]
```

**13.74.3.35 application\_name()**

```
const std::string seq64::rc_settings::application_name ( ) const [inline]
```

**13.74.3.36 app\_client\_name()**

```
const std::string& seq64::rc_settings::app_client_name ( ) const [inline]
```

**13.74.3.37 auto\_option\_save()** [2/2]

```
void seq64::rc_settings::auto_option_save (
    bool flag ) [inline], [protected]
```

**13.74.3.38 legacy\_format()** [2/2]

```
void seq64::rc_settings::legacy_format (
    bool flag ) [inline], [protected]
```

**13.74.3.39 lash\_support()** [2/2]

```
void seq64::rc_settings::lash_support (
    bool flag ) [inline], [protected]
```

**13.74.3.40 allow\_mod4\_mode()** [2/2]

```
void seq64::rc_settings::allow_mod4_mode (
    bool flag ) [inline], [protected]
```

**13.74.3.41 allow\_snap\_split()** [2/2]

```
void seq64::rc_settings::allow_snap_split (
    bool flag ) [inline], [protected]
```

**13.74.3.42 allow\_click\_edit()** [2/2]

```
void seq64::rc_settings::allow_click_edit (
    bool flag ) [inline], [protected]
```

**13.74.3.43 show\_midi()** [2/2]

```
void seq64::rc_settings::show_midi (
    bool flag ) [inline], [protected]
```

**13.74.3.44 priority()** [2/2]

```
void seq64::rc_settings::priority (
    bool flag ) [inline], [protected]
```

**13.74.3.45 stats()** [2/2]

```
void seq64::rc_settings::stats (
    bool flag ) [inline], [protected]
```

**13.74.3.46 pass\_sysex()** [2/2]

```
void seq64::rc_settings::pass_sysex (
    bool flag ) [inline], [protected]
```

**13.74.3.47 with\_jack\_transport()** [2/2]

```
void seq64::rc_settings::with_jack_transport (
    bool flag ) [protected]
```

**13.74.3.48 with\_jack\_master()** [2/2]

```
void seq64::rc_settings::with_jack_master (
    bool flag ) [protected]
```

**13.74.3.49 with\_jack\_master\_cond()** [2/2]

```
void seq64::rc_settings::with_jack_master_cond (
    bool flag ) [protected]
```

**13.74.3.50 with\_jack\_midi()** [2/2]

```
void seq64::rc_settings::with_jack_midi (
    bool flag )    [inline], [protected]
```

**13.74.3.51 filter\_by\_channel()** [2/2]

```
void seq64::rc_settings::filter_by_channel (
    bool flag )    [inline], [protected]
```

**13.74.3.52 manual\_alsa\_ports()** [2/2]

```
void seq64::rc_settings::manual_alsa_ports (
    bool flag )    [inline], [protected]
```

**13.74.3.53 reveal\_alsa\_ports()** [2/2]

```
void seq64::rc_settings::reveal_alsa_ports (
    bool flag )    [inline], [protected]
```

**13.74.3.54 print\_keys()** [2/2]

```
void seq64::rc_settings::print_keys (
    bool flag )    [inline], [protected]
```

**13.74.3.55 device\_ignore()** [2/2]

```
void seq64::rc_settings::device_ignore (
    bool flag )    [inline], [protected]
```

**13.74.3.56 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.
--------------	---------------------------------------

**13.74.3.57 interaction\_method()** [2/2]

```
void seq64::rc_settings::interaction_method (  
    interaction_method_t value ) [protected]
```

## Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

**13.74.3.58 filename()** [2/2]

```
void seq64::rc_settings::filename (  
    const std::string & value ) [protected]
```

## Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

**13.74.3.59 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.
--------------	---------------------------------------

**13.74.3.60 last\_used\_dir()** [2/2]

```
void seq64::rc_settings::last_used_dir (  
    const std::string & value ) [protected]
```

## Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

**13.74.3.61** `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.
--------------	---------------------------------------

**13.74.3.62** `set_config_files()`

```
void seq64::rc_settings::set_config_files (
    const std::string & value ) [protected]
```

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.
--------------	--

**13.74.3.63** `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.
--------------	---

**13.74.3.64** `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.
--------------	--

## 13.74.3.65 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.
--------------	---

## 13.74.3.66 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.
--------------	---------------------------------------

## 13.74.3.67 home\_config\_directory()

```
std::string seq64::rc_settings::home_config_directory ( ) const [private]
```

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".

## Returns

Returns the selected home configuration directory. If it does not exist, or could not be created, then an empty string is returned.

## 13.74.4 Friends And Related Function Documentation

#### 13.74.4.1 optionsfile

```
friend class optionsfile [friend]
```

#### 13.74.4.2 options

```
friend class options [friend]
```

#### 13.74.4.3 mainwnd

```
friend class mainwnd [friend]
```

#### 13.74.4.4 rtmidi\_info

```
friend class rtmidi_info [friend]
```

#### 13.74.4.5 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.

#### 13.74.4.6 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.

### 13.74.5 Field Documentation

#### 13.74.5.1 m\_auto\_option\_save

```
bool seq64::rc_settings::m_auto_option_save [private]
```

#### 13.74.5.2 m\_legacy\_format

```
bool seq64::rc_settings::m_legacy_format [private]
```

#### 13.74.5.3 m\_lash\_support

```
bool seq64::rc_settings::m_lash_support [private]
```

#### 13.74.5.4 m\_allow\_mod4\_mode

```
bool seq64::rc_settings::m_allow_mod4_mode [private]
```

#### 13.74.5.5 m\_allow\_snap\_split

```
bool seq64::rc_settings::m_allow_snap_split [private]
```

#### 13.74.5.6 m\_allow\_click\_edit

```
bool seq64::rc_settings::m_allow_click_edit [private]
```

#### 13.74.5.7 m\_show\_midi

```
bool seq64::rc_settings::m_show_midi [private]
```

#### 13.74.5.8 m\_priority

```
bool seq64::rc_settings::m_priority [private]
```

#### 13.74.5.9 m\_stats

```
bool seq64::rc_settings::m_stats [private]
```

#### 13.74.5.10 m\_pass\_sysex

```
bool seq64::rc_settings::m_pass_sysex [private]
```

#### 13.74.5.11 m\_with\_jack\_transport

```
bool seq64::rc_settings::m_with_jack_transport [private]
```

#### 13.74.5.12 m\_with\_jack\_master

```
bool seq64::rc_settings::m_with_jack_master [private]
```

**13.74.5.13 m\_with\_jack\_master\_cond**

```
bool seq64::rc_settings::m_with_jack_master_cond [private]
```

**13.74.5.14 m\_with\_jack\_midi**

```
bool seq64::rc_settings::m_with_jack_midi [private]
```

**13.74.5.15 m\_filter\_by\_channel**

```
bool seq64::rc_settings::m_filter_by_channel [private]
```

**13.74.5.16 m\_manual\_alsa\_ports**

```
bool seq64::rc_settings::m_manual_alsa_ports [private]
```

**13.74.5.17 m\_reveal\_alsa\_ports**

```
bool seq64::rc_settings::m_reveal_alsa_ports [private]
```

**13.74.5.18 m\_print\_keys**

```
bool seq64::rc_settings::m_print_keys [private]
```

**13.74.5.19 m\_device\_ignore**

```
bool seq64::rc_settings::m_device_ignore [private]
```

**13.74.5.20 m\_device\_ignore\_num**

```
int seq64::rc_settings::m_device_ignore_num [private]
```

#### 13.74.5.21 m\_interaction\_method

```
interaction_method_t seq64::rc_settings::m_interaction_method [private]
```

#### 13.74.5.22 m\_filename

```
std::string seq64::rc_settings::m_filename [private]
```

#### 13.74.5.23 m\_jack\_session\_uuid

```
std::string seq64::rc_settings::m_jack_session_uuid [private]
```

#### 13.74.5.24 m\_last\_used\_dir

```
std::string seq64::rc_settings::m_last_used_dir [private]
```

#### 13.74.5.25 m\_config\_directory

```
std::string seq64::rc_settings::m_config_directory [private]
```

This value is "~/config/sequencer64" by default.

#### 13.74.5.26 m\_config\_filename

```
std::string seq64::rc_settings::m_config_filename [private]
```

This value is "sequencer64.rc" by default.

#### 13.74.5.27 m\_user\_filename

```
std::string seq64::rc_settings::m_user_filename [private]
```

This value is "sequencer64 usr" by default.

#### 13.74.5.28 m\_config\_filename\_alt

```
std::string seq64::rc_settings::m_config_filename_alt [private]
```



#### 13.74.5.29 m\_user\_filename\_alt

```
std::string seq64::rc_settings::m_user_filename_alt [private]
```

#### 13.74.5.30 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.

#### 13.74.5.31 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.

## 13.75 seq64::rect Class Reference

A small helper class representing a rectangle.

### Data Fields

- int [x](#)  
*The x-coordinate of the origin of the rectangle.*
- int [y](#)  
*The y-coordinate of the origin of the rectangle.*
- int [height](#)  
*The height of the rectangle, in units of pixels.*
- int [width](#)  
*The width of the rectangle, in units of pixels.*

### 13.75.1 Field Documentation

#### 13.75.1.1 x

```
int seq64::rect::x
```

### 13.75.1.2 y

```
int seq64::rect::y
```

### 13.75.1.3 height

```
int seq64::rect::height
```

### 13.75.1.4 width

```
int seq64::rect::width
```

## 13.76 seq64::gui\_drawingarea\_gtk2::rect Struct Reference

A small helper structure representing a rectangle.

### Data Fields

- int [x](#)
- int [y](#)
- int [height](#)
- int [width](#)

### 13.76.1 Field Documentation

#### 13.76.1.1 x

```
int seq64::gui_drawingarea_gtk2::rect::x
```

#### 13.76.1.2 y

```
int seq64::gui_drawingarea_gtk2::rect::y
```

### 13.76.1.3 height

```
int seq64::gui_drawingarea_gtk2::rect::height
```

### 13.76.1.4 width

```
int seq64::gui_drawingarea_gtk2::rect::width
```

## 13.77 seq64::rterror Class Reference

Exception handling class for rtexmidi.

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.*

### 13.77.1 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.

### 13.77.2 Member Enumeration Documentation

#### 13.77.2.1 Type

```
enum seq64::rterror::Type
```

##### Enumerator

WARNING	A non-critical error.
DEBUG_WARNING	Non-critical error useful for debugging.
UNSPECIFIED	The default, unspecified error type.
NO_DEVICES_FOUND	No devices found on system.
INVALID_DEVICE	An invalid device ID was specified.
MEMORY_ERROR	An error occurred during memory allocation.
INVALID_PARAMETER	Invalid parameter specified to a function.
INVALID_USE	The function was called incorrectly.
DRIVER_ERROR	A system driver error occurred.
SYSTEM_ERROR	A system error occurred.
THREAD_ERROR	A thread error occurred.

### 13.77.3 Constructor & Destructor Documentation

#### 13.77.3.1 `rterror()`

```
seq64::rterror::rterror (
    const std::string & message,
    Type type = rterror::UNSPECIFIED ) [inline]
```

#### 13.77.3.2 `~rterror()`

```
virtual seq64::rterror::~~rterror ( ) [inline], [virtual]
```

## 13.77.4 Member Function Documentation

### 13.77.4.1 print\_message()

```
virtual void seq64::rterror::print_message ( ) const [inline], [virtual]
```

### 13.77.4.2 getType()

```
virtual const Type& seq64::rterror::getType ( ) const [inline], [virtual]
```

### 13.77.4.3 get\_message()

```
virtual const std::string& seq64::rterror::get_message ( ) const [inline], [virtual]
```

### 13.77.4.4 what()

```
virtual const char* seq64::rterror::what ( ) const [inline], [virtual], [noexcept]
```

## 13.77.5 Field Documentation

### 13.77.5.1 m\_message

```
std::string seq64::rterror::m_message [private]
```

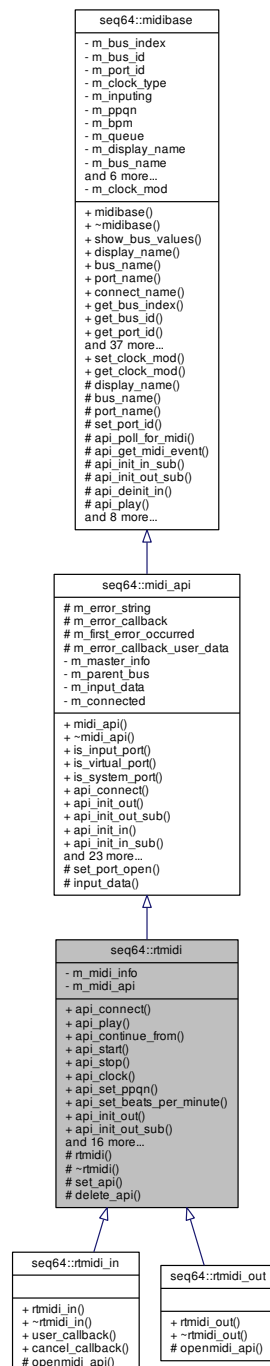
### 13.77.5.2 m\_type

```
Type seq64::rterror::m_type [private]
```

## 13.78 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)  
*Default constructor.*
- virtual [~rtmidi](#) ()  
*Destructor.*
- 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

### 13.78.1 Detailed Description

We moved the enum `Api` definition into the new `rtmidi_types.hpp` module to make refactoring the code easier.

### 13.78.2 Constructor & Destructor Documentation

#### 13.78.2.1 `rtmidi()`

```
seq64::rtmidi::rtmidi (
    midibus & parentbus,
    rtmidi_info & info ) [protected]
```

##### Parameters

<i>parentbus</i>	This is the midibus that the <code>rtmidi</code> object is going to implement, by forwarding calls to the selected MIDI subsystem (e.g. ALSA or JACK).
<i>info</i>	This object provides the system's enumerated busses/ports as found by the selected MIDI subsystem.

#### 13.78.2.2 `~rtmidi()`

```
seq64::rtmidi::~~rtmidi ( ) [protected], [virtual]
```

### 13.78.3 Member Function Documentation

#### 13.78.3.1 `api_connect()`

```
virtual bool seq64::rtmidi::api_connect ( ) [inline], [virtual]
```

Reimplemented from [seq64::midi\\_api](#).



### 13.78.3.2 api\_play()

```
virtual void seq64::rtmidi::api_play (
    event * e24,
    midibyte channel ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

### 13.78.3.3 api\_continue\_from()

```
virtual void seq64::rtmidi::api_continue_from (
    midipulse tick,
    midipulse beats ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

### 13.78.3.4 api\_start()

```
virtual void seq64::rtmidi::api_start ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

### 13.78.3.5 api\_stop()

```
virtual void seq64::rtmidi::api_stop ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

### 13.78.3.6 api\_clock()

```
virtual void seq64::rtmidi::api_clock (
    midipulse tick ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

### 13.78.3.7 api\_set\_ppqn()

```
virtual void seq64::rtmidi::api_set_ppqn (
    int ppqn ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.8 `api_set_beats_per_minute()`

```
virtual void seq64::rtmidi::api_set_beats_per_minute (
    midi\_bpm bpm ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.9 `api_init_out()`

```
virtual bool seq64::rtmidi::api_init_out ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.10 `api_init_out_sub()`

```
virtual bool seq64::rtmidi::api_init_out_sub ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.11 `api_init_in()`

```
virtual bool seq64::rtmidi::api_init_in ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.12 `api_init_in_sub()`

```
virtual bool seq64::rtmidi::api_init_in_sub ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.13 `api_deinit_in()`

```
virtual bool seq64::rtmidi::api_deinit_in ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.14 api\_get\_midi\_event()

```
virtual bool seq64::rtmidi::api_get_midi_event (
    event * inev ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.15 api\_poll\_for\_midi()

```
virtual int seq64::rtmidi::api_poll_for_midi ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.16 api\_sysex()

```
virtual void seq64::rtmidi::api_sysex (
    event * e24 ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.17 api\_flush()

```
virtual void seq64::rtmidi::api_flush ( ) [inline], [virtual]
```

Implements [seq64::midi\\_api](#).

#### 13.78.3.18 is\_port\_open()

```
virtual bool seq64::rtmidi::is_port_open ( ) const [inline], [virtual]
```

#### 13.78.3.19 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.

#### 13.78.3.20 get\_bus\_name()

```
virtual std::string seq64::rtmidi::get_bus_name ( ) [inline], [virtual]
```

##### Returns

Returns the buss name from the selected API subsystem.

#### 13.78.3.21 get\_port\_id()

```
virtual int seq64::rtmidi::get_port_id ( ) [inline], [virtual]
```

##### Returns

Returns the port ID number from the selected API subsystem.

#### 13.78.3.22 get\_port\_name()

```
virtual std::string seq64::rtmidi::get_port_name ( ) [inline], [virtual]
```

##### Returns

Returns the port name from the selected API subsystem.

#### 13.78.3.23 get\_port\_count()

```
int seq64::rtmidi::get_port_count ( ) [inline]
```

##### Returns

This value depends on the MIDI mode setting (input versus output).

#### 13.78.3.24 full\_port\_count()

```
int seq64::rtmidi::full_port_count ( ) [inline]
```

##### Returns

This value is the sum of the number of input and output ports.

### 13.78.3.25 get\_api() [1/2]

```
const midi_api* seq64::rtmidi::get_api ( ) const [inline]
```

### 13.78.3.26 get\_api() [2/2]

```
midi_api* seq64::rtmidi::get_api ( ) [inline]
```

### 13.78.3.27 set\_api()

```
void seq64::rtmidi::set_api (
    midi_api * ma ) [inline], [protected]
```

### 13.78.3.28 delete\_api()

```
void seq64::rtmidi::delete_api ( ) [inline], [protected]
```

## 13.78.4 Friends And Related Function Documentation

### 13.78.4.1 midibus

```
friend class midibus [friend]
```

## 13.78.5 Field Documentation

### 13.78.5.1 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.

### 13.78.5.2 m\_midi\_api

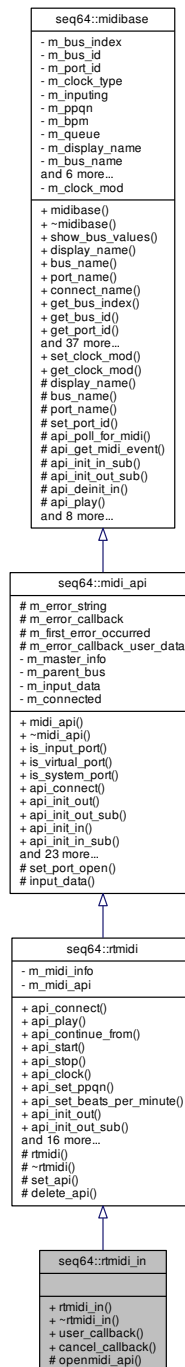
```
midi_api* seq64::rtmidi::m_midi_api [private]
```

[midi\\_alsa](#) or [midi\\_jack](#)) for which this class is a wrapper.

## 13.79 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)  
*Constructs the desired MIDI API.*
- virtual [~rtmidi\\_in](#) ()  
*A do-nothing virtual destructor.*
- 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)`

*Opens the desired MIDI API.*

## Additional Inherited Members

### 13.79.1 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.

### 13.79.2 Constructor & Destructor Documentation

#### 13.79.2.1 `rtmidi_in()`

```
seq64::rtmidi_in::rtmidi_in (
    midibus & parentbus,
    rtmidi_info & info )
```

If no compiled support for specified API value is found, we issue a warning and continue as if no API was specified. In this case, we iterate through the compiled APIs and return as soon as we find one with at least one port or we reach the end of the list.

#### Parameters

<i>parentbus</i>	This is the midibus that the <code>rtmidi</code> object is going to implement, by forwarding calls to the selected MIDI subsystem (e.g. ALSA or JACK).
<i>info</i>	Contains information about the existing ports and the selected MIDI API. Actually, the selected MIDI API is a static value of the <code>rtmidi_info</code> class.

#### Exceptions

<i>This</i>	function will throw an <code>rterror</code> object if it cannot find a MIDI API to use.
-------------	---



## 13.79.2.2 ~rtmidi\_in()

```
seq64::rtmidi_in::~~rtmidi_in ( ) [virtual]
```

## 13.79.3 Member Function Documentation

## 13.79.3.1 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.

## 13.79.3.2 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.

## 13.79.3.3 openmidi\_api()

```
void seq64::rtmidi_in::openmidi_api (
    rtmidi_api api,
    rtmidi_info & info ) [protected]
```

## Parameters

<i>api</i>	The enum value for the desired MIDI API. If not specified, first JACK is tried, then ALSA.
<i>info</i>	Holds information about the API's setup (e.g. a list of the ALSA ports found on the system, the main ALSA "handle" pointer, plus the PPQN, BPM, and other information).

## 13.80 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`

### 13.80.1 Detailed Description

Used to be nested in the [rtmidi\\_in](#) class.

### 13.80.2 Constructor & Destructor Documentation

#### 13.80.2.1 rtmidi\_in\_data()

```
seq64::rtmidi_in_data::rtmidi_in_data ( )
```

### 13.80.3 Member Function Documentation

#### 13.80.3.1 queue() [1/2]

```
const midi\_queue& seq64::rtmidi_in_data::queue ( ) const [inline]
```

#### 13.80.3.2 queue() [2/2]

```
midi\_queue& seq64::rtmidi_in_data::queue ( ) [inline]
```

#### 13.80.3.3 message() [1/2]

```
const midi\_message& seq64::rtmidi_in_data::message ( ) const [inline]
```

#### 13.80.3.4 message() [2/2]

```
midi\_message& seq64::rtmidi_in_data::message ( ) [inline]
```

#### 13.80.3.5 ignore\_flags() [1/2]

```
midibyte seq64::rtmidi_in_data::ignore_flags ( ) const [inline]
```

#### 13.80.3.6 test\_ignore\_flags()

```
bool seq64::rtmidi_in_data::test_ignore_flags (
    midibyte testbits ) [inline]
```

#### 13.80.3.7 ignore\_flags() [2/2]

```
void seq64::rtmidi_in_data::ignore_flags (
    midibyte setbits ) [inline]
```

#### 13.80.3.8 do\_input() [1/2]

```
bool seq64::rtmidi_in_data::do_input ( ) const [inline]
```

#### 13.80.3.9 do\_input() [2/2]

```
void seq64::rtmidi_in_data::do_input (
    bool flag ) [inline]
```

#### 13.80.3.10 first\_message() [1/2]

```
bool seq64::rtmidi_in_data::first_message ( ) const [inline]
```

#### 13.80.3.11 first\_message() [2/2]

```
void seq64::rtmidi_in_data::first_message (
    bool flag ) [inline]
```

#### 13.80.3.12 continue\_sysex() [1/2]

```
bool seq64::rtmidi_in_data::continue_sysex ( ) const [inline]
```

**13.80.3.13** continue\_sysex() [2/2]

```
void seq64::rtmidi_in_data::continue_sysex (
    bool flag ) [inline]
```

**13.80.3.14** using\_callback() [1/2]

```
bool seq64::rtmidi_in_data::using_callback ( ) const [inline]
```

**13.80.3.15** using\_callback() [2/2]

```
void seq64::rtmidi_in_data::using_callback (
    bool flag ) [inline]
```

**13.80.3.16** api\_data() [1/3]

```
const void* seq64::rtmidi_in_data::api_data ( ) const [inline]
```

**13.80.3.17** api\_data() [2/3]

```
void* seq64::rtmidi_in_data::api_data ( ) [inline]
```

**13.80.3.18** api\_data() [3/3]

```
void seq64::rtmidi_in_data::api_data (
    void * dataptr ) [inline]
```

**13.80.3.19** user\_data() [1/3]

```
const void* seq64::rtmidi_in_data::user_data ( ) const [inline]
```

**13.80.3.20 user\_data()** [2/3]

```
void* seq64::rtmidi_in_data::user_data ( ) [inline]
```

**13.80.3.21 user\_data()** [3/3]

```
void seq64::rtmidi_in_data::user_data (
    void * dataptr ) [inline]
```

**13.80.3.22 user\_callback()** [1/2]

```
rtmidi_callback_t seq64::rtmidi_in_data::user_callback ( ) const [inline]
```

**13.80.3.23 user\_callback()** [2/2]

```
void seq64::rtmidi_in_data::user_callback (
    rtmidi_callback_t cbptr ) [inline]
```

**13.80.4 Field Documentation****13.80.4.1 m\_queue**

```
midi_queue seq64::rtmidi_in_data::m_queue [private]
```

**13.80.4.2 m\_message**

```
midi_message seq64::rtmidi_in_data::m_message [private]
```

**13.80.4.3 m\_ignore\_flags**

```
midibyte seq64::rtmidi_in_data::m_ignore_flags [private]
```

#### 13.80.4.4 m\_do\_input

```
bool seq64::rtmidi_in_data::m_do_input [private]
```

#### 13.80.4.5 m\_first\_message

```
bool seq64::rtmidi_in_data::m_first_message [private]
```

#### 13.80.4.6 m\_api\_data

```
void* seq64::rtmidi_in_data::m_api_data [private]
```

#### 13.80.4.7 m\_using\_callback

```
bool seq64::rtmidi_in_data::m_using_callback [private]
```

#### 13.80.4.8 m\_user\_callback

```
rtmidi_callback_t seq64::rtmidi_in_data::m_user_callback [private]
```

#### 13.80.4.9 m\_user\_data

```
void* seq64::rtmidi_in_data::m_user_data [private]
```

#### 13.80.4.10 m\_continue\_sysex

```
bool seq64::rtmidi_in_data::m_continue_sysex [private]
```

## 13.81 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`)

*Default constructor.*

- virtual `~rtmidi_info` ()

*Destructor.*

- 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](#) ()  
*'Getter' function for member SEQ64\_RTmidi\_VERSION This is a static function to replace the [midi\\_api](#) version.*
- static void [get\\_compiled\\_api](#) (std::vector< [rtmidi\\_api](#) > &apis)  
*Gets the list of APIs compiled into the application.*
- 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)  
*Opens the desired MIDI API.*

## 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](#)

### 13.81.1 Detailed Description

New, but ripe for refactoring nonetheless.

## 13.81.2 Constructor & Destructor Documentation

### 13.81.2.1 `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 )
```

Code basically cut-and-paste from `rtmidi_in` or `rtmidi_out`. Common code!

### 13.81.2.2 `~rtmidi_info()`

```
seq64::rtmidi_info::~~rtmidi_info ( ) [virtual]
```

## 13.81.3 Member Function Documentation

### 13.81.3.1 `get_version()`

```
std::string seq64::rtmidi_info::get_version ( ) [static]
```

### 13.81.3.2 `get_compiled_api()`

```
void seq64::rtmidi_info::get_compiled_api (
    std::vector< rtmidi_api > & apis ) [static]
```

Note that we make ALSA versus JACK a runtime option as it is in the legacy Sequencer64 application.

This is a static function to replace the `midi_api` version.

#### Parameters

<code>apis</code>	The API structure.
-------------------	--------------------

### 13.81.3.3 `midi_mode()` [1/2]

```
bool seq64::rtmidi_info::midi_mode ( ) const [inline]
```

#### 13.81.3.4 midi\_mode() [2/2]

```
void seq64::rtmidi_info::midi_mode (
    bool flag ) [inline]
```

#### 13.81.3.5 clear()

```
void seq64::rtmidi_info::clear ( ) [inline]
```

#### 13.81.3.6 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.

#### 13.81.3.7 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.

#### 13.81.3.8 add\_bus()

```
void seq64::rtmidi_info::add_bus (
    const midibus * m ) [inline]
```

See the calls to this function in mastermidibus.

#### 13.81.3.9 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.

**13.81.3.10 get\_bus\_name()**

```
std::string seq64::rtmidi_info::get_bus_name (
    int index ) const [inline]
```

**13.81.3.11 get\_port\_count()**

```
int seq64::rtmidi_info::get_port_count ( ) const [inline]
```

**13.81.3.12 full\_port\_count()**

```
int seq64::rtmidi_info::full_port_count ( ) const [inline]
```

**13.81.3.13 get\_port\_id()**

```
int seq64::rtmidi_info::get_port_id (
    int index ) const [inline]
```

**13.81.3.14 get\_port\_name()**

```
std::string seq64::rtmidi_info::get_port_name (
    int index ) const [inline]
```

**13.81.3.15 get\_input()**

```
bool seq64::rtmidi_info::get_input (
    int index ) const [inline]
```

**13.81.3.16 get\_virtual()**

```
bool seq64::rtmidi_info::get_virtual (
    int index ) const [inline]
```

**13.81.3.17 get\_system()**

```
bool seq64::rtmidi_info::get_system (
    int index ) const [inline]
```

**13.81.3.18 get\_all\_port\_info()**

```
int seq64::rtmidi_info::get_all_port_info ( ) [inline]
```

**13.81.3.19 queue\_number()**

```
int seq64::rtmidi_info::queue_number (
    int index ) const [inline]
```

**13.81.3.20 app\_name()**

```
const std::string& seq64::rtmidi_info::app_name ( ) const [inline]
```

**13.81.3.21 global\_queue()**

```
int seq64::rtmidi_info::global_queue ( ) const [inline]
```

### 13.81.3.22 ppqn()

```
int seq64::rtmidi_info::ppqn ( ) const [inline]
```

### 13.81.3.23 api\_set\_ppqn()

```
void seq64::rtmidi_info::api_set_ppqn (
    int p ) [inline]
```

### 13.81.3.24 bpm()

```
midibpm seq64::rtmidi_info::bpm ( ) const [inline]
```

### 13.81.3.25 api\_set\_beats\_per\_minute()

```
void seq64::rtmidi_info::api_set_beats_per_minute (
    midibpm b ) [inline]
```

### 13.81.3.26 api\_port\_start()

```
void seq64::rtmidi_info::api_port_start (
    mastermidibus & masterbus,
    int bus,
    int port ) [inline]
```

### 13.81.3.27 api\_get\_midi\_event()

```
bool seq64::rtmidi_info::api_get_midi_event (
    event * inev ) [inline]
```

### 13.81.3.28 api\_flush()

```
void seq64::rtmidi_info::api_flush ( ) [inline]
```

### 13.81.3.29 api\_poll\_for\_midi()

```
int seq64::rtmidi_info::api_poll_for_midi ( ) [inline]
```

### 13.81.3.30 port\_list()

```
std::string seq64::rtmidi_info::port_list ( ) const [inline]
```

### 13.81.3.31 selected\_api() [1/2]

```
static rtmidi_api& seq64::rtmidi_info::selected_api ( ) [inline], [static]
```

### 13.81.3.32 get\_api\_info() [1/2]

```
const midi_info* seq64::rtmidi_info::get_api_info ( ) const [inline]
```

### 13.81.3.33 get\_api\_info() [2/2]

```
midi_info* seq64::rtmidi_info::get_api_info ( ) [inline]
```

### 13.81.3.34 api\_connect()

```
bool seq64::rtmidi_info::api_connect ( ) [inline], [protected]
```

### 13.81.3.35 selected\_api() [2/2]

```
static void seq64::rtmidi_info::selected_api (
    const rtmidi_api & api ) [inline], [static], [protected]
```

### 13.81.3.36 set\_api\_info()

```
bool seq64::rtmidi_info::set_api_info (
    midinfo * ma ) [inline], [protected]
```

This feature is important to allow a missing API (e.g. the JACK server is not running) to be detected.

### 13.81.3.37 delete\_api()

```
void seq64::rtmidi_info::delete_api ( ) [inline], [protected]
```

### 13.81.3.38 openmidi\_api()

```
bool seq64::rtmidi_info::openmidi_api (
    rtmidi_api api,
    const std::string & appname,
    int ppqn,
    midibpm bpm ) [protected]
```

If the JACK API is tried, and found missing, we turn off all of the other JACK flags found in the "rc" configuration file. Also, the loop in the constructor will come back here to try the other compiled-in APIs (currently just ALSA).

#### Parameters

<i>api</i>	The desired MIDI API.
<i>appname</i>	The name of the application, to be passed to the <i>midinfo</i> -derived constructor.
<i>ppqn</i>	The PPQN value to pass along to the <i>midinfo</i> -derived constructor.
<i>bpm</i>	The BPM (beats per minute) value to pass along to the <i>midinfo</i> -derived constructor.

#### Returns

Returns true if a valid API is found. A valid API is one that is both compiled into the application and is found existing on the host computer (system).

## 13.81.4 Friends And Related Function Documentation

### 13.81.4.1 mastermidibus

```
friend class mastermidibus [friend]
```



#### 13.81.4.2 midibus

```
friend class midibus [friend]
```

#### 13.81.4.3 rtmidi\_in

```
friend class rtmidi_in [friend]
```

#### 13.81.4.4 rtmidi\_out

```
friend class rtmidi_out [friend]
```

### 13.81.5 Field Documentation

#### 13.81.5.1 m\_info\_api

```
midi_info* seq64::rtmidi_info::m_info_api [private]
```

#### 13.81.5.2 sm\_selected\_api

```
rtmidi_api seq64::rtmidi_info::sm_selected_api [static], [private]
```

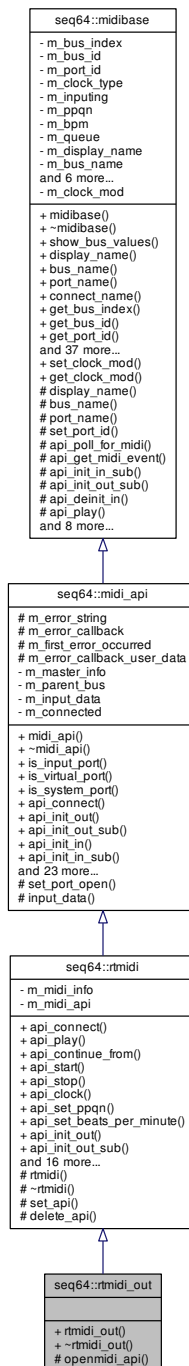
Holds the selected API code.

Its default value is RTMIDI\_API\_UNSPECIFIED.

## 13.82 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)

*Principal constructor.*

- 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

### 13.82.1 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.

### 13.82.2 Constructor & Destructor Documentation

#### 13.82.2.1 rtmidi\_out()

```
seq64::rtmidi_out::rtmidi_out (
    midibus & parentbus,
    rtmidi_info & info )
```

Attempt to open the specified API. If there is no compiled support for specified API value, then issue a warning and continue as if no API was specified. In that case, we iterate through the compiled APIs and return as soon as we find one with at least one port or we reach the end of the list.

#### Parameters

<i>parentbus</i>	This is the midibus that the rtmidi object is going to implement, by forwarding calls to the selected MIDI subsystem (e.g. ALSA or JACK).
<i>info</i>	Contains information about the existing ports and the selected MIDI API. Actually, the selected MIDI API is a static value of the <code>rtmidi_info</code> class.

#### Exceptions

<i>This</i>	function will throw an <code>rterror</code> object if it cannot find a MIDI API to use.
-------------	---

#### 13.82.2.2 ~rtmidi\_out()

```
seq64::rtmidi_out::~~rtmidi_out ( ) [virtual]
```

A do-nothing virtual destructor.

### 13.82.3 Member Function Documentation

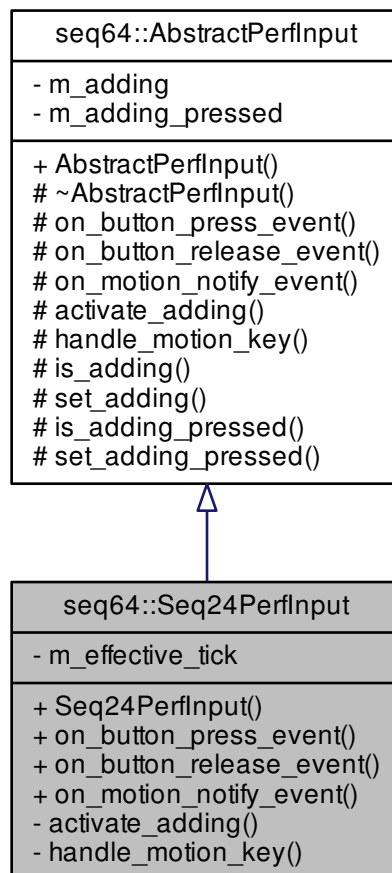
#### 13.82.3.1 openmidi\_api()

```
void seq64::rtmidi_out::openmidi_api (
    rtmidi_api api,
    rtmidi_info & info ) [protected]
```

### 13.83 seq64::Seq24PerfInput Class Reference

Implements the default (Seq24) performance input characteristics of this application.

Inheritance diagram for seq64::Seq24PerfInput:



## Public Member Functions

- [Seq24PerfInput](#) ()
- bool [on\\_button\\_press\\_event](#) (GdkEventButton \*a\_ev, [perfroll](#) &roll)  
*Handles the normal variety of button-press event.*
- bool [on\\_button\\_release\\_event](#) (GdkEventButton \*a\_ev, [perfroll](#) &roll)  
*Handles various button-release events.*
- bool [on\\_motion\\_notify\\_event](#) (GdkEventMotion \*a\_ev, [perfroll](#) &roll)  
*Handles the normal motion-notify event.*

## Private Member Functions

- virtual void [activate\\_adding](#) (bool a\_adding, [perfroll](#) &roll)  
*A popup menu (which one?) calls this.*
- bool [handle\\_motion\\_key](#) (bool is\_left, [perfroll](#) &roll)  
*Handles the keystroke motion-notify event for moving a pattern back and forth in the performance.*

## Private Attributes

- [midipulse m\\_effective\\_tick](#)  
*The current tick for the current segment?*

## Friends

- class [perfroll](#)

## Additional Inherited Members

### 13.83.1 Constructor & Destructor Documentation

#### 13.83.1.1 Seq24PerfInput()

```
seq64::Seq24PerfInput::Seq24PerfInput ( ) [inline]
```

### 13.83.2 Member Function Documentation

### 13.83.2.1 on\_button\_press\_event()

```
bool seq64::Seq24PerfInput::on_button_press_event (
    GdkEventButton * ev,
    perfroll & roll ) [virtual]
```

Is there any easy way to use ctrl-left-click as the middle button here?

Stazed:

roll.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.

#### Returns

Returns true if a modification occurred.

Implements [seq64::AbstractPerfInput](#).

### 13.83.2.2 on\_button\_release\_event()

```
bool seq64::Seq24PerfInput::on_button_release_event (
    GdkEventButton * ev,
    perfroll & roll ) [virtual]
```

Any use for the middle-button or ctrl-left-click we can add?

#### Returns

Returns true if any modification occurred.

Implements [seq64::AbstractPerfInput](#).

### 13.83.2.3 on\_motion\_notify\_event()

```
bool seq64::Seq24PerfInput::on_motion_notify_event (
    GdkEventMotion * ev,
    perfroll & roll ) [virtual]
```

#### Returns

Returns true if a modification occurs. This function used to always return true.

Implements [seq64::AbstractPerfInput](#).

## 13.83.2.4 activate\_adding()

```
void seq64::Seq24PerfInput::activate_adding (
    bool adding,
    perffroll & roll ) [private], [virtual]
```

What does it mean?

Implements [seq64::AbstractPerfInput](#).

## 13.83.2.5 handle\_motion\_key()

```
bool seq64::Seq24PerfInput::handle_motion_key (
    bool is_left,
    perffroll & roll ) [private], [virtual]
```

What happens when the mouse is used to drag the pattern is that, first, roll.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 roll.m\_moving = true, and calculates roll.m\_drop\_tick\_trigger\_offset = roll.m\_drop\_tick - p.get\_sequence(dropseq)->selected\_trigger\_start();

The motion handler sees that roll.m\_moving is true, gets the new tick value from the new x value, offsets it, and calls p.get\_sequence(dropseq)->move\_selected\_triggers\_to(tick, true).

When the user releases the left button, then roll.m\_growing is turned of and the roll draw\_all()'s.

## Parameters

<i>is_left</i>	False denotes the right arrow key, and true denotes the left arrow key.
<i>roll</i>	Provides a reference to the parent roll, which keeps track of most of the information about the status of the window.

## Returns

Returns true if there was some action able to happen that would necessitate a window update. We've updated [triggers::move\\_selected\(\)](#) [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::AbstractPerfInput](#).

## 13.83.3 Friends And Related Function Documentation

## 13.83.3.1 perffroll

```
friend class perffroll [friend]
```

### 13.83.4 Field Documentation

#### 13.83.4.1 m\_effective\_tick

```
midipulse seq64::Seq24PerfInput::m_effective_tick [private]
```

## 13.84 seq64::Seq24SeqEventInput Struct Reference

This structure implement the normal interaction methods for Seq24.

### Public Member Functions

- [Seq24SeqEventInput](#) ()  
*Default constructor.*
- void [set\\_adding](#) (bool adding, [sequevent](#) &ths)  
*Changes the mouse cursor to a pencil or a left pointer in the given sequevent object, depending on the first parameter.*
- bool [on\\_button\\_press\\_event](#) (GdkEventButton \*ev, [sequevent](#) &ths)  
*Implements the on-button-press event callback.*
- bool [on\\_button\\_release\\_event](#) (GdkEventButton \*ev, [sequevent](#) &ths)  
*Implements the on-button-release callback.*
- bool [on\\_motion\\_notify\\_event](#) (GdkEventMotion \*ev, [sequevent](#) &ths)  
*Implements the on-motion-notify event.*

### Data Fields

- bool [m\\_adding](#)  
*True if we're adding events via the mouse.*

### 13.84.1 Constructor & Destructor Documentation

#### 13.84.1.1 Seq24SeqEventInput()

```
seq64::Seq24SeqEventInput::Seq24SeqEventInput ( ) [inline]
```

### 13.84.2 Member Function Documentation

#### 13.84.2.1 set\_adding()

```
void seq64::Seq24SeqEventInput::set_adding (
    bool adding,
    sequevent & segev )
```

Modifies m\_adding as well.



## Parameters

<i>adding</i>	The value to set m_adding to, and if true, sets the mouse cursor to a pencil icon, otherwise sets it to a standard mouse-pointer icon.
<i>segev</i>	The sequevent whose window will be set to "adding" mode.

## 13.84.2.2 on\_button\_press\_event()

```
bool seq64::Seq24SeqEventInput::on_button_press_event (
    GdkEventButton * ev,
    sequevent & segev )
```

Set values for dragging, then reset the box that holds dirty redraw spot. Then do the rest.

## Parameters

<i>ev</i>	The button event for the press of a mouse button.
<i>segev</i>	Provides the sequevent strip to be affected by this button event.

## Returns

Returns true if a likely modification was made. This function used to return true all the time.

Needs update. segev.m\_seq.unselect(); ???????

## 13.84.2.3 on\_button\_release\_event()

```
bool seq64::Seq24SeqEventInput::on_button_release_event (
    GdkEventButton * ev,
    sequevent & segev )
```

## Parameters

<i>ev</i>	The button event for the release of a mouse button.
<i>segev</i>	Provides the sequevent strip to be affected by this button event.

## Returns

Returns true if a likely modification was made. This function used to return true all the time.

## 13.84.2.4 on\_motion\_notify\_event()

```
bool seq64::Seq24SeqEventInput::on_motion_notify_event (
    GdkEventMotion * ev,
    sequevent & segev )
```

**Parameters**

<i>ev</i>	The button event for the motion of the mouse cursor.
<i>segev</i>	Provides the sequevent strip to be affected by this button event.

**Returns**

Returns true if a likely modification was made. This function used to return true all the time.

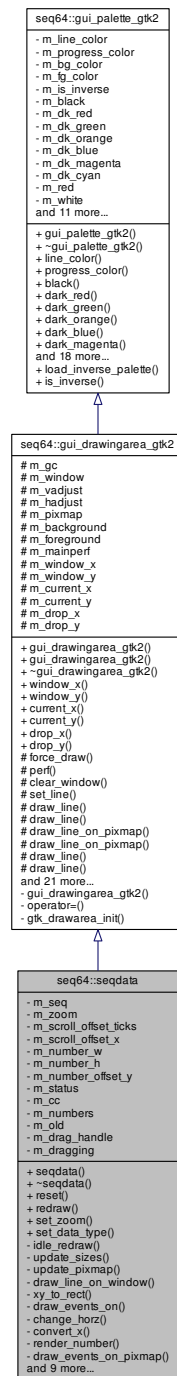
**13.84.3 Field Documentation****13.84.3.1 m\_adding**

```
bool seq64::Seq24SeqEventInput::m_adding
```

**13.85 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 `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.*
- Glib::RefPtr< Gdk::Pixmap > [m\\_numbers](#) [c\_dataarea\_y]  
*Holds the pixmaps for each number (0 to 127) that can be drawn for a data value in the data pane.*
- 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 [sequevent](#)
- class [seqroll](#)

## Additional Inherited Members

### 13.85.1 Constructor & Destructor Documentation

#### 13.85.1.1 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 <a href="#">perform::modify()</a> function.
<i>zoom</i>	The starting zoom of this pane.
<i>hadjust</i>	The horizontal adjustment object provided by the parent class, <a href="#">seqedit</a> , that created this pane.

13.85.1.2 [~seqdata\(\)](#)

```
virtual seq64::seqdata::~~seqdata ( ) [inline], [virtual]
```

## 13.85.2 Member Function Documentation

13.85.2.1 [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.

13.85.2.2 [redraw\(\)](#)

```
void seq64::seqdata::redraw ( ) [inline]
```

13.85.2.3 [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 <a href="#">seqedit::set_zoom()</a> function.
----------	--

#### 13.85.2.4 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.

#### 13.85.2.5 idle\_redraw()

```
int seq64::seqdata::idle_redraw ( ) [private]
```

This drawing is done only if there is no dragging in progress, to guarantee no flicker.

#### 13.85.2.6 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!

#### 13.85.2.7 update\_pixmap()

```
void seq64::seqdata::update_pixmap ( ) [private]
```

#### 13.85.2.8 draw\_line\_on\_window()

```
void seq64::seqdata::draw_line_on_window ( ) [private]
```

### 13.85.2.9 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.

## 13.85.2.10 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.
-----------------	----------------------------

## 13.85.2.11 change\_horz()

```
void seq64::seqdata::change_horz ( ) [private]
```

Basically identical to `seqevent::change_horz()`.

**13.85.2.12 convert\_x()**

```
void seq64::seqdata::convert_x (
    int x,
    midipulse & tick ) [inline], [private]
```

**13.85.2.13 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.

**13.85.2.14 draw\_events\_on\_pixmap()**

```
void seq64::seqdata::draw_events_on_pixmap ( ) [inline], [private]
```

**13.85.2.15 draw\_pixmap\_on\_window()**

```
void seq64::seqdata::draw_pixmap_on_window ( ) [inline], [private]
```

**13.85.2.16 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).

**13.85.2.17 on\_expose\_event()**

```
bool seq64::seqdata::on_expose_event (
    GdkEventExpose * ev ) [private]
```

**Parameters**

<i>ev</i>	Provides the expose-event.
-----------	----------------------------

**Returns**

Always returns true.

**13.85.2.18 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.

**13.85.2.19 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.

### 13.85.2.20 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

ev	The motion event.
----	-------------------

#### Returns

Returns true if a change in event data occurred. If true, then the perform modification flag is set.

### 13.85.2.21 on\_leave\_notify\_event()

```
bool seq64::seqdata::on_leave_notify_event (
    GdkEventCrossing * ev ) [private]
```

Parameter "p0", the crossing point for the event, is unused.

### 13.85.2.22 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

ev	Provides the scroll-event.
----	----------------------------

#### Returns

Always returns true.

**13.85.2.23 on\_size\_allocate()**

```
void seq64::seqdata::on_size_allocate (
    Gtk::Allocation & r ) [private]
```

**Parameters**

<i>r</i>	Provides the allocation event.
----------	--------------------------------

**13.85.3 Friends And Related Function Documentation****13.85.3.1 lfownd**

```
friend class lfownd [friend]
```

**13.85.3.2 sequevent**

```
friend class sequevent [friend]
```

**13.85.3.3 seqroll**

```
friend class seqroll [friend]
```

**13.85.4 Field Documentation****13.85.4.1 m\_seq**

```
sequence& seq64::seqdata::m_seq [private]
```

**13.85.4.2 m\_zoom**

```
int seq64::seqdata::m_zoom [private]
```

the unit is ticks/pixel.

#### 13.85.4.3 m\_scroll\_offset\_ticks

```
int seq64::seqdata::m_scroll_offset_ticks [private]
```

Adjusted in the [change\\_horz\(\)](#) function.

#### 13.85.4.4 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..

#### 13.85.4.5 m\_number\_w

```
int seq64::seqdata::m_number_w [private]
```

By "adjusted", well this is just a minor tweak for appearances.

#### 13.85.4.6 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.

#### 13.85.4.7 m\_number\_offset\_y

```
int seq64::seqdata::m_number_offset_y [private]
```

This value was hardwired as 8, for a character height of 10.

#### 13.85.4.8 m\_status

```
midibyte seq64::seqdata::m_status [private]
```

#### 13.85.4.9 m\_cc

```
midibyte seq64::seqdata::m_cc [private]
```

#### 13.85.4.10 m\_numbers

```
Glib::RefPtr<Gdk::Pixmap> seq64::seqdata::m_numbers[c_dataarea_y] [private]
```

This array is filled only once, in the [on\\_realize\(\)](#) function.

#### 13.85.4.11 m\_old

GdkRectangle seq64::seqdata::m\_old [private]

#### 13.85.4.12 m\_drag\_handle

bool seq64::seqdata::m\_drag\_handle [private]

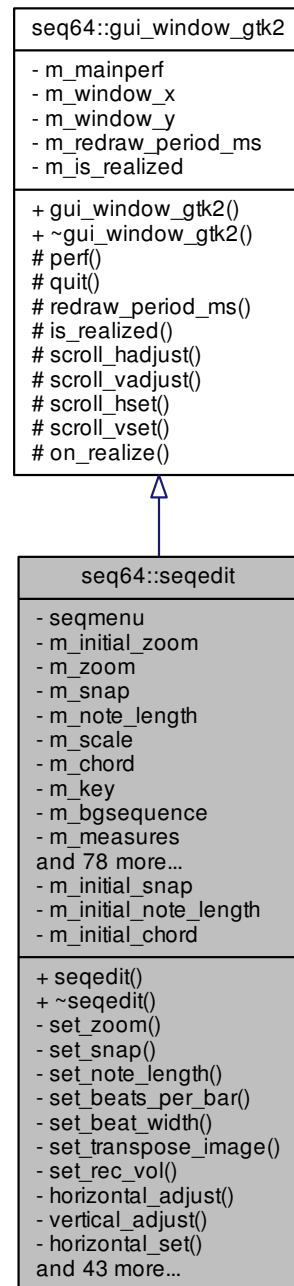
#### 13.85.4.13 m\_dragging

bool seq64::seqdata::m\_dragging [private]

## 13.86 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\\_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\\_rec\\_vol](#) (int recvol)  
*Passes the given parameter to [sequence::set\\_rec\\_vol\(\)](#).*
- 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.*
- void [set\\_measures](#) (int lim)  
*Set the measures value, using the given parameter, and some internal values passed to [apply\\_length\(\)](#).*
- void [apply\\_length](#) (midibpm bpm, int bw, int measures)  
*Sets the sequence length based on the three given parameters.*
- long [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\\_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 [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 [set\\_data\\_type](#) (midibyte status, midibyte control=0)  
*Sets the data type based on the given parameters.*
- 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 [popup\\_menu](#) (Gtk::Menu \*menu)  
*Pops up the given pop-up menu.*
- void [popup\\_event\\_menu](#) ()  
*Populates the event-selection menu that drops from the "Event" button in the bottom row of the Pattern editor.*
- 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.*
- 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\\_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::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::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::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\\_name](#)  
*Name of the sequence.*
- [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.*
- 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

#### 13.86.1 Detailed Description

- perform
- seqroll
- seqkeys
- seqdata
- seqtime
- seqevent
- sequence

This class has a metric ton of user-interface objects and other members.

## 13.86.2 Constructor & Destructor Documentation

### 13.86.2.1 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!

### 13.86.2.2 ~seqedit()

```
seq64::seqedit::~~seqedit ( ) [virtual]
```

## 13.86.3 Member Function Documentation

### 13.86.3.1 set\_zoom()

```
void seq64::seqedit::set_zoom (
    int z ) [private]
```

It is passed to the seqroll, seqtime, seqdata, and seqevent 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 <a href="#">usr().min_zoom()</a> and <a href="#">usr().max_zoom()</a> function.
----------	--

13.86.3.2 `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.
----------	--

13.86.3.3 `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.
-------------------	---

13.86.3.4 `set_beats_per_bar()`

```
void seq64::seqedit::set_beats_per_bar (
    int bpm ) [private]
```



**Todo** Check if verification is needed at this point.

#### Parameters

<i>bpm</i>	Provides the BPM (beats per measure) value to set.
------------	--

#### 13.86.3.5 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.
-----------	---------------------------------------

#### 13.86.3.6 set\_transpose\_image()

```
void seq64::seqedit::set_transpose_image (
    bool istransposable ) [private]
```

#### Parameters

<i>istransposable</i>	If true, set the image to the "Transpose" icon. Otherwise, set it to the "Drum" (not transposable) icon.
-----------------------	--

#### 13.86.3.7 set\_rec\_vol()

```
void seq64::seqedit::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.
---------------	--

### 13.86.3.8 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 <a href="#">gui_drawingarea_gtk2::scroll_hadjust()</a> for more information.
-------------	---

### 13.86.3.9 vertical\_adjust()

```
void seq64::segedit::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 <a href="#">gui_drawingarea_gtk2::scroll_vadjust()</a> for more information.
-------------	---

### 13.86.3.10 horizontal\_set()

```
void seq64::segedit::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).
--------------	---

### 13.86.3.11 vertical\_set()

```
void seq64::segedit::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).
--------------	---

## 13.86.3.12 set\_measures()

```
void seq64::seqedit::set_measures (
    int lim ) [private]
```

**Todo** Check if verification is needed at this point.

## Parameters

<i>lim</i>	Provides the sequence length, in measures.
------------	--

## 13.86.3.13 apply\_length()

```
void seq64::seqedit::apply_length (
    midibpm bpm,
    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().

## 13.86.3.14 get\_measures()

```
long seq64::seqedit::get_measures ( ) [private]
```

**Todo** Create a [sequence::set\\_units\(\)](#) function or a [sequence::get\\_measures\(\)](#) function to forward to.

## 13.86.3.15 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.

13.86.3.16 `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.
<i>user_change</i>	True if the user made this change, and thus has potentially modified the song.

13.86.3.17 `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()`.

13.86.3.18 `set_chord()`

```
void seq64::seqedit::set_chord (
    int chord ) [private]
```

13.86.3.19 `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()`.

### 13.86.3.20 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\(\)](#).

### 13.86.3.21 transpose\_change\_callback()

```
void seq64::seqedit::transpose_change_callback ( ) [private]
```

### 13.86.3.22 name\_change\_callback()

```
void seq64::seqedit::name_change_callback ( ) [private]
```

That name is the name the user has given to the sequence being edited.

### 13.86.3.23 play\_change\_callback()

```
void seq64::seqedit::play_change_callback ( ) [private]
```

### 13.86.3.24 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.
```

### 13.86.3.25 q\_rec\_change\_callback()

```
void seq64::seqedit::q_rec_change_callback ( ) [private]
```

**13.86.3.26 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.

**13.86.3.27 undo\_callback()**

```
void seq64::seqedit::undo_callback ( ) [private]
```

**13.86.3.28 redo\_callback()**

```
void seq64::seqedit::redo_callback ( ) [private]
```

**13.86.3.29 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!

**13.86.3.30 update\_all\_windows()**

```
void seq64::seqedit::update_all_windows ( ) [private]
```

**13.86.3.31 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.

**13.86.3.32 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 *perfed*.

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").

**13.86.3.33 popup\_menu()**

```
void seq64::seqedit::popup_menu (
    Gtk::Menu * menu ) [private]
```

**13.86.3.34 popup\_event\_menu()**

```
void seq64::seqedit::popup_event_menu ( ) [private]
```

This menu has a large number of items. I think they are filled in in code, but can also be loaded from `~/seq24usr`. To be determined. Create the 8 sub-menus for the various ranges of controller changes, shown 16 per sub-menu.

**13.86.3.35 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.

**13.86.3.36 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.

**13.86.3.37 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.

**13.86.3.38 popup\_midich\_menu()**

```
void seq64::seqedit::popup_midich_menu ( ) [private]
```

**13.86.3.39 create\_menu\_image()**

```
Gtk::Image * seq64::seqedit::create_menu_image (
    bool state = false ) [private]
```

**13.86.3.40 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.



#### 13.86.3.41 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).

#### 13.86.3.42 mouse\_action()

```
void seq64::seqedit::mouse_action (
    mouse_action_e action ) [private]
```

#### 13.86.3.43 start\_playing()

```
void seq64::seqedit::start_playing ( ) [private]
```

#### 13.86.3.44 stop\_playing()

```
void seq64::seqedit::stop_playing ( ) [private]
```

#### 13.86.3.45 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.
---------------------------	---

#### 13.86.3.46 handle\_close()

```
void seq64::seqedit::handle_close ( ) [private]
```

#### 13.86.3.47 on\_realize()

```
void seq64::seqedit::on_realize ( ) [private]
```

#### 13.86.3.48 on\_set\_focus()

```
void seq64::seqedit::on_set_focus (
    Widget * focus ) [private]
```

Only works in certain circumstances.

#### 13.86.3.49 on\_focus\_in\_event()

```
bool seq64::seqedit::on_focus_in_event (
    GdkEventFocus * ) [private]
```

#### 13.86.3.50 on\_focus\_out\_event()

```
bool seq64::seqedit::on_focus_out_event (
    GdkEventFocus * ) [private]
```

#### 13.86.3.51 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.

## 13.86.3.52 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.

## 13.86.3.53 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

<code>ev</code>	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)`.

**13.86.4 Field Documentation****13.86.4.1 seqmenu**

```
friend seq64::seqedit::seqmenu [private]
```

**13.86.4.2 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.

**13.86.4.3 m\_initial\_note\_length**

```
int seq64::seqedit::m_initial_note_length [static], [private]
```

**13.86.4.4 m\_initial\_chord**

```
int seq64::seqedit::m_initial_chord [static], [private]
```

#### 13.86.4.5 m\_initial\_zoom

```
const int seq64::seqedit::m_initial_zoom [private]
```

#### 13.86.4.6 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.

#### 13.86.4.7 m\_snap

```
int seq64::seqedit::m_snap [private]
```

#### 13.86.4.8 m\_note\_length

```
int seq64::seqedit::m_note_length [private]
```

#### 13.86.4.9 m\_scale

```
int seq64::seqedit::m_scale [private]
```

#### 13.86.4.10 m\_chord

```
int seq64::seqedit::m_chord [private]
```

#### 13.86.4.11 m\_key

```
int seq64::seqedit::m_key [private]
```

#### 13.86.4.12 m\_bgsequence

```
int seq64::seqedit::m_bgsequence [private]
```

#### 13.86.4.13 m\_measures

```
long seq64::seqedit::m_measures [private]
```

#### 13.86.4.14 m\_ppqn

```
int seq64::seqedit::m_ppqn [private]
```

#### 13.86.4.15 m\_pp\_whole

```
int seq64::seqedit::m_pp_whole [private]
```

#### 13.86.4.16 m\_pp\_eighth

```
int seq64::seqedit::m_pp_eighth [private]
```

#### 13.86.4.17 m\_pp\_sixteenth

```
int seq64::seqedit::m_pp_sixteenth [private]
```

#### 13.86.4.18 m\_seq

```
sequence& seq64::seqedit::m_seq [private]
```

#### 13.86.4.19 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.

#### 13.86.4.20 m\_menu\_tools

Gtk::Menu\* seq64::seqedit::m\_menu\_tools [private]

#### 13.86.4.21 m\_menu\_zoom

Gtk::Menu\* seq64::seqedit::m\_menu\_zoom [private]

#### 13.86.4.22 m\_menu\_snap

Gtk::Menu\* seq64::seqedit::m\_menu\_snap [private]

#### 13.86.4.23 m\_menu\_note\_length

Gtk::Menu\* seq64::seqedit::m\_menu\_note\_length [private]

#### 13.86.4.24 m\_menu\_length

Gtk::Menu\* seq64::seqedit::m\_menu\_length [private]

#### 13.86.4.25 m\_toggle\_transpose

Gtk::ToggleButton\* seq64::seqedit::m\_toggle\_transpose [private]

#### 13.86.4.26 m\_image\_transpose

Gtk::Image\* seq64::seqedit::m\_image\_transpose [private]

#### 13.86.4.27 m\_menu\_midich

Gtk::Menu\* seq64::seqedit::m\_menu\_midich [private]

**13.86.4.28 m\_menu\_midibus**

Gtk::Menu\* seq64::seqedit::m\_menu\_midibus [private]

**13.86.4.29 m\_menu\_data**

Gtk::Menu\* seq64::seqedit::m\_menu\_data [private]

**13.86.4.30 m\_menu\_key**

Gtk::Menu\* seq64::seqedit::m\_menu\_key [private]

**13.86.4.31 m\_menu\_scale**

Gtk::Menu\* seq64::seqedit::m\_menu\_scale [private]

**13.86.4.32 m\_menu\_chords**

Gtk::Menu\* seq64::seqedit::m\_menu\_chords [private]

**13.86.4.33 m\_menu\_sequences**

Gtk::Menu\* seq64::seqedit::m\_menu\_sequences [private]

**13.86.4.34 m\_menu\_bpm**

Gtk::Menu\* seq64::seqedit::m\_menu\_bpm [private]

**13.86.4.35 m\_menu\_bw**

Gtk::Menu\* seq64::seqedit::m\_menu\_bw [private]



#### 13.86.4.36 m\_menu\_rec\_vol

Gtk::Menu\* seq64::seqedit::m\_menu\_rec\_vol [private]

#### 13.86.4.37 m\_vadjust

Gtk::Adjustment\* seq64::seqedit::m\_vadjust [private]

Vertical position descriptor.

#### 13.86.4.38 m\_hadjust

Gtk::Adjustment\* seq64::seqedit::m\_hadjust [private]

#### 13.86.4.39 m\_vscroll\_new

Gtk::VScrollbar\* seq64::seqedit::m\_vscroll\_new [private]

#### 13.86.4.40 m\_hscroll\_new

Gtk::HScrollbar\* seq64::seqedit::m\_hscroll\_new [private]

#### 13.86.4.41 m\_seqkeys\_wid

seqkeys\* seq64::seqedit::m\_seqkeys\_wid [private]

This item draws the piano-keys at the left of the seqedit window.

#### 13.86.4.42 m\_seqtime\_wid

seqtime\* seq64::seqedit::m\_seqtime\_wid [private]

This is the location where the measure numbers and the END marker are shown.

#### 13.86.4.43 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.

#### 13.86.4.44 m\_seqevent\_wid

```
seqevent* seq64::seqedit::m_seqevent_wid [private]
```

#### 13.86.4.45 m\_seqroll\_wid

```
seqroll* seq64::seqedit::m_seqroll_wid [private]
```

#### 13.86.4.46 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.

#### 13.86.4.47 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.

#### 13.86.4.48 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.

#### 13.86.4.49 m\_vbox

```
Gtk::VBox* seq64::seqedit::m_vbox [private]
```

#### 13.86.4.50 m\_hbox

```
Gtk::HBox* seq64::seqedit::m_hbox [private]
```

#### 13.86.4.51 m\_hbox2

```
Gtk::HBox* seq64::seqedit::m_hbox2 [private]
```

#### 13.86.4.52 m\_button\_undo

Gtk::Button\* seq64::seqedit::m\_button\_undo [private]

#### 13.86.4.53 m\_button\_redo

Gtk::Button\* seq64::seqedit::m\_button\_redo [private]

#### 13.86.4.54 m\_button\_quantize

Gtk::Button\* seq64::seqedit::m\_button\_quantize [private]

#### 13.86.4.55 m\_button\_tools

Gtk::Button\* seq64::seqedit::m\_button\_tools [private]

#### 13.86.4.56 m\_button\_sequence

Gtk::Button\* seq64::seqedit::m\_button\_sequence [private]

#### 13.86.4.57 m\_entry\_sequence

Gtk::Entry\* seq64::seqedit::m\_entry\_sequence [private]

#### 13.86.4.58 m\_button\_bus

Gtk::Button\* seq64::seqedit::m\_button\_bus [private]

#### 13.86.4.59 m\_entry\_bus

Gtk::Entry\* seq64::seqedit::m\_entry\_bus [private]

**13.86.4.60 m\_button\_channel**

```
Gtk::Button* seq64::seqedit::m_button_channel [private]
```

**13.86.4.61 m\_entry\_channel**

```
Gtk::Entry* seq64::seqedit::m_entry_channel [private]
```

**13.86.4.62 m\_button\_snap**

```
Gtk::Button* seq64::seqedit::m_button_snap [private]
```

**13.86.4.63 m\_entry\_snap**

```
Gtk::Entry* seq64::seqedit::m_entry_snap [private]
```

**13.86.4.64 m\_button\_note\_length**

```
Gtk::Button* seq64::seqedit::m_button_note_length [private]
```

**13.86.4.65 m\_entry\_note\_length**

```
Gtk::Entry* seq64::seqedit::m_entry_note_length [private]
```

**13.86.4.66 m\_button\_zoom**

```
Gtk::Button* seq64::seqedit::m_button_zoom [private]
```

**13.86.4.67 m\_entry\_zoom**

```
Gtk::Entry* seq64::seqedit::m_entry_zoom [private]
```

**13.86.4.68 m\_button\_length**

Gtk::Button\* seq64::seqedit::m\_button\_length [private]

**13.86.4.69 m\_entry\_length**

Gtk::Entry\* seq64::seqedit::m\_entry\_length [private]

**13.86.4.70 m\_button\_key**

Gtk::Button\* seq64::seqedit::m\_button\_key [private]

**13.86.4.71 m\_entry\_key**

Gtk::Entry\* seq64::seqedit::m\_entry\_key [private]

**13.86.4.72 m\_button\_scale**

Gtk::Button\* seq64::seqedit::m\_button\_scale [private]

**13.86.4.73 m\_entry\_scale**

Gtk::Entry\* seq64::seqedit::m\_entry\_scale [private]

**13.86.4.74 m\_button\_chord**

Gtk::Button\* seq64::seqedit::m\_button\_chord [private]

**13.86.4.75 m\_entry\_chord**

Gtk::Entry\* seq64::seqedit::m\_entry\_chord [private]

**13.86.4.76 m\_tooltips**

Gtk::Tooltips\* seq64::seqedit::m\_tooltips [private]

**13.86.4.77 m\_button\_data**

Gtk::Button\* seq64::seqedit::m\_button\_data [private]

**13.86.4.78 m\_entry\_data**

Gtk::Entry\* seq64::seqedit::m\_entry\_data [private]

**13.86.4.79 m\_button\_bpm**

Gtk::Button\* seq64::seqedit::m\_button\_bpm [private]

**13.86.4.80 m\_entry\_bpm**

Gtk::Entry\* seq64::seqedit::m\_entry\_bpm [private]

**13.86.4.81 m\_button\_bw**

Gtk::Button\* seq64::seqedit::m\_button\_bw [private]

**13.86.4.82 m\_entry\_bw**

Gtk::Entry\* seq64::seqedit::m\_entry\_bw [private]

**13.86.4.83 m\_button\_rec\_vol**

Gtk::Button\* seq64::seqedit::m\_button\_rec\_vol [private]

**13.86.4.84 m\_toggle\_play**

Gtk::ToggleButton\* seq64::seqedit::m\_toggle\_play [private]

**13.86.4.85 m\_toggle\_record**

Gtk::ToggleButton\* seq64::seqedit::m\_toggle\_record [private]

**13.86.4.86 m\_toggle\_q\_rec**

Gtk::ToggleButton\* seq64::seqedit::m\_toggle\_q\_rec [private]

**13.86.4.87 m\_toggle\_thru**

Gtk::ToggleButton\* seq64::seqedit::m\_toggle\_thru [private]

**13.86.4.88 m\_entry\_name**

Gtk::Entry\* seq64::seqedit::m\_entry\_name [private]

**13.86.4.89 m\_editing\_status**

midibyte seq64::seqedit::m\_editing\_status [private]

**13.86.4.90 m\_editing\_cc**

midibyte seq64::seqedit::m\_editing\_cc [private]

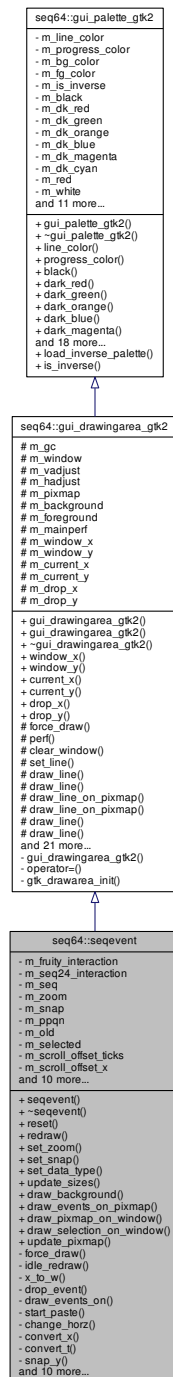
**13.86.4.91 m\_have\_focus**

bool seq64::seqedit::m\_have\_focus [private]

## 13.87 seq64::sequest Class Reference

Implements the piano event drawing area.

Inheritance diagram for seq64::sequest:



### Public Member Functions

- [sequest](#) (perform &p, [sequence](#) &seq, int zoom, int snap, [seqdata](#) &seqdata\_wid, Gtk::Adjustment &hadjust, int ppqn=SEQ64\_USE\_DEFAULT\_PPQN)



- Principal constructor.*

  - virtual `~sequevent ()`

*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_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 window dimensions, the 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.*

## Private 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)`

*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.*
- void [on\\_realize](#) ()  
*Implements the on-realize callback.*
- bool [on\\_expose\\_event](#) (GdkEventExpose \*ev)  
*Implements the on-expose event callback.*
- 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 \*ev)  
*Implements the on-motion-notify event callback.*
- bool [on\\_focus\\_in\\_event](#) (GdkEventFocus \*)  
*Responds to a focus event by setting the HAS\_FOCUS flag.*
- bool [on\\_focus\\_out\\_event](#) (GdkEventFocus \*)  
*Responds to a unfocus event by resetting the HAS\_FOCUS flag.*
- bool [on\\_key\\_press\\_event](#) (GdkEventKey \*p0)  
*Implements the key-press event callback function.*
- void [on\\_size\\_allocate](#) (Gtk::Allocation &)  
*Implements the on-size-allocate event callback.*

## Private Attributes

- [FruitySeqEventInput m\\_fruity\\_interaction](#)  
*Provides the mouse-handling paradigm for the fruity interaction.*
- [Seq24SeqEventInput m\\_seq24\\_interaction](#)  
*Provides the normal mouse-handling for Sequencer64.*
- [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.*
- int [m\\_ppqn](#)  
*The value to use for the PPQN for this sequence.*
- 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\\_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.*

## Friends

- struct [FruitySeqEventInput](#)
- struct [Seq24SeqEventInput](#)

## Additional Inherited Members

### 13.87.1 Constructor & Destructor Documentation

#### 13.87.1.1 sequest()

```
seq64::sequest::sequest (
    perform & p,
    sequence & seq,
    int zoom,
    int snap,
    seqdata & seqdata_wid,
    Gtk::Adjustment & hadjust,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

#### 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.
<i>ppqn</i>	The initial PPQN value.

**13.87.1.2 ~segevent()**

```
virtual seq64::segevent::~~segevent ( ) [inline], [virtual]
```

**13.87.2 Member Function Documentation****13.87.2.1 reset()**

```
void seq64::segevent::reset ( )
```

Basically identical to [seqtime::reset\(\)](#).

**13.87.2.2 redraw()**

```
void seq64::segevent::redraw ( )
```

Somewhat similar to [seqroll::redraw\(\)](#).

**13.87.2.3 set\_zoom()**

```
void seq64::segevent::set_zoom (
    int z )
```

**Parameters**

<b>z</b>	The desired zoom value, assumed to be validated already. See the <a href="#">seqedit::set_zoom()</a> function.
----------	--

**13.87.2.4 set\_snap()**

```
void seq64::segevent::set_snap (
    int snap ) [inline]
```

The parameter is not validated.

**13.87.2.5 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.

**13.87.2.6 update\_sizes()**

```
void seq64::segevent::update_sizes ( )
```

This ends up filling the background with dotted lines, etc.

**13.87.2.7 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.

**13.87.2.8 draw\_events\_on\_pixmap()**

```
void seq64::segevent::draw_events_on_pixmap ( )
```

**13.87.2.9 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.
```

**13.87.2.10 draw\_selection\_on\_window()**

```
void seq64::segevent::draw_selection_on_window ( )
```

### 13.87.2.11 update\_pixmap()

```
void seq64::segevent::update_pixmap ( )
```

### 13.87.2.12 force\_draw()

```
void seq64::segevent::force_draw ( ) [private], [virtual]
```

Reimplemented from [seq64::gui\\_drawingarea\\_gtk2](#).

### 13.87.2.13 idle\_redraw()

```
int seq64::segevent::idle_redraw ( ) [private]
```

Who calls this routine? Probably the default timer routine, but not sure.

#### Returns

Always returns true.

### 13.87.2.14 x\_to\_w()

```
void seq64::segevent::x_to_w (
    int x1,
    int x2,
    int & x,
    int & w ) [private]
```

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.

### 13.87.2.15 drop\_event()

```
void seq64::segevent::drop_event (
    midipulse tick ) [private]
```

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.
-------------	--

#### 13.87.2.16 draw\_events\_on()

```
void seq64::segevent::draw_events_on (
    Glib::RefPtr< Gdk::Drawable > drawable ) [private]
```

Very similar to [seqdata::draw\\_events\\_on\(\)](#).

#### Parameters

<i>drawable</i>	The given drawable object.
-----------------	----------------------------

#### 13.87.2.17 start\_paste()

```
void seq64::segevent::start_paste ( ) [private]
```

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.

#### 13.87.2.18 change\_horz()

```
void seq64::segevent::change_horz ( ) [private]
```

Very similar to [seqroll::change\\_horz\(\)](#). Basically identical to [seqdata::change\\_horz\(\)](#).

#### 13.87.2.19 convert\_x()

```
void seq64::segevent::convert_x (
    int x,
    midipulse & tick ) [inline], [private]
```

Why not just return it normally?

#### Parameters

	<i>x</i>	The x (pixel) value to convert.
<i>out</i>	<i>tick</i>	The destination for the converted x value.

**13.87.2.20 convert\_t()**

```
void seq64::segevent::convert_t (
    midipulse tick,
    int & x ) [inline], [private]
```

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.

**13.87.2.21 snap\_y()**

```
void seq64::segevent::snap_y (
    int & y ) [inline], [private]
```

**Parameters**

out	<i>y</i>	The return parameter for the conversion. Why not just return the value?
-----	----------	---

**13.87.2.22 snap\_x()**

```
void seq64::segevent::snap_x (
    int & x ) [private]
```

- snap = number pulses to snap to
- m\_zoom = number of pulses per pixel
- Therefore snap / m\_zoom = number of pixels to snap to.

**Parameters**

out	<i>x</i>	The output destination for the snapped x value.
-----	----------	---



### 13.87.2.23 on\_realize()

```
void seq64::segevent::on_realize ( ) [private]
```

It calls the base-class version, and then allocates additional resource not allocated in the constructor. Finally, it connects up the change\_horz function.

### 13.87.2.24 on\_expose\_event()

```
bool seq64::segevent::on_expose_event (
    GdkEventExpose * ev ) [private]
```

#### Parameters

<i>ev</i>	The expose event.
-----------	-------------------

### 13.87.2.25 on\_button\_press\_event()

```
bool seq64::segevent::on_button_press_event (
    GdkEventButton * ev ) [private]
```

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

<i>ev</i>	The button event.
-----------	-------------------

#### Returns

Returns true if the button-press was handled.

### 13.87.2.26 on\_button\_release\_event()

```
bool seq64::segevent::on_button_release_event (
    GdkEventButton * ev ) [private]
```

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.

**13.87.2.27 on\_motion\_notify\_event()**

```
bool seq64::segevent::on_motion_notify_event (
    GdkEventMotion * ev ) [private]
```

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.

**13.87.2.28 on\_focus\_in\_event()**

```
bool seq64::segevent::on_focus_in_event (
    GdkEventFocus * ) [private]
```

Parameter "ev" is the focus event, unused.

**Returns**

Always returns false.

**13.87.2.29 on\_focus\_out\_event()**

```
bool seq64::segevent::on_focus_out_event (
    GdkEventFocus * ) [private]
```

Parameter "ev" is the focus event, unused.

**Returns**

Always returns false.

## 13.87.2.30 on\_key\_press\_event()

```
bool seq64::segevent::on_key_press_event (
    GdkEventKey * ev ) [private]
```

It handles deleted 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

<i>ev</i>	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.

## 13.87.2.31 on\_size\_allocate()

```
void seq64::segevent::on_size_allocate (
    Gtk::Allocation & a ) [private]
```

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.
----------	---------------------------------

## 13.87.3 Friends And Related Function Documentation

## 13.87.3.1 FruitySeqEventInput

```
friend struct FruitySeqEventInput [friend]
```

## 13.87.3.2 Seq24SeqEventInput

```
friend struct Seq24SeqEventInput [friend]
```

## 13.87.4 Field Documentation

### 13.87.4.1 m\_fruity\_interaction

```
FruitySeqEventInput seq64::segevent::m_fruity_interaction [private]
```

Why should we need both at the same time? Just load the one that is specified in the configuration.

### 13.87.4.2 m\_seq24\_interaction

```
Seq24SeqEventInput seq64::segevent::m_seq24_interaction [private]
```

### 13.87.4.3 m\_seq

```
sequence& seq64::segevent::m_seq [private]
```

### 13.87.4.4 m\_zoom

```
int seq64::segevent::m_zoom [private]
```

### 13.87.4.5 m\_snap

```
int seq64::segevent::m_snap [private]
```

Same meaning as for the piano roll. This value is the denominator of the note size used for the snap.

### 13.87.4.6 m\_ppqn

```
int seq64::segevent::m_ppqn [private]
```

Used in snap and zoom scaling.

### 13.87.4.7 m\_old

```
GdkRectangle seq64::segevent::m_old [private]
```

#### 13.87.4.8 m\_selected

```
GdkRectangle seq64::segevent::m_selected [private]
```

#### 13.87.4.9 m\_scroll\_offset\_ticks

```
int seq64::segevent::m_scroll_offset_ticks [private]
```

#### 13.87.4.10 m\_scroll\_offset\_x

```
int seq64::segevent::m_scroll_offset_x [private]
```

Set to m\_scroll\_offset\_ticks divided by m\_zoom.

#### 13.87.4.11 m\_seqdata\_wid

```
seqdata& seq64::segevent::m_seqdata_wid [private]
```

#### 13.87.4.12 m\_selecting

```
bool seq64::segevent::m_selecting [private]
```

#### 13.87.4.13 m\_moving\_init

```
bool seq64::segevent::m_moving_init [private]
```

#### 13.87.4.14 m\_moving

```
bool seq64::segevent::m_moving [private]
```

WARNING: This operation seems to have a bug. It makes the events very very long. This bug exists in Seq24.

#### 13.87.4.15 m\_growing

```
bool seq64::segevent::m_growing [private]
```

Does growing work in this view? Need to do some better testing.

#### 13.87.4.16 m\_painting

```
bool seq64::segevent::m_painting [private]
```

#### 13.87.4.17 m\_paste

```
bool seq64::segevent::m_paste [private]
```

#### 13.87.4.18 m\_move\_snap\_offset\_x

```
int seq64::segevent::m_move_snap_offset_x [private]
```

#### 13.87.4.19 m\_status

```
midibyte seq64::segevent::m_status [private]
```

The current status/event byte.

#### 13.87.4.20 m\_cc

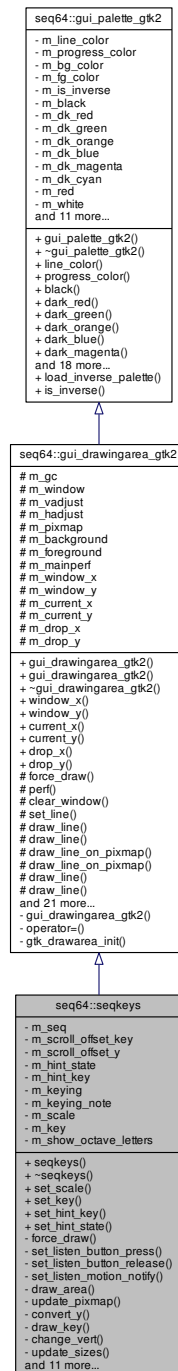
```
midibyte seq64::segevent::m_cc [private]
```

The current MIDI CC value.

### 13.88 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 &note)*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

### 13.88.1 Detailed Description

Note the friends of this class, [seqroll](#) and [FruitySeqRollInput](#). Where is Seq24SeqRollInput? Gone. It has been folded back into [seqroll](#).

### 13.88.2 Constructor & Destructor Documentation

#### 13.88.2.1 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.

## 13.88.2.2 ~seqkeys()

```
virtual seq64::seqkeys::~seqkeys ( ) [inline], [virtual]
```

## 13.88.3 Member Function Documentation

## 13.88.3.1 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.
--------------	------------------------------------

## 13.88.3.2 set\_key()

```
void seq64::seqkeys::set_key (
    int key )
```

## Parameters

<i>key</i>	The musical key value to be set.
------------	----------------------------------

## 13.88.3.3 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.
------------	---------------------------------------

## 13.88.3.4 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.
--------------	---

## 13.88.3.5 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](#).

## 13.88.3.6 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.
-----------	---

## 13.88.3.7 set\_listen\_button\_release()

```
void seq64::seqkeys::set_listen_button_release (
    GdkEventButton * ev ) [inline], [private]
```

**13.88.3.8 set\_listen\_motion\_notify()**

```
void seq64::seqkeys::set_listen_motion_notify (
    GdkEventMotion * ev ) [inline], [private]
```

**13.88.3.9 draw\_area()**

```
void seq64::seqkeys::draw_area ( ) [private]
```

**13.88.3.10 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.

**13.88.3.11 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.

**13.88.3.12 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.

**13.88.3.13** 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();
```

**13.88.3.14** update\_sizes()

```
void seq64::seqkeys::update_sizes ( ) [private]
```

**13.88.3.15** reset()

```
void seq64::seqkeys::reset ( ) [private]
```

**13.88.3.16** 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).

### 13.88.3.17 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.

### 13.88.3.18 on\_expose\_event()

```
bool seq64::seqkeys::on_expose_event (
    GdkEventExpose * ev ) [private]
```

#### Parameters

<i>ev</i>	The expose-event object.
-----------	--------------------------

### 13.88.3.19 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.

### 13.88.3.20 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.

#### 13.88.3.21 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.

#### 13.88.3.22 on\_enter\_notify\_event()

```
bool seq64::seqkeys::on_enter_notify_event (
    GdkEventCrossing * p0 ) [private]
```

This greys the current key.

#### 13.88.3.23 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.

#### 13.88.3.24 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.

**13.88.3.25 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.
------------	--

**13.88.4 Friends And Related Function Documentation****13.88.4.1 seqroll**

```
friend class seqroll [friend]
```

**13.88.4.2 FruitySeqRollInput**

```
friend class FruitySeqRollInput [friend]
```

**13.88.5 Field Documentation****13.88.5.1 m\_seq**

```
sequence& seq64::seqkeys::m_seq [private]
```



#### 13.88.5.2 m\_scroll\_offset\_key

```
int seq64::seqkeys::m_scroll_offset_key [private]
```

Modified in [change\\_vert\(\)](#).

#### 13.88.5.3 m\_scroll\_offset\_y

```
int seq64::seqkeys::m_scroll_offset_y [private]
```

Modified in [change\\_vert\(\)](#).

#### 13.88.5.4 m\_hint\_state

```
bool seq64::seqkeys::m_hint_state [private]
```

#### 13.88.5.5 m\_hint\_key

```
int seq64::seqkeys::m_hint_key [private]
```

#### 13.88.5.6 m\_keying

```
bool seq64::seqkeys::m_keying [private]
```

Used in playing the sound for each note as it is clicked in the seqkeys pane.

#### 13.88.5.7 m\_keying\_note

```
int seq64::seqkeys::m_keying_note [private]
```

#### 13.88.5.8 m\_scale

```
int seq64::seqkeys::m_scale [private]
```

#### 13.88.5.9 m\_key

```
int seq64::seqkeys::m_key [private]
```

#### 13.88.5.10 m\_show\_octave\_letters

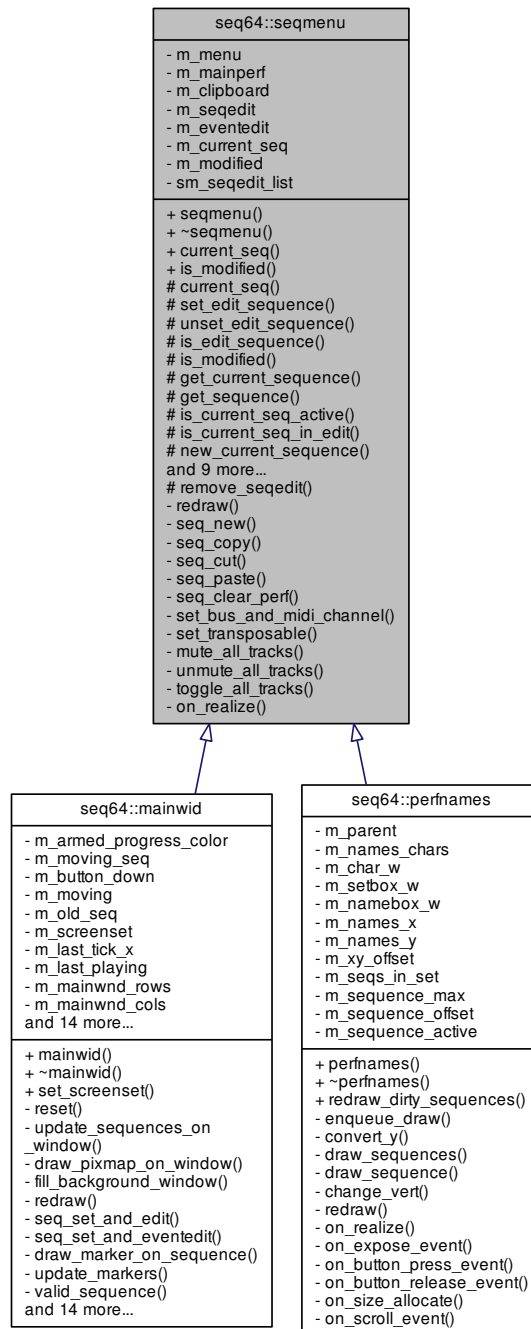
```
bool seq64::seqkeys::m_show_octave_letters [private]
```

If false, then the MIDI key numbers are shown instead. This is a new feature of Sequencer64.

### 13.89 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 [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.*
- [sequence](#) [m\\_clipboard](#)  
*Holds a copy of data concerning a sequence, which can then be pasted into another pattern slot.*
- [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.*

## Friends

- class [mainwnd](#)
- class [seqedit](#)

## 13.89.1 Detailed Description

It is an abstract base class.

## 13.89.2 Member Typedef Documentation

### 13.89.2.1 SeqeditMap

```
typedef std::map<int, seqedit *> seq64::seqmenu::SeqeditMap [private]
```

### 13.89.2.2 SeqeditPair

```
typedef std::pair<int, seqedit *> seq64::seqmenu::SeqeditPair [private]
```

### 13.89.2.3 iterator

```
typedef std::map<int, seqedit *>::iterator seq64::seqmenu::iterator [private]
```

### 13.89.2.4 const\_iterator

```
typedef std::map<int, seqedit *>::const_iterator seq64::seqmenu::const\_iterator [private]
```

## 13.89.3 Constructor & Destructor Documentation

### 13.89.3.1 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.
----------	---

## 13.89.3.2 ~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.

## 13.89.4 Member Function Documentation

## 13.89.4.1 current\_seq() [1/2]

```
int seq64::seqmenu::current_seq ( ) const [inline]
```

## 13.89.4.2 is\_modified() [1/2]

```
bool seq64::seqmenu::is_modified ( ) const [inline]
```

## 13.89.4.3 current\_seq() [2/2]

```
void seq64::seqmenu::current_seq (
    int seq ) [inline], [protected]
```

## 13.89.4.4 set\_edit\_sequence()

```
void seq64::seqmenu::set_edit_sequence (
    int seqnum ) [inline], [protected]
```

Now a pass-along to the perform object.

#### 13.89.4.5 unset\_edit\_sequence()

```
void seq64::seqmenu::unset_edit_sequence (
    int seqnum ) [inline], [protected]
```

#### 13.89.4.6 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.

#### 13.89.4.7 is\_modified() [2/2]

```
void seq64::seqmenu::is_modified (
    bool flag ) [inline], [protected]
```

#### 13.89.4.8 get\_current\_sequence()

```
sequence* seq64::seqmenu::get_current_sequence ( ) const [inline], [protected]
```

#### 13.89.4.9 get\_sequence()

```
sequence* seq64::seqmenu::get_sequence (
    int seqnum ) const [inline], [protected]
```

#### 13.89.4.10 is\_current\_seq\_active()

```
bool seq64::seqmenu::is_current_seq_active ( ) const [inline], [protected]
```

#### 13.89.4.11 is\_current\_seq\_in\_edit()

```
bool seq64::seqmenu::is_current_seq_in_edit ( ) const [inline], [protected]
```



#### 13.89.4.12 new\_current\_sequence()

```
void seq64::seqmenu::new_current_sequence ( ) [inline], [protected]
```

#### 13.89.4.13 new\_sequence()

```
void seq64::seqmenu::new_sequence (
    int seqnum ) [inline], [protected]
```

#### 13.89.4.14 delete\_current\_sequence()

```
void seq64::seqmenu::delete_current_sequence ( ) [inline], [protected]
```

#### 13.89.4.15 toggle\_current\_sequence()

```
void seq64::seqmenu::toggle_current_sequence ( ) [inline], [protected]
```

#### 13.89.4.16 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.

#### 13.89.4.17 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.

**13.89.4.18 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.

**13.89.4.19 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**

<b>s</b>	Provides the sequence for which the <code>seqedit</code> will be created. The perform 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.

**13.89.4.20 remove\_seqedit()**

```
void seq64::seqmenu::remove_seqedit (
    sequence & s ) [static], [protected]
```

Called by `seqedit::on_delete_event()`.

**13.89.4.21 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](#).

## 13.89.4.22 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](#).

## 13.89.4.23 redraw()

```
virtual void seq64::seqmenu::redraw (
    int a_sequence ) [private], [pure virtual]
```

Implemented in [seq64::mainwid](#), and [seq64::perfnames](#).

## 13.89.4.24 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.

## 13.89.4.25 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.

#### 13.89.4.26 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.

#### 13.89.4.27 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.

#### 13.89.4.28 seq\_clear\_perf()

```
void seq64::seqmenu::seq_clear_perf ( ) [private]
```

#### 13.89.4.29 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.

#### 13.89.4.30 set\_transposable()

```
void seq64::seqmenu::set_transposable (
    bool flag ) [private]
```

## Parameters

<i>flag</i>	The value to use to set the flag.
-------------	-----------------------------------

**13.89.4.31** mute\_all\_tracks()

```
void seq64::seqmenu::mute_all_tracks ( ) [inline], [private]
```

**13.89.4.32** unmute\_all\_tracks()

```
void seq64::seqmenu::unmute_all_tracks ( ) [inline], [private]
```

**13.89.4.33** toggle\_all\_tracks()

```
void seq64::seqmenu::toggle_all_tracks ( ) [inline], [private]
```

**13.89.4.34** on\_realize()

```
void seq64::seqmenu::on_realize ( ) [private]
```

**13.89.5** Friends And Related Function Documentation**13.89.5.1** mainwnd

```
friend class mainwnd [friend]
```

**13.89.5.2** seqedit

```
friend class seqedit [friend]
```

## 13.89.6 Field Documentation

### 13.89.6.1 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.

### 13.89.6.2 m\_menu

```
Gtk::Menu* seq64::seqmenu::m_menu [private]
```

### 13.89.6.3 m\_mainperf

```
perform& seq64::seqmenu::m_mainperf [private]
```

### 13.89.6.4 m\_clipboard

```
sequence seq64::seqmenu::m_clipboard [private]
```

### 13.89.6.5 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.

### 13.89.6.6 m\_eventedit

```
eventedit* seq64::seqmenu::m_eventedit [private]
```

#### 13.89.6.7 m\_current\_seq

```
int seq64::seqmenu::m_current_seq [private]
```

#### 13.89.6.8 m\_modified

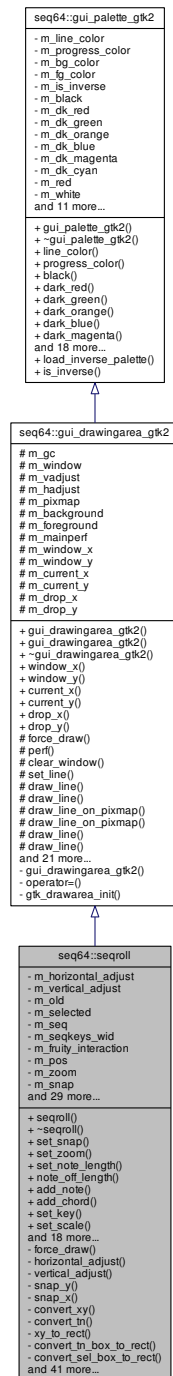
```
bool seq64::seqmenu::m_modified [private]
```

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

## 13.90 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, int ppqn=SEQ64\_USE\_DEFAULT\_PPQN)

*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 [add\\_chord](#) (midipulse tick, int note)  
*Convenience wrapper for [sequence::add\\_chord\(\)](#).*
- 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 selecton 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](#) ()

## Private 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` &ticks, int &note)
- void `convert_tn` (`midipulse` ticks, 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)  
*Converts rectangle corner coordinates to a starting coordinate, plus a width and height.*
- 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_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 &note\_h, `midipulse` &tick\_f, int &note\_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 &note\_h, `midipulse` &tick\_f, int &note\_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\\_selected](#) ()
  - 'Setter' function for member m\_old*
  - void [clear\\_old](#) ()
  - 'Setter' function for member m\_old*
  - void [clear\\_flags](#) ()
  - Clears all the mouse-action flags.*
  - 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 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*
  - void [on\\_realize](#) ()
  - Implements the on-realize event handling.*
  - bool [on\\_expose\\_event](#) (GdkEventExpose \*ev)
  - Implements the on-expose event handling.*
  - bool [on\\_button\\_press\\_event](#) (GdkEventButton \*ev)
  - Implements the on-button-press event handling.*
  - bool [on\\_button\\_release\\_event](#) (GdkEventButton \*ev)
  - Implements the on-button-release event handling.*
  - bool [on\\_motion\\_notify\\_event](#) (GdkEventMotion \*ev)
  - Implements the on-motion-notify event handling.*
  - 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)
  - Implements the on-key-press event handling.*
  - bool [on\\_scroll\\_event](#) (GdkEventScroll \*a\_ev)
  - Implements the on-scroll event handling.*
  - void [on\\_size\\_allocate](#) (Gtk::Allocation &)
  - Implements the on-size-allocate event handling.*
  - bool [on\\_leave\\_notify\\_event](#) (GdkEventCrossing \*p0)
  - Implements the on-leave-notify event handling.*
  - bool [on\\_enter\\_notify\\_event](#) (GdkEventCrossing \*p0)
  - Implements the on-enter-notify event handling.*

## Private 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.*
- `FruitySeqRollInput m_fruity_interaction`  
*Provides a fruity input object, whether it is needed or not.*
- `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_ppqn`  
*The value of PPQN for the current MIDI song.*
- `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 painting 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.*
- bool [m\\_transport\\_follow](#)  
*TBD.*
- 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.*
- midibyte [m\\_status](#)  
*Set to true to avoid the call to [update\\_and\\_draw\(\)](#).*
- midibyte [m\\_cc](#)  
*The current MIDI control value selected in the seqedit.*

## Friends

- class [FruitySeqRollInput](#)  
*This friend implements fruity interaction-specific behavior.*

## Additional Inherited Members

### 13.90.1 Constructor & Destructor Documentation

#### 13.90.1.1 seqroll()

```
seq64::seqroll::seqroll (
    perform & p,
    sequence & seq,
    int zoom,
    int snap,
    seqkeys & seqkeys_wid,
    int pos,
    Gtk::Adjustment & hadjust,
    Gtk::Adjustment & vadjust,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

## 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 <a href="#">seqroll::m_pos</a> . 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.
<i>ppqn</i>	The initial value of the PPQN for this sequence. Useful in scale calculations.

13.90.1.2 `~seqroll()`

```
seq64::seqroll::~~seqroll ( ) [virtual]
```

The only thing to delete here is the clipboard. Except it is never used, so is commented out.

## 13.90.2 Member Function Documentation

13.90.2.1 `set_snap()`

```
void seq64::seqroll::set_snap (
    int snap ) [inline]
```

## Parameters

<i>snap</i>	Provides the sname value to set.
-------------	----------------------------------

13.90.2.2 `set_zoom()`

```
void seq64::seqroll::set_zoom (
    int zoom )
```

## Parameters

<i>zoom</i>	The desired zoom value, assumed to be validated already. See the <a href="#">seqedit::set_zoom()</a> function.
-------------	--

## 13.90.2.3 set\_note\_length()

```
void seq64::seqroll::set_note_length (
    int note_length ) [inline]
```

## 13.90.2.4 note\_off\_length()

```
int seq64::seqroll::note_off_length ( ) const [inline]
```

## 13.90.2.5 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 <a href="#">sequence::add_note()</a> is false.

## 13.90.2.6 add\_chord()

```
void seq64::seqroll::add_chord (
    midipulse tick,
    int note ) [inline]
```

Implicit parameters are the `m_chord` and [note\\_off\\_length\(\)](#) members. The latter deducts just a little from the snap value.

**Parameters**

<i>tick</i>	The tick at which to add the chord.
<i>note</i>	The base (bottom) note of the chord.

**13.90.2.7 set\_key()**

```
void seq64::seqroll::set_key (
    int key )
```

**Parameters**

<i>key</i>	The desired key value.
------------	------------------------

**13.90.2.8 set\_scale()**

```
void seq64::seqroll::set_scale (
    int scale )
```

**Parameters**

<i>scale</i>	The desired scale value.
--------------	--------------------------

**13.90.2.9 set\_chord()**

```
void seq64::seqroll::set_chord (
    int chord )
```

**Parameters**

<i>chord</i>	The desired chord value.
--------------	--------------------------

**13.90.2.10 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.



**13.90.2.11 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.

**13.90.2.12 update\_pixmap()**

```
void seq64::seqroll::update_pixmap ( )
```

**13.90.2.13 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.

**13.90.2.14 update\_background()**

```
void seq64::seqroll::update_background ( )
```

The first thing done is to clear the background, painting it white.

**13.90.2.15 draw\_background\_on\_pixmap()**

```
void seq64::seqroll::draw_background_on_pixmap ( )
```

#### 13.90.2.16 draw\_events\_on\_pixmap()

```
void seq64::seqroll::draw_events_on_pixmap ( )
```

Just calls [draw\\_events\\_on\(\)](#).

#### 13.90.2.17 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.

#### 13.90.2.18 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 its 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.

#### 13.90.2.19 reset()

```
void seq64::seqroll::reset ( )
```

It's almost identical to the [change\\_horz\(\)](#) function, just calling [update\\_sizes\(\)](#) before [update\\_and\\_draw\(\)](#).

#### 13.90.2.20 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.
--------------	---

## 13.90.2.21 redraw()

```
void seq64::seqroll::redraw ( )
```

Somewhat similar to [seqevent::redraw\(\)](#). Actually, we don't seem to need to ignore redraw when making settings in the seqedit constructor, so this member no longer exists.

## 13.90.2.22 redraw\_events()

```
void seq64::seqroll::redraw_events ( )
```

Actually, that member is not needed and no longer exists.

## 13.90.2.23 start\_paste()

```
void seq64::seqroll::start_paste ( )
```

## 13.90.2.24 complete\_paste() [1/2]

```
void seq64::seqroll::complete_paste ( ) [inline]
```

## 13.90.2.25 complete\_paste() [2/2]

```
void seq64::seqroll::complete_paste (
    int x,
    int y )
```

## 13.90.2.26 follow\_progress()

```
void seq64::seqroll::follow_progress ( )
```

## 13.90.2.27 force\_draw()

```
void seq64::seqroll::force_draw ( ) [private], [virtual]
```

Reimplemented from [seq64::gui\\_drawingarea\\_gtk2](#).

## 13.90.2.28 horizontal\_adjust()

```
void seq64::seqroll::horizontal_adjust (
    double step ) [inline], [private]
```

A duplicate of the one in seqedit.

## Parameters

<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. See <a href="#">gui_drawingarea_gtk2::scroll_hadjust()</a> for more information.
-------------	---

13.90.2.29 `vertical_adjust()`

```
void seq64::seqroll::vertical_adjust (
    double step ) [inline], [private]
```

A duplicate of the one in `seqedit`.

## Parameters

<i>step</i>	Provides the step value to use for adjusting the vertical scrollbar. See <a href="#">gui_drawingarea_gtk2::scroll_vadjust()</a> for more information.
-------------	---

13.90.2.30 `snap_y()`

```
void seq64::seqroll::snap_y (
    int & y ) [inline], [private]
```

## Parameters

<i>out</i>	<i>y</i>	The y-value to be snapped.
------------	----------	----------------------------

13.90.2.31 `snap_x()`

```
void seq64::seqroll::snap_x (
    int & x ) [private]
```

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

<i>out</i>	<i>x</i>	Provides the x-value to be snapped and returned. A return value would be better.
------------	----------	--

### 13.90.2.32 convert\_xy()

```
void seq64::seqroll::convert_xy (
    int x,
    int y,
    midipulse & ticks,
    int & note ) [private]
```

### 13.90.2.33 convert\_tn()

```
void seq64::seqroll::convert_tn (
    midipulse tick,
    int note,
    int & x,
    int & y ) [private]
```

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.

### 13.90.2.34 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 ) [private]
```

This function checks the mins / maxes, and then fills in the x, y, width, and height values.

We should refactor this function to use the utility class `seqroll::rect` as the destination for the conversion.

#### Parameters

	<i>x1</i>	The x value of the first corner.
	<i>y1</i>	The y value of the first corner.

## Parameters

	<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.

13.90.2.35 `convert_tn_box_to_rect()`

```
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 ) [private]
```

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↔ _h</i>	The high note of the rectangle.
	<i>note↔ _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.

13.90.2.36 `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 ) [private]
```

## Parameters

<i>tick_s</i>	The starting tick of the rectangle.
<i>tick_f</i>	The finishing tick of the rectangle.
<i>note↔ _h</i>	The high note of the rectangle.
<i>note↔ _l</i>	The low note of the rectangle.

## 13.90.2.37 get\_selected\_box()

```
void seq64::seqroll::get_selected_box (
    midipulse & tick_s,
    int & note_h,
    midipulse & tick_f,
    int & note_l ) [private]
```

## 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↔ _h</i>	The high note of the rectangle.
out	<i>note↔ _l</i>	The low note of the rectangle.

## 13.90.2.38 draw\_events\_on()

```
void seq64::seqroll::draw_events_on (
    Glib::RefPtr< Gdk::Drawable > draw ) [private]
```

"Method 0" draws the background sequence, if active. "Method 1" draws the sequence itself.

## Parameters

<i>draw</i>	The "drawable" area to draw on.
-------------	---------------------------------

## 13.90.2.39 idle\_redraw()

```
int seq64::seqroll::idle_redraw ( ) [private]
```

**13.90.2.40 idle\_progress()**

```
int seq64::seqroll::idle_progress ( ) [private]
```

**13.90.2.41 change\_horz()**

```
void seq64::seqroll::change_horz ( ) [private]
```

Roughly similar to [seqevent::change\\_horz\(\)](#).

**13.90.2.42 change\_vert()**

```
void seq64::seqroll::change_vert ( ) [private]
```

**13.90.2.43 move\_selection\_box()**

```
void seq64::seqroll::move_selection_box (
    int dx,
    int dy ) [private]
```

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.

**13.90.2.44 move\_selected\_notes()**

```
void seq64::seqroll::move_selected_notes (
    int dx,
    int dy ) [private]
```

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.

**13.90.2.45 grow\_selected\_notes()**

```
void seq64::seqroll::grow_selected_notes (
    int dx ) [private]
```

**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.
-----------	---

**13.90.2.46 set\_adding()**

```
void seq64::seqroll::set_adding (
    bool adding ) [private]
```

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.
---------------	------------------------

**13.90.2.47 update\_mouse\_pointer()**

```
void seq64::seqroll::update_mouse_pointer (
    bool adding = false ) [private]
```

Moved here from the "fruity" seqroll class.

**13.90.2.48 button\_press\_initial()**

```
bool seq64::seqroll::button_press_initial (
    GdkEventButton * ev,
    int & norm_x,
    int & snapped_x,
    int & snapped_y ) [private]
```

**13.90.2.49 align\_selection()**

```
void seq64::seqroll::align_selection (
    midipulse & tick_s,
    int & note_h,
    midipulse & tick_f,
    int & note_l,
    int snapped_x ) [private]
```

Save offset that we get from the snap above. Align selection for drawing. Could be used in XRollInput::on\_button↵\_press\_event().

**13.90.2.50 button\_press()**

```
bool seq64::seqroll::button_press (
    GdkEventButton * ev ) [private]
```

This function now uses the needs\_update flag to determine if the perform object should modify().

**Parameters**

<b>ev</b>	Provides the button-press event to process.
-----------	---

**Returns**

Returns the value of needs\_update. It used to return only true.

**13.90.2.51 button\_release()**

```
bool seq64::seqroll::button_release (
    GdkEventButton * ev ) [private]
```

This function now uses the needs\_update flag to determine if the perform object should modify().

**Parameters**

<b>ev</b>	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 perfolinput.cpp.

**13.90.2.52 motion\_notify()**

```
bool seq64::seqroll::motion_notify (
    GdkEventMotion * ev ) [private]
```

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**

<b>ev</b>	Provides the button-release event to process.
-----------	---

**Returns**

Returns true if the event was processed.

**13.90.2.53 clear\_selected()**

```
void seq64::seqroll::clear_selected ( ) [inline], [private]
```

**13.90.2.54 clear\_old()**

```
void seq64::seqroll::clear_old ( ) [inline], [private]
```

**13.90.2.55 clear\_flags()**

```
void seq64::seqroll::clear_flags ( ) [inline], [private]
```

**13.90.2.56 scroll\_offset\_x()**

```
int seq64::seggroll::scroll_offset_x (  
    int x ) const [inline], [private]
```

Offsets the x value by the x origin of the current page.

## Parameters

<code>x</code>	The x value to offset.
----------------	------------------------

## 13.90.2.57 scroll\_offset\_y()

```
int seq64::seqroll::scroll_offset_y (
    int y ) const [inline], [private]
```

Offsets the y value by the y origin of the current page.

## Parameters

<code>y</code>	The y value to offset.
----------------	------------------------

## 13.90.2.58 set\_current\_offset\_x\_y()

```
void seq64::seqroll::set_current_offset_x_y (
    int x,
    int y ) [inline], [private]
```

Offsets the current x value by the x origin of the current page.

## Parameters

<code>x</code>	The x value to offset.
----------------	------------------------

`void set_current_offset_x (int x) { m_current_x = x + m_scroll_offset_x; }` Useful y calculation. Offsets the current y value by the y origin of the current page.

## Parameters

<code>y</code>	The y value to offset.
----------------	------------------------

`void set_current_offset_y (int y) { m_current_y = y + m_scroll_offset_y; }` Useful x and y calculation. Offsets the current x and y values by the x and y origin of the current page.

## Parameters

<code>x</code>	The y value to offset.
<code>y</code>	The y value to offset.

**13.90.2.59 adding()**

```
bool seq64::seqroll::adding ( ) const [inline], [private]
```

**13.90.2.60 selecting()**

```
bool seq64::seqroll::selecting ( ) const [inline], [private]
```

**13.90.2.61 growing()**

```
bool seq64::seqroll::growing ( ) const [inline], [private]
```

**13.90.2.62 normal\_action()**

```
bool seq64::seqroll::normal_action ( ) const [inline], [private]
```

**Returns**

Returns true if one of those five flags are set.

**13.90.2.63 select\_action()**

```
bool seq64::seqroll::select_action ( ) const [inline], [private]
```

**Returns**

Returns true if one of those four flags are set.

**13.90.2.64 drop\_action()**

```
bool seq64::seqroll::drop_action ( ) const [inline], [private]
```

**Returns**

Returns true if one of those two flags are set.

### 13.90.2.65 moving()

```
bool seq64::seqroll::moving ( ) const [inline], [private]
```

### 13.90.2.66 on\_realize()

```
void seq64::seqroll::on_realize ( ) [private]
```

### 13.90.2.67 on\_expose\_event()

```
bool seq64::seqroll::on_expose_event (
    GdkEventExpose * ev ) [private]
```

#### Parameters

<i>ev</i>	The expose event to process.
-----------	------------------------------

#### Returns

Always returns true.

### 13.90.2.68 on\_button\_press\_event()

```
bool seq64::seqroll::on_button_press_event (
    GdkEventButton * ev ) [private]
```

#### 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 Seq24Seq↔RollInput::on\_button\_press\_event() or [FruitySeqRollInput::on\\_button\\_press\\_event\(\)](#).

### 13.90.2.69 on\_button\_release\_event()

```
bool seq64::seqroll::on_button_release_event (
    GdkEventButton * ev ) [private]
```

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

<code>ev</code>	The button release event to process.
-----------------	--------------------------------------

#### Returns

Returns the return value of `Seq24SeqRollInput::on_button_release_event()` or [FruitySeqRollInput::on\\_button\\_release\\_event\(\)](#).

### 13.90.2.70 `on_motion_notify_event()`

```
bool seq64::seckill::on_motion_notify_event (
    GdkEventMotion * ev ) [private]
```

#### Parameters

<code>ev</code>	The motion-notification event to process.
-----------------	---

#### Returns

Returns the return value of `Seq24SeqRollInput::on_motion_notify_event()` or [FruitySeqRollInput::on\\_motion\\_notify\\_event\(\)](#).

### 13.90.2.71 `on_focus_in_event()`

```
bool seq64::seckill::on_focus_in_event (
    GdkEventFocus * ) [private]
```

Sets the GDK HAS\_FOCUS flag. Parameter "ev" is the event-focus event, not used.

#### Returns

Always returns false.



### 13.90.2.72 on\_focus\_out\_event()

```
bool seq64::seqroll::on_focus_out_event (
    GdkEventFocus * ) [private]
```

Resets the GDK HAS\_FOCUS flag. Parameter "ev" is the event-focus event, not used.

#### Returns

Always returns false.

### 13.90.2.73 on\_key\_press\_event()

```
bool seq64::seqroll::on_key_press_event (
    GdkEventKey * ev ) [private]
```

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)

### 13.90.2.74 on\_scroll\_event()

```
bool seq64::seqroll::on_scroll_event (
    GdkEventScroll * ev ) [private]
```

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.

**13.90.2.75 on\_size\_allocate()**

```
void seq64::seqroll::on_size_allocate (
    Gtk::Allocation & a ) [private]
```

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.
----------	----------------------------------

**13.90.2.76 on\_leave\_notify\_event()**

```
bool seq64::seqroll::on_leave_notify_event (
    GdkEventCrossing * p0 ) [private]
```

Calls `m_seqkeys_wid.set_hint_state(false)`. Parameter "ev" is the event-crossing event, not used.

**Returns**

Always returns false.

**13.90.2.77 on\_enter\_notify\_event()**

```
bool seq64::seqroll::on_enter_notify_event (
    GdkEventCrossing * p0 ) [private]
```

Calls `m_seqkeys_wid.set_hint_state(true)`. Parameter "ev" is the event-crossing event, not used.

**Returns**

Always returns false.

### 13.90.3 Friends And Related Function Documentation

#### 13.90.3.1 FruitySeqRollInput

```
friend class FruitySeqRollInput [friend]
```

We've absorbed the Seq24SeqRollInput class functionality back into seqroll, to save code.

### 13.90.4 Field Documentation

#### 13.90.4.1 m\_horizontal\_adjust

```
Gtk::Adjustment& seq64::seqroll::m_horizontal_adjust [private]
```

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.

#### 13.90.4.2 m\_vertical\_adjust

```
Gtk::Adjustment& seq64::seqroll::m_vertical_adjust [private]
```

#### 13.90.4.3 m\_old

```
rect seq64::seqroll::m_old [private]
```

#### 13.90.4.4 m\_selected

```
rect seq64::seqroll::m_selected [private]
```

#### 13.90.4.5 m\_seq

```
sequence& seq64::seqroll::m_seq [private]
```

#### 13.90.4.6 m\_seqkeys\_wid

```
seqkeys& seq64::seqroll::m_seqkeys_wid [private]
```

#### 13.90.4.7 m\_fruity\_interaction

```
FruitySeqRollInput seq64::seqroll::m_fruity_interaction [private]
```

#### 13.90.4.8 m\_pos

```
int seq64::seqroll::m_pos [private]
```

Need to clarify what exactly this member is used for.

#### 13.90.4.9 m\_zoom

```
int seq64::seqroll::m_zoom [private]
```

#### 13.90.4.10 m\_snap

```
int seq64::seqroll::m_snap [private]
```

Same meaning as for the event-bar grid. This value is the denominator of the note size used for the snap.

#### 13.90.4.11 m\_ppqn

```
int seq64::seqroll::m_ppqn [private]
```

Supports values other than the default of 192.

#### 13.90.4.12 m\_note\_length

```
int seq64::seqroll::m_note_length [private]
```

Used in the seq24seqroll module only.

**13.90.4.13 m\_scale**

```
int seq64::seqroll::m_scale [private]
```

**13.90.4.14 m\_chord**

```
int seq64::seqroll::m_chord [private]
```

**13.90.4.15 m\_key**

```
int seq64::seqroll::m_key [private]
```

**13.90.4.16 m\_adding**

```
bool seq64::seqroll::m_adding [private]
```

This flag was moved from both the fruity and the seq24 seqroll classes.

**13.90.4.17 m\_selecting**

```
bool seq64::seqroll::m_selecting [private]
```

**13.90.4.18 m\_moving**

```
bool seq64::seqroll::m_moving [private]
```

**13.90.4.19 m\_moving\_init**

```
bool seq64::seqroll::m_moving_init [private]
```

Used in the fruity and seq24 mouse-handling modules.

**13.90.4.20 m\_growing**

```
bool seq64::seqroll::m_growing [private]
```

**13.90.4.21 m\_painting**

```
bool seq64::seqroll::m_painting [private]
```

Used in the fruity and seq24 mouse-handling modules.

**13.90.4.22 m\_paste**

```
bool seq64::seqroll::m_paste [private]
```

**13.90.4.23 m\_is\_drag\_pasting**

```
bool seq64::seqroll::m_is_drag_pasting [private]
```

Used in the fruity mouse-handling module.

**13.90.4.24 m\_is\_drag\_pasting\_start**

```
bool seq64::seqroll::m_is_drag_pasting_start [private]
```

Used in the fruity mouse-handling module.

**13.90.4.25 m\_justselected\_one**

```
bool seq64::seqroll::m_justselected_one [private]
```

Used in the fruity mouse-handling module.

**13.90.4.26 m\_move\_delta\_x**

```
int seq64::seqroll::m_move_delta_x [private]
```

**13.90.4.27 m\_move\_delta\_y**

```
int seq64::seqroll::m_move_delta_y [private]
```

**13.90.4.28 m\_move\_snap\_offset\_x**

```
int seq64::seqroll::m_move_snap_offset_x [private]
```

**13.90.4.29 m\_progress\_x**

```
int seq64::seqroll::m_progress_x [private]
```

**13.90.4.30 m\_scroll\_offset\_ticks**

```
int seq64::seqroll::m_scroll_offset_ticks [private]
```

**13.90.4.31 m\_scroll\_offset\_key**

```
int seq64::seqroll::m_scroll_offset_key [private]
```

**13.90.4.32 m\_scroll\_offset\_x**

```
int seq64::seqroll::m_scroll_offset_x [private]
```

**13.90.4.33 m\_scroll\_offset\_y**

```
int seq64::seqroll::m_scroll_offset_y [private]
```

**13.90.4.34 m\_transport\_follow**

```
bool seq64::seqroll::m_transport_follow [private]
```

**13.90.4.35 m\_trans\_button\_press**

```
bool seq64::seqroll::m_trans_button_press [private]
```

**13.90.4.36 m\_background\_sequence**

```
int seq64::seqroll::m_background_sequence [private]
```

#### 13.90.4.37 m\_drawing\_background\_seq

```
bool seq64::seqroll::m_drawing_background_seq [private]
```

#### 13.90.4.38 m\_status

```
midibyte seq64::seqroll::m_status [private]
```

Used in [set\\_background\\_sequence\(\)](#), [change\\_horz\(\)](#), [change\\_vert\(\)](#), [reset\(\)](#).... Never set to true, except in seq24, let's just comment it out for now. It hasn't been used in sequencer64 for awhile now.

bool m\_ignore\_redraw; The current status/event selected in the seqedit. Not used in seqroll at present.

#### 13.90.4.39 m\_cc

```
midibyte seq64::seqroll::m_cc [private]
```

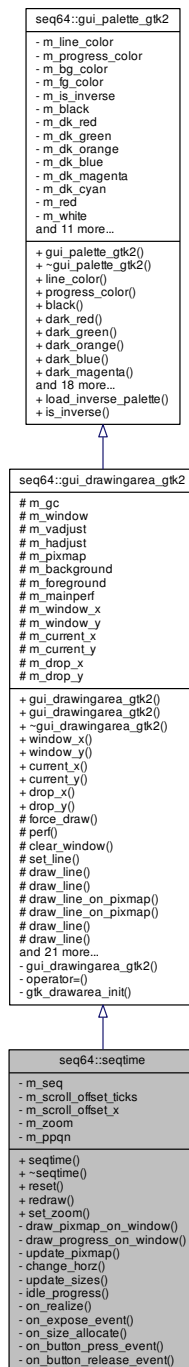
Not used in seqroll at present.

## 13.91 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, int ppqn=SEQ64\_USE\_DEFAULT\_PPQN)

*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*
- int `m_ppqn`

## Additional Inherited Members

### 13.91.1 Constructor & Destructor Documentation

## 13.91.1.1 seqtime()

```
seq64::seqtime::seqtime (
    sequence & seq,
    perform & p,
    int zoom,
    Gtk::Adjustment & hadjust,
    int ppqn = SEQ64_USE_DEFAULT_PPQN )
```

In the constructor you can only allocate colors; `get_window()` returns 0 because the window is not yet realized>

## 13.91.1.2 ~seqtime()

```
virtual seq64::seqtime::~~seqtime ( ) [inline], [virtual]
```

## 13.91.2 Member Function Documentation

## 13.91.2.1 reset()

```
void seq64::seqtime::reset ( )
```

Basically identical to [sequevent::reset\(\)](#).

## 13.91.2.2 redraw()

```
void seq64::seqtime::redraw ( )
```

## 13.91.2.3 set\_zoom()

```
void seq64::seqtime::set_zoom (
    int zoom )
```

## Parameters

<i>zoom</i>	The desired zoom value, assumed to be validated already. See the <a href="#">seqedit::set_zoom()</a> function.
-------------	--

## 13.91.2.4 draw\_pixmap\_on\_window()

```
void seq64::seqtime::draw_pixmap_on_window ( ) [private]
```

### 13.91.2.5 draw\_progress\_on\_window()

```
void seq64::seqtime::draw_progress_on_window ( ) [private]
```

### 13.91.2.6 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
m1					
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.

### 13.91.2.7 change\_horz()

```
void seq64::seqtime::change_horz ( ) [private]
```

### 13.91.2.8 update\_sizes()

```
void seq64::seqtime::update_sizes ( ) [private]
```

### 13.91.2.9 idle\_progress()

```
bool seq64::seqtime::idle_progress ( ) [inline], [private]
```

### 13.91.2.10 on\_realize()

```
void seq64::seqtime::on_realize ( ) [private]
```

Call the base-class version of this function first. Then addition resources are allocated.

### 13.91.2.11 on\_expose\_event()

```
bool seq64::seqtime::on_expose_event (
    GdkEventExpose * a_ev ) [private]
```

### 13.91.2.12 on\_size\_allocate()

```
void seq64::seqtime::on_size_allocate (
    Gtk::Allocation & a ) [private]
```

### 13.91.2.13 on\_button\_press\_event()

```
bool seq64::seqtime::on_button_press_event (
    GdkEventButton * ) [inline], [private]
```

Simply returns false.

### 13.91.2.14 on\_button\_release\_event()

```
bool seq64::seqtime::on_button_release_event (
    GdkEventButton * ) [inline], [private]
```

Simply returns false.

### 13.91.3 Field Documentation

#### 13.91.3.1 m\_seq

```
sequence& seq64::seqtime::m_seq [private]
```

#### 13.91.3.2 m\_scroll\_offset\_ticks

```
int seq64::seqtime::m_scroll_offset_ticks [private]
```

#### 13.91.3.3 m\_scroll\_offset\_x

```
int seq64::seqtime::m_scroll_offset_x [private]
```

#### 13.91.3.4 m\_zoom

```
int seq64::seqtime::m_zoom [private]
```

#### 13.91.3.5 m\_ppqn

```
int seq64::seqtime::m_ppqn [private]
```

## 13.92 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\\_e](#) {  
    [e\\_select](#),  
    [e\\_select\\_one](#),  
    [e\\_is\\_selected](#),  
    [e\\_would\\_select](#),  
    [e\\_deselect](#),  
    [e\\_toggle\\_selection](#),  
    [e\\_remove\\_one](#) }

*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 [get\\_trigger\\_count](#) () const  
*Gets the trigger count, useful for exporting a sequence.*
- void [set\\_trigger\\_paste\\_tick](#) (midipulse tick)
- midipulse [get\\_trigger\\_paste\\_tick](#) () const
- 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.*
- 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`.*
- void `set_name (char *name)`  
*Sets the sequence name member, `m_name`.*
- void `set_measures (int lengthmeasures)`
- int `get_measures ()`
- 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 pernames 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*
- const char \* [get\\_name](#) () const  
*'Getter' function for member m\_name pointer*
- const std::string & [name](#) () const  
*'Getter' function for member m\_name*
- 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.*
- [midipulse get\\_length](#) () const  
*'Getter' function for member m\_length*
- [midipulse get\\_last\\_tick](#) ()  
*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](#) ()  
*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.*
- void [off\\_queued](#) ()  
*'Setter' function for member m\_queued Turns off (resets) the queued flag and sets the dirty-mp flag.*
- void [on\\_queued](#) ()  
*'Setter' function for member m\_queued Turns on (sets) 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)  
*'Setter' function for member m\_recording and m\_notes\_on*
- bool [get\\_recording](#) () const  
*'Getter' function for member m\_recording*
- void [set\\_snap\\_tick](#) (int st)  
*'Setter' function for member m\_snap\_tick*
- void [set\\_quantized\\_rec](#) (bool qr)

- 'Setter' function for member m\_quantized\_rec*
- bool `get_quantized_rec` () const
- 'Getter' function for member m\_quantized\_rec*
- void `set_thru` (bool)
- 'Setter' function for member m\_thru*
- bool `get_thru` () const
- 'Getter' function for member m\_thru*
- 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>*
- 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)
- The `play()` function dumps notes starting from the given tick, and it pre-buffers ahead.*
- void `play_queue` (midipulse tick, bool playbackmode)
- Provides encapsulation for a series of called used in `perform::play()`.*
- void `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_event` (const event &er)
- Adds an event to the internal event list in a sorted manner.*
- void `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.*
- void `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 `grow_trigger` (midipulse tick\_from, midipulse tick\_to, midipulse len)

- Grows a trigger.*

  - void `del_trigger` (`midipulse` tick)

*Deletes a trigger, that brackets the given tick, from the trigger-list.*
- bool `get_trigger_state` (`midipulse` tick)

*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_triggers` ()

*Unselects all triggers.*
- bool `intersect_triggers` (`midipulse` position, `midipulse` &start, `midipulse` &ender)

*This function examines each trigger in the trigger list.*
- bool `intersect_notes` (`midipulse` position, `midipulse` position\_note, `midipulse` &start, `midipulse` &ender, int &note)

*This function examines each note 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 `del_selected_trigger` ()

*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_selected_triggers_to` (`midipulse` tick, bool `adjust_offset`, `triggers::grow_edit_t` which=`triggers::G←ROW_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` ()

*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\_masterbus 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_e` 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\_e 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)
- 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 &note\_h, midipulse &tick\_f, int &note\_l)

*Returns the 'box' of the selected items.*
- void `get_clipboard_box` (midipulse &tick\_s, int &note\_h, midipulse &tick\_f, int &note\_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 [remove\\_marked](#) ()  
*Removes marked events.*
- bool [mark\\_selected](#) ()  
*Marks the selected events.*
- void [remove\\_selected](#) ()  
*Removes selected 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 [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 [reset\\_draw\\_marker](#) ()  
*This refreshes the play marker to the last tick.*
- void [reset\\_draw\\_trigger\\_marker](#) ()  
*Sets the draw-trigger iterator to the beginning of the trigger list.*
- [draw\\_type\\_t](#) [get\\_next\\_note\\_event](#) ([midipulse](#) \*tick\_s, [midipulse](#) \*tick\_f, int \*note, 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, [midipulse](#) \*tick, [midibyte](#) \*d0, [midibyte](#) \*d1, bool \*selected, int evtype=EVENTS\_ALL)  
*Get the next event in the event list that matches the given status and control character.*
- bool [get\\_next\\_event](#) ([midibyte](#) \*status, [midibyte](#) \*cc)  
*Get the next event in the event list.*
- 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)
  - 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*

## 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 `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 `channel_match` (const `event` &e) const
- Checks to see if the event's channel matches the sequence's nominal channel.*

## 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](#)  
*The 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.*
- [int](#) [m\\_notes\\_on](#)  
*Provides a member to hold the polyphonic step-edit note counter.*
- [mastermidibus](#) \* [m\\_masterbus](#)  
*Provides the master MIDI buss which handles the output of the sequence to the proper buss and MIDI channel.*
- [int](#) [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\\_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 next tick to play?*
- 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.*
- int [m\\_ppqn](#)  
*Holds the PPQN value for this sequence, so that we don't have to rely on a global constant value.*
- int [m\\_seq\\_number](#)  
*A new member so that the sequence number is carried along with the sequence.*
- 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).*
- int [m\\_time\\_beats\\_per\\_measure](#)  
*Provides the number of beats per bar used in this sequence.*
- int [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.*
- int [m\\_rec\\_vol](#)  
*The volume to be used when recording.*
- int [m\\_note\\_on\\_velocity](#)  
*The Note On velocity used.*
- int [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.*
- int [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.*

## Friends

- class [perform](#)
- class [triggers](#)

### 13.92.1 Detailed Description

More members than you can shake a stick at.

### 13.92.2 Member Typedef Documentation

#### 13.92.2.1 EventStack

```
typedef std::stack<event\_list> seq64::sequence::EventStack [private]
```

### 13.92.3 Member Enumeration Documentation

#### 13.92.3.1 select\_action\_e

```
enum seq64::sequence::select\_action\_e
```

Se the [select\\_note\\_events\(\)](#) and [select\\_events\(\)](#) functions.

#### Enumerator

<a href="#">e_select</a>	Selection in progress.
<a href="#">e_select_one</a>	To select a single event.
<a href="#">e_is_selected</a>	The events are selected.
<a href="#">e_would_select</a>	The events would be selected.
<a href="#">e_deselect</a>	To deselect event under the cursor.
<a href="#">e_toggle_selection</a>	Toggle selection under cursor.
<a href="#">e_remove_one</a>	To remove one note under the cursor.

### 13.92.4 Constructor & Destructor Documentation

#### 13.92.4.1 `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.
-------------	---

#### 13.92.4.2 `~sequence()`

```
seq64::sequence::~~sequence ( )
```

### 13.92.5 Member Function Documentation

#### 13.92.5.1 `operator=()`

```
sequence& seq64::sequence::operator= (
    const sequence & rhs ) [private]
```

#### 13.92.5.2 `partial_assign()`

```
void seq64::sequence::partial_assign (
    const sequence & rhs )
```

We're replacing that incomplete 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.
------------	---

**13.92.5.3 events()** [1/2]

```
event_list& seq64::sequence::events ( ) [inline]
```

**13.92.5.4 events()** [2/2]

```
const event_list& seq64::sequence::events ( ) const [inline]
```

**13.92.5.5 any\_selected\_notes()**

```
bool seq64::sequence::any_selected_notes ( ) const [inline]
```

**13.92.5.6 triggerlist()** [1/2]

```
const triggers::List& seq64::sequence::triggerlist ( ) const [inline]
```

**13.92.5.7 triggerlist()** [2/2]

```
triggers::List& seq64::sequence::triggerlist ( ) [inline]
```

**13.92.5.8 get\_trigger\_count()**

```
int seq64::sequence::get_trigger_count ( ) const [inline]
```

**13.92.5.9 set\_trigger\_paste\_tick()**

```
void seq64::sequence::set_trigger_paste_tick (
    midipulse tick ) [inline]
```

**13.92.5.10 get\_trigger\_paste\_tick()**

```
midipulse seq64::sequence::get_trigger_paste_tick ( ) const [inline]
```

**13.92.5.11 number()** [1/2]

```
int seq64::sequence::number ( ) const [inline]
```

**13.92.5.12 number()** [2/2]

```
void seq64::sequence::number (
    int seqnum ) [inline]
```

**13.92.5.13 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..

**13.92.5.14 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()`.

**13.92.5.15 set\_hold\_undo()**

```
void seq64::sequence::set_hold_undo (
    bool hold )
```

## Parameters

<i>hold</i>	If true, then the events in the m_events container are added to the m_events_undo_hold container. Otherwise, that container is cleared.
-------------	---

**13.92.5.16 get\_hold\_undo()**

```
int seq64::sequence::get_hold_undo ( ) const [inline]
```

**13.92.5.17 set\_have\_undo()**

```
void seq64::sequence::set_have_undo ( ) [inline]
```

**13.92.5.18 have\_undo()**

```
bool seq64::sequence::have_undo ( ) const [inline]
```

**13.92.5.19 set\_have\_redo()**

```
void seq64::sequence::set_have_redo ( ) [inline]
```

**13.92.5.20 have\_redo()**

```
bool seq64::sequence::have_redo ( ) const [inline]
```

**13.92.5.21 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.
-------------	--

**13.92.5.22 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*

**13.92.5.23 pop\_redo()**

```
void seq64::sequence::pop_redo ( )
```

*Threadsafe*

**13.92.5.24 push\_trigger\_undo()**

```
void seq64::sequence::push_trigger_undo ( )
```

*Threadsafe*

**13.92.5.25 pop\_trigger\_undo()**

```
void seq64::sequence::pop_trigger_undo ( )
```

*Threadsafe*

**13.92.5.26 pop\_trigger\_redo()**

```
void seq64::sequence::pop_trigger_redo ( )
```

*Threadsafe*

**13.92.5.27 set\_name()** [1/2]

```
void seq64::sequence::set_name (
    const std::string & name )
```

**13.92.5.28** set\_name() [2/2]

```
void seq64::sequence::set_name (
    char * name )
```

**13.92.5.29** set\_measures()

```
void seq64::sequence::set_measures (
    int lengthmeasures )
```

**13.92.5.30** get\_measures()

```
int seq64::sequence::get_measures ( )
```

**13.92.5.31** get\_ppqn()

```
int seq64::sequence::get_ppqn ( ) const [inline]
```

**13.92.5.32** 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.
------------------------	---

**13.92.5.33** get\_beats\_per\_bar()

```
int seq64::sequence::get_beats_per_bar ( ) const [inline]
```

**13.92.5.34 set\_beat\_width()**

```
void seq64::sequence::set_beat_width (
    int beatwidth )
```

*Threadsafe*

Parameters

<i>beatwidth</i>	The new setting of the beat width value.
------------------	--

**13.92.5.35 get\_beat\_width()**

```
int seq64::sequence::get_beat_width ( ) const [inline]
```

*Threadsafe*

**13.92.5.36 measures\_to\_ticks()**

```
midipulse seq64::sequence::measures_to_ticks (
    int measures = 1 ) const [inline]
```

**13.92.5.37 clocks\_per\_metronome()** [1/2]

```
void seq64::sequence::clocks_per_metronome (
    int cpm ) [inline]
```

**13.92.5.38 clocks\_per\_metronome()** [2/2]

```
int seq64::sequence::clocks_per_metronome ( ) const [inline]
```

**13.92.5.39 set\_32nds\_per\_quarter()**

```
void seq64::sequence::set_32nds_per_quarter (
    int tpq ) [inline]
```



**13.92.5.40 get\_32nds\_per\_quarter()**

```
int seq64::sequence::get_32nds_per_quarter ( ) const [inline]
```

**13.92.5.41 us\_per\_quarter\_note()** [1/2]

```
void seq64::sequence::us_per_quarter_note (
    long upqn ) [inline]
```

**13.92.5.42 us\_per\_quarter\_note()** [2/2]

```
long seq64::sequence::us_per_quarter_note ( ) const [inline]
```

**13.92.5.43 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.
---------------	---

**13.92.5.44 set\_song\_mute()**

```
void seq64::sequence::set_song_mute (
    bool mute ) [inline]
```

**13.92.5.45 toggle\_song\_mute()**

```
void seq64::sequence::toggle_song_mute ( ) [inline]
```

**13.92.5.46 get\_song\_mute()**

```
bool seq64::sequence::get_song_mute ( ) const [inline]
```

**13.92.5.47 apply\_song\_transpose()**

```
void seq64::sequence::apply_song_transpose ( )
```

**13.92.5.48 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.

**13.92.5.49 get\_transposable()**

```
bool seq64::sequence::get_transposable ( ) const [inline]
```

**13.92.5.50 get\_name()**

```
const char* seq64::sequence::get_name ( ) const [inline]
```

**Deprecated****13.92.5.51 name()**

```
const std::string& seq64::sequence::name ( ) const [inline]
```

**13.92.5.52 set\_editing()**

```
void seq64::sequence::set_editing (
    bool edit ) [inline]
```

**13.92.5.53 get\_editing()**

```
bool seq64::sequence::get_editing ( ) const [inline]
```

**13.92.5.54 set\_raise()**

```
void seq64::sequence::set_raise (
    bool edit ) [inline]
```

**13.92.5.55 get\_raise()**

```
bool seq64::sequence::get_raise (
    void ) const [inline]
```

**13.92.5.56 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 m_length is used and does not change. It also sets the length value for the sequence's triggers.
<i>adjust_triggers</i>	If true, m_triggers.adjust_offsets_to_length() is called. The value defaults to true.
<i>verify</i>	This new parameter defaults to true. If true, <a href="#">verify_and_link()</a> is called. Otherwise, it is not, and the caller should call this function with the default value after reading all the events.

## 13.92.5.57 get\_length()

```
midipulse seq64::sequence::get_length ( ) const [inline]
```

## 13.92.5.58 get\_last\_tick()

```
midipulse seq64::sequence::get_last_tick ( )
```

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?

## 13.92.5.59 set\_last\_tick()

```
void seq64::sequence::set_last_tick (
    midipulse tick )
```

*Threadsafe*

## 13.92.5.60 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.

## 13.92.5.61 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.
----------	--

**13.92.5.62 get\_playing()**

```
bool seq64::sequence::get_playing ( ) const [inline]
```

**13.92.5.63 toggle\_playing()**

```
void seq64::sequence::toggle_playing ( ) [inline]
```

How exactly does this differ from toggling the mute status?

**13.92.5.64 toggle\_queued()**

```
void seq64::sequence::toggle_queued ( )
```

Also calculates the queued tick based on `m_last_tick`.

*Threadsafe*

**13.92.5.65 off\_queued()**

```
void seq64::sequence::off_queued ( )
```

Do we need to set `m_queued_tick` as in [toggle\\_queued\(\)](#)? Currently not used.

*Threadsafe*

**13.92.5.66 on\_queued()**

```
void seq64::sequence::on_queued ( )
```

Do we need to set `m_queued_tick` as in [toggle\\_queued\(\)](#)? Currently not used.

*Threadsafe*

**13.92.5.67 get\_queued()**

```
bool seq64::sequence::get_queued ( ) const [inline]
```

**13.92.5.68 get\_queued\_tick()**

```
midipulse seq64::sequence::get_queued_tick ( ) const [inline]
```

**13.92.5.69 check\_queued\_tick()**

```
bool seq64::sequence::check_queued_tick (
    midipulse tick ) const [inline]
```

**13.92.5.70 set\_recording()**

```
void seq64::sequence::set_recording (
    bool r )
```

*Threadsafe*

**13.92.5.71 get\_recording()**

```
bool seq64::sequence::get_recording ( ) const [inline]
```

**13.92.5.72 set\_snap\_tick()**

```
void seq64::sequence::set_snap_tick (
    int st )
```

*Threadsafe*

**13.92.5.73 set\_quantized\_rec()**

```
void seq64::sequence::set_quantized_rec (
    bool qr )
```

*Threadsafe*

**13.92.5.74 get\_quantized\_rec()**

```
bool seq64::sequence::get_quantized_rec ( ) const [inline]
```

### 13.92.5.75 set\_thru()

```
void seq64::sequence::set_thru (
    bool r )
```

*Threadsafe*

### 13.92.5.76 get\_thru()

```
bool seq64::sequence::get_thru ( ) const [inline]
```

### 13.92.5.77 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.

### 13.92.5.78 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.

### 13.92.5.79 is\_dirty\_perf()

```
bool seq64::sequence::is_dirty_perf ( )
```

*Threadsafe*

#### Returns

Returns the dirty status.

#### 13.92.5.80 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.

#### 13.92.5.81 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*

#### 13.92.5.82 set\_dirty()

```
void seq64::sequence::set_dirty ( )
```

*Threadsafe*

#### 13.92.5.83 get\_midi\_channel()

```
midibyte seq64::sequence::get_midi_channel ( ) const [inline]
```

#### 13.92.5.84 is\_smf\_0()

```
bool seq64::sequence::is_smf_0 ( ) const [inline]
```

#### 13.92.5.85 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 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.

## 13.92.5.86 print()

```
void seq64::sequence::print ( ) const
```

*Not threadsafe*

## 13.92.5.87 print\_triggers()

```
void seq64::sequence::print_triggers ( ) const
```

*Not threadsafe*

## 13.92.5.88 play()

```
void seq64::sequence::play (
    midipulse end_tick,
    bool playback_mode )
```

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.

The trigger calculations have been offloaded to the [triggers::play\(\)](#) function. It's return value and side-effects tell if there's a change in playing based on triggers and tells the ticks that bracket it.

## Parameters

<i>end_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.

*Threadsafe*

### 13.92.5.89 play\_queue()

```
void seq64::sequence::play_queue (
    midipulse tick,
    bool playbackmode )
```

Starts the playing of a pattern/sequence. This function just has the sequence dump its events. It ignores the sequence if it has no playable MIDI events.

*Change Note* ca 2016-10-12 Issue #39. Removed the check for a non-zero event count. This lets the seqroll show the progress bar in motion.

#### Parameters

<i>tick</i>	Provides the tick/pulse from which to start playing.
<i>playbackmode</i>	Indicates if the playback is in live mode (false) or song mode (true).

### 13.92.5.90 add\_note()

```
void 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!

## 13.92.5.91 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.

**13.92.5.92 add\_chord()**

```
void 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.

**13.92.5.93 add\_event()** [2/2]

```
void 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.

## 13.92.5.94 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 added.

## 13.92.5.95 sort\_events()

```
void seq64::sequence::sort_events ( ) [inline]
```

## 13.92.5.96 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.

## 13.92.5.97 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.
------------------	---------------------------------

#### 13.92.5.98 `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*

#### 13.92.5.99 `del_trigger()`

```
void seq64::sequence::del_trigger (
    midipulse tick )
```

See [triggers::remove\(\)](#).

*Threadsafe*

Parameters

<i>tick</i>	Provides the tick to be used for finding the trigger to be erased.
-------------	--

#### 13.92.5.100 `get_trigger_state()`

```
bool seq64::sequence::get_trigger_state (
    midipulse tick )
```

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.

**13.92.5.101 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()`.

**13.92.5.102 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()`.

**13.92.5.103 unselect\_triggers()**

```
bool seq64::sequence::unselect_triggers ( )
```

**Returns**

Returns the `m_triggers.unselect()` return value.

### 13.92.5.104 intersect\_triggers()

```
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.

### 13.92.5.105 intersect\_notes()

```
bool seq64::sequence::intersect_notes (
    midipulse position,
    midipulse position_note,
    midipulse & start,
    midipulse & ender,
    int & note )
```

If the given position is between the current notes on and off time values, values, the these values are copied to the start and end parameters, respectively, the note value is copied to the note parameter, and then we exit.

*Threadsafe*

#### Parameters

	<i>position</i>	The position to examine.
	<i>position_note</i>	I think this is the note value we might be looking for ???
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. Why is this an int value???

#### Returns

Returns true if a 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.



**13.92.5.106 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.

**13.92.5.107 del\_selected\_trigger()**

```
void seq64::sequence::del_selected_trigger ( )
```

**13.92.5.108 cut\_selected\_trigger()**

```
void seq64::sequence::cut_selected_trigger ( )
```

**13.92.5.109 copy\_selected\_trigger()**

```
void seq64::sequence::copy_selected_trigger ( )
```

Then it copies the first selected trigger that is found.

**13.92.5.110 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.
-------------------	--

## 13.92.5.111 move\_selected\_triggers\_to()

```
bool seq64::sequence::move_selected_triggers_to (
    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 indicate that the movement could be made. Used in [Seq24Perflnput::handle\\_motion\\_key\(\)](#).

## 13.92.5.112 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.

**13.92.5.113 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.

**13.92.5.114 get\_max\_trigger()**

```
midipulse seq64::sequence::get_max_trigger ( )
```

*Threadsafe*

**Returns**

Returns the maximum trigger value.

**13.92.5.115 move\_triggers()**

```
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.

**13.92.5.116 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.

13.92.5.117 `clear_triggers()`

```
void seq64::sequence::clear_triggers ( )
```

*Threadsafe*13.92.5.118 `get_trigger_offset()`

```
midipulse seq64::sequence::get_trigger_offset ( ) const [inline]
```

13.92.5.119 `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.

13.92.5.120 `get_midi_bus()`

```
char seq64::sequence::get_midi_bus ( ) const [inline]
```

13.92.5.121 `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.
------------	---

## 13.92.5.122 select\_note\_events()

```
int seq64::sequence::select_note_events (
    midipulse tick_s,
    int note_h,
    midipulse tick_f,
    int note_l,
    select_action_e 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</i> <sub>↔</sub> <i>_h</i>	The highest note selected.
<i>tick_f</i>	The ending, or finishing, tick.
<i>note</i> <sub>↔</sub> <i>_l</i>	The lowest note selected.
<i>action</i>	The action to perform on the selection.

## Returns

Returns the number of notes selected.

## 13.92.5.123 select\_events() [1/3]

```
int seq64::sequence::select_events (
    midipulse tick_s,
```

```

midipulse tick_f,
midibyte status,
midibyte cc,
select_action_e 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.
<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.

#### 13.92.5.124 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.

**13.92.5.125 select\_events()** [3/3]

```
int seq64::sequence::select_events (
    midipulse tick_s,
    midipulse tick_f,
    midibyte status )
```

**13.92.5.126 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.

**13.92.5.127 select\_linked()**

```
int seq64::sequence::select_linked (
    long tick_s,
    long tick_f,
    midibyte status )
```

**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.

**13.92.5.128 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.

**13.92.5.129 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.
----------------	---

**13.92.5.130 get\_num\_selected\_notes()**

```
int seq64::sequence::get_num_selected_notes ( ) const
```

**Threadsafe****Returns**

Returns m\_events.count\_selected\_notes().



**13.92.5.131 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()`.

**13.92.5.132 select\_all()**

```
void seq64::sequence::select_all ( )
```

*Threadsafe*

**13.92.5.133 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*

**13.92.5.134 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.
-------------------	---

## 13.92.5.135 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.

**13.92.5.136** 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.

**13.92.5.137** 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</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.

**13.92.5.138** 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

<code>t</code>	Provides the timestamp to be adjusted based on <code>m_length</code> .
<code>isnoteoff</code>	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.

#### 13.92.5.139 trim\_timestamp()

```
midipulse seq64::sequence::trim_timestamp (
    midipulse t )
```

Similar to `adjust_timestamp`, but it doesn't have an `isnoteoff` parameter.

#### Parameters

<code>t</code>	Provides the timestamp to be adjusted based on <code>m_length</code> .
----------------	--

#### Returns

Returns the adjusted timestamp.

#### 13.92.5.140 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.

## 13.92.5.141 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.

**13.92.5.142 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.

**13.92.5.143 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.

#### Returns

Returns true if the data was changed.

#### 13.92.5.144 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 <a href="#">wave_func()</a> 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.

#### 13.92.5.145 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.

#### 13.92.5.146 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.

#### 13.92.5.147 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.
--------------	--

#### 13.92.5.148 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.
-------------------	--

**13.92.5.149 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.

**13.92.5.150 mark\_selected()**

```
bool seq64::sequence::mark_selected ( )
```

*Threadsafe*

**Returns**

Returns true if there were any events that got marked.

**13.92.5.151 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.

**13.92.5.152 unpaint\_all()**

```
void seq64::sequence::unpaint_all ( )
```

*Threadsafe*

**13.92.5.153 unselect()**

```
void seq64::sequence::unselect ( )
```

*Threadsafe*

**13.92.5.154 verify\_and\_link()**

```
void seq64::sequence::verify_and_link ( )
```

*Threadsafe*

**13.92.5.155 link\_new()**

```
void seq64::sequence::link_new ( )
```

*Threadsafe*

**13.92.5.156 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.

**13.92.5.157 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.
-------------	--

**13.92.5.158 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.
-------------	--

**13.92.5.159 off\_playing\_notes()**

```
void seq64::sequence::off_playing_notes ( )
```

This function does not bother checking if `m_masterbus` is a null pointer.

*Threadsafe*

**13.92.5.160 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.
------------------	---

**13.92.5.161 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.

**13.92.5.162 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.

*Threadsafe*

**13.92.5.163 reset\_draw\_trigger\_marker()**

```
void seq64::sequence::reset_draw_trigger_marker ( )
```

*Threadsafe*

13.92.5.164 `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.
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.

**13.92.5.165 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 SEQ64_MIDI_COUNT_MAX-1.
<i>highest</i>	A reference parameter to return the note with the highest value. if there are no notes, then it is set to -1.

**Returns**

If there are no notes in the list, then false is returned, and the results should be disregarded.

**13.92.5.166 get\_next\_event()** [1/2]

```
bool seq64::sequence::get_next_event (
    midibyte status,
    midibyte cc,
    midipulse * tick,
    midibyte * d0,
    midibyte * d1,
    bool * selected,
    int evtype = EVENTS_ALL )
```

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 EVENT_ANY can be provided so that no event statuses are filtered.
<i>cc</i>	The continuous controller value that might be desired.
<i>tick</i>	A pointer return value for the tick value of the next event found.
<i>d0</i>	A pointer return value for the first data value of the event.
<i>d1</i>	A pointer return value for the second data value of the event.
<i>selected</i>	A pointer return value for the is-selected status of the event.
<i>evtype</i>	A stazed parameter for picking either all event or unselected events.

## 13.92.5.167 get\_next\_event() [2/2]

```
bool seq64::sequence::get_next_event (
    midibyte * status,
    midibyte * cc )
```

Then set the status and control character parameters using that event.

## 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.

## 13.92.5.168 get\_next\_trigger()

```
bool seq64::sequence::get_next_trigger (
    midipulse * tick_on,
    midipulse * tick_off,
    bool * selected,
    midipulse * tick_offset )
```

## 13.92.5.169 quantize\_events()

```
void seq64::sequence::quantize_events (
    midibyte status,
    midibyte cc,
    midipulse snap_tick,
    int divide,
    bool linked = false )
```

One confusing things 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.

## 13.92.5.170 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.

## 13.92.5.171 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.

**13.92.5.172** `shift_notes()`

```
void seq64::sequence::shift_notes (
    midipulse ticks )
```

**13.92.5.173** `multiply_pattern()`

```
void seq64::sequence::multiply_pattern (
    double multiplier )
```

**13.92.5.174** `musical_key()` [1/2]

```
midibyte seq64::sequence::musical_key ( ) const [inline]
```

**13.92.5.175** `musical_key()` [2/2]

```
void seq64::sequence::musical_key (
    int key ) [inline]
```

**13.92.5.176** `musical_scale()` [1/2]

```
midibyte seq64::sequence::musical_scale ( ) const [inline]
```

**13.92.5.177** `musical_scale()` [2/2]

```
void seq64::sequence::musical_scale (
    int scale ) [inline]
```

**13.92.5.178 background\_sequence()** [1/2]

```
int seq64::sequence::background_sequence ( ) const [inline]
```

**13.92.5.179 background\_sequence()** [2/2]

```
void seq64::sequence::background_sequence (
    int bs ) [inline]
```

Disabling the sequence number (setting it to SEQ64\_SEQUENCE\_LIMIT) is valid.

**13.92.5.180 show\_events()**

```
void seq64::sequence::show_events ( ) const
```

**13.92.5.181 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.

*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.
------------------	--

**13.92.5.182 note\_off\_margin()**

```
midipulse seq64::sequence::note_off_margin ( ) const [inline]
```

## 13.92.5.183 event\_in\_range()

```
bool seq64::sequence::event_in_range (
    const event & e,
    midibyte status,
    midipulse tick_s,
    midipulse tick_f ) const [inline], [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.
<i>tick</i> ↔ <i>_s</i>	The lower end of the range of timestamps that the event must fall within.
<i>tick</i> ↔ <i>_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.

## 13.92.5.184 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.
----------	---

## 13.92.5.185 put\_event\_on\_bus()

```
void seq64::sequence::put_event_on_bus (
    event & ev ) [private]
```

This function does not bother checking if m\_masterbus is a null pointer.

## Parameters

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

*Threadsafe*

#### 13.92.5.186 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.
-----------------------	---------------------------------

#### 13.92.5.187 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.
------------------	---

#### 13.92.5.188 adjust\_offset()

```
midipulse seq64::sequence::adjust_offset (
    midipulse offset ) [private]
```

#### 13.92.5.189 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.
----------	---

#### 13.92.5.190 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

<i>e</i>	Provides a reference to the event to be removed.
----------	--

#### 13.92.5.191 remove\_all()

```
void seq64::sequence::remove_all ( ) [private]
```

Unsets the modified flag. (Why?) Also see the new `copy_events()` function.

#### 13.92.5.192 channel\_match()

```
bool seq64::sequence::channel_match (
    const event & e ) const [inline], [private]
```

#### Parameters

<i>e</i>	The event whose channel nybble is to be checked.
----------	--

#### Returns

Returns true if the channel-matching feature is enable and the channels match, or true if the channel-matching feature is turned off.

### 13.92.6 Friends And Related Function Documentation

### 13.92.6.1 perform

```
friend class perform [friend]
```

### 13.92.6.2 triggers

```
friend class triggers [friend]
```

## 13.92.7 Field Documentation

### 13.92.7.1 m\_events\_clipboard

```
event_list seq64::sequence::m_events_clipboard [static], [private]
```

Being static allows for copy/paste between patterns.

### 13.92.7.2 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.

### 13.92.7.3 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.

### 13.92.7.4 m\_triggers

```
triggers seq64::sequence::m_triggers [private]
```

### 13.92.7.5 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;
```

### 13.92.7.6 m\_have\_undo

```
bool seq64::sequence::m_have_undo [private]
```

### 13.92.7.7 m\_have\_redo

```
bool seq64::sequence::m_have_redo [private]
```

Previously, unlike the perfedit, the seqedit did not provide a redo facility.

### 13.92.7.8 m\_events\_undo

```
EventStack seq64::sequence::m_events_undo [private]
```

### 13.92.7.9 m\_events\_redo

```
EventStack seq64::sequence::m_events_redo [private]
```

### 13.92.7.10 m\_iterator\_draw

```
event_list::iterator seq64::sequence::m_iterator_draw [private]
```

### 13.92.7.11 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.

### 13.92.7.12 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.

### 13.92.7.13 m\_bus

```
midibyte seq64::sequence::m_bus [private]
```

### 13.92.7.14 m\_song\_mute

```
bool seq64::sequence::m_song_mute [private]
```

### 13.92.7.15 m\_transposable

```
bool seq64::sequence::m_transposable [private]
```

A potential feature from stazed's seq32 project. Now it is an actual, configurable feature.

### 13.92.7.16 m\_notes\_on

```
int seq64::sequence::m_notes_on [private]
```

### 13.92.7.17 m\_masterbus

```
mastermidibus* seq64::sequence::m_masterbus [private]
```

### 13.92.7.18 m\_playing\_notes

```
int seq64::sequence::m_playing_notes[SEQ64_MIDI_NOTES_MAX] [private]
```

It is used when muting, to shut off the notes that are playing.

### 13.92.7.19 m\_was\_playing

```
bool seq64::sequence::m_was_playing [private]
```

### 13.92.7.20 m\_playing

```
bool seq64::sequence::m_playing [private]
```



**13.92.7.21 m\_recording**

```
bool seq64::sequence::m_recording [private]
```

**13.92.7.22 m\_quantized\_rec**

```
bool seq64::sequence::m_quantized_rec [private]
```

**13.92.7.23 m\_thru**

```
bool seq64::sequence::m_thru [private]
```

**13.92.7.24 m\_queued**

```
bool seq64::sequence::m_queued [private]
```

**13.92.7.25 m\_dirty\_main**

```
bool seq64::sequence::m_dirty_main [private]
```

Provides the main dirtiness flag.

**13.92.7.26 m\_dirty\_edit**

```
bool seq64::sequence::m_dirty_edit [private]
```

**13.92.7.27 m\_dirty\_perf**

```
bool seq64::sequence::m_dirty_perf [private]
```

**13.92.7.28 m\_dirty\_names**

```
bool seq64::sequence::m_dirty_names [private]
```

**13.92.7.29 m\_editing**

```
bool seq64::sequence::m_editing [private]
```

**13.92.7.30 m\_raise**

```
bool seq64::sequence::m_raise [private]
```

It allows a sequence editor window to pop up if not already raised, in [seqedit::timeout\(\)](#).

**13.92.7.31 m\_name**

```
std::string seq64::sequence::m_name [private]
```

**13.92.7.32 m\_last\_tick**

```
midipulse seq64::sequence::m_last_tick [private]
```

Provides the last tick played.

**13.92.7.33 m\_queued\_tick**

```
midipulse seq64::sequence::m_queued_tick [private]
```

**13.92.7.34 m\_trigger\_offset**

```
midipulse seq64::sequence::m_trigger_offset [private]
```

**13.92.7.35 m\_maxbeats**

```
const int seq64::sequence::m_maxbeats [private]
```

Hardwired to c\_maxbeats at present.

**13.92.7.36 m\_ppqn**

```
int seq64::sequence::m_ppqn [private]
```

**13.92.7.37 m\_seq\_number**

```
int seq64::sequence::m_seq_number [private]
```

This number is set in the [perform::install\\_sequence\(\)](#) function.

**13.92.7.38 m\_length**

```
midipulse seq64::sequence::m_length [private]
```

This value should be a power of two when used as a bar unit.

**13.92.7.39 m\_snap\_tick**

```
midipulse seq64::sequence::m_snap_tick [private]
```

It starts out as the value `m_ppqn / 4`.

**13.92.7.40 m\_time\_beats\_per\_measure**

```
int 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.

**13.92.7.41 m\_time\_beat\_width**

```
int 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.

**13.92.7.42 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.

**13.92.7.43 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.

#### 13.92.7.44 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.

#### 13.92.7.45 m\_rec\_vol

```
int seq64::sequence::m_rec_vol [private]
```

#### 13.92.7.46 m\_note\_on\_velocity

```
int 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.

#### 13.92.7.47 m\_note\_off\_velocity

```
int seq64::sequence::m_note_off_velocity [private]
```

Currently set to SEQ64\_DEFAULT\_NOTE\_OFF\_VELOCITY, and currently unmodifiable. A "stazed" feature.

#### 13.92.7.48 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.

#### 13.92.7.49 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.

#### 13.92.7.50 m\_background\_sequence

```
int seq64::sequence::m_background_sequence [private]
```

If the value is greater than `max_sequence()`, then there is no background sequence to be set.

## 13.92.7.51 m\_mutex

```
mutex seq64::sequence::m_mutex [mutable], [private]
```

Made mutable for use in certain locked getter functions.

## 13.92.7.52 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.

## 13.93 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*
- [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.*

### 13.93.1 Detailed Description

This class is used in playback, and is contained in the triggers class.

### 13.93.2 Constructor & Destructor Documentation

#### 13.93.2.1 trigger()

```
seq64::trigger::trigger ( ) [inline]
```

### 13.93.3 Member Function Documentation

#### 13.93.3.1 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.

#### 13.93.3.2 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.

#### 13.93.3.3 tick\_start() [1/2]

```
midipulse seq64::trigger::tick_start ( ) const [inline]
```

#### 13.93.3.4 tick\_start() [2/2]

```
void seq64::trigger::tick_start (
    midipulse s ) [inline]
```

#### 13.93.3.5 increment\_tick\_start()

```
void seq64::trigger::increment_tick_start (
    midipulse s ) [inline]
```

#### 13.93.3.6 decrement\_tick\_start()

```
void seq64::trigger::decrement_tick_start (
    midipulse s ) [inline]
```

#### 13.93.3.7 tick\_end() [1/2]

```
midipulse seq64::trigger::tick_end ( ) const [inline]
```

#### 13.93.3.8 tick\_end() [2/2]

```
void seq64::trigger::tick_end (
    midipulse e ) [inline]
```

#### 13.93.3.9 increment\_tick\_end()

```
void seq64::trigger::increment_tick_end (
    midipulse s ) [inline]
```

#### 13.93.3.10 decrement\_tick\_end()

```
void seq64::trigger::decrement_tick_end (
    midipulse s ) [inline]
```

#### 13.93.3.11 offset() [1/2]

```
midipulse seq64::trigger::offset ( ) const [inline]
```

#### 13.93.3.12 offset() [2/2]

```
void seq64::trigger::offset (
    midipulse o ) [inline]
```

#### 13.93.3.13 increment\_offset()

```
void seq64::trigger::increment_offset (
    midipulse s ) [inline]
```

#### 13.93.3.14 decrement\_offset()

```
void seq64::trigger::decrement_offset (
    midipulse s ) [inline]
```

#### 13.93.3.15 selected() [1/2]

```
bool seq64::trigger::selected ( ) const [inline]
```



### 13.93.3.16 selected() [2/2]

```
void seq64::trigger::selected (
    bool s ) [inline]
```

## 13.93.4 Field Documentation

### 13.93.4.1 m\_tick\_start

```
midipulse seq64::trigger::m_tick_start [private]
```

### 13.93.4.2 m\_tick\_end

```
midipulse seq64::trigger::m_tick_end [private]
```

### 13.93.4.3 m\_offset

```
midipulse seq64::trigger::m_offset [private]
```

### 13.93.4.4 m\_selected

```
bool seq64::trigger::m_selected [private]
```

## 13.94 seq64::triggers Class Reference

The triggers class is a receptable the triggers that can be used with a sequence object.

## 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*
- 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)  
*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 [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)  
*Checks the list of triggers against the given tick.*
- bool [select](#) ([midipulse](#) tick)  
*Checks the list of triggers against the given tick.*
- bool [unselect](#) ()  
*Unselects all triggers.*
- bool [intersect](#) ([midipulse](#) position, [midipulse](#) &start, [midipulse](#) &end)  
*This function examines each trigger in the trigger list.*
- 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` ()

*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.*

- 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

- 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.*

- 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 &trig, midipulse splittick)

*Splits the trigger given by the parameter into two triggers.*

## 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.*
- [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](#)

## 13.94.1 Member Typedef Documentation

### 13.94.1.1 List

```
typedef std::list<trigger> seq64::triggers::List [private]
```

### 13.94.1.2 Stack

```
typedef std::stack<List> seq64::triggers::Stack [private]
```

## 13.94.2 Member Enumeration Documentation

### 13.94.2.1 grow\_edit\_t

```
enum seq64::triggers::grow\_edit\_t [private]
```

## Enumerator

GROW_START	Grow the start of the trigger.
GROW_END	Grow the end of the trigger.
GROW_MOVE	Move the entire trigger block.

## 13.94.3 Constructor &amp; Destructor Documentation

## 13.94.3.1 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).
---------------	---

## 13.94.3.2 ~triggers()

```
seq64::triggers::~~triggers ( )
```

## 13.94.4 Member Function Documentation

## 13.94.4.1 operator=()

```
triggers & seq64::triggers::operator= (
    const triggers & rhs )
```

Follows the stock rules for such an operator, but does a little more than 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.

### 13.94.4.2 set\_ppqn()

```
void seq64::triggers::set_ppqn (
    int ppqn ) [inline]
```

### 13.94.4.3 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.

### 13.94.4.4 triggerlist() [1/2]

```
const List& seq64::triggers::triggerlist ( ) const [inline]
```

### 13.94.4.5 triggerlist() [2/2]

```
List& seq64::triggers::triggerlist ( ) [inline]
```

### 13.94.4.6 push\_undo()

```
void seq64::triggers::push_undo ( )
```

### 13.94.4.7 pop\_undo()

```
void seq64::triggers::pop_undo ( )
```

### 13.94.4.8 pop\_redo()

```
void seq64::triggers::pop_redo ( )
```

### 13.94.4.9 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.
----------------	---

## 13.94.4.10 play()

```
bool seq64::triggers::play (
    midipulse & start_tick,
    midipulse & end_tick )
```

This is a new function for [sequence::play\(\)](#) to call.

The for-loop goes through all the triggers, determining if there is a 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.

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.

## Returns

Returns true if we're through playing the frame (trigger turning off), and the caller should stop the playback.

## 13.94.4.11 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 <code>adjust_offset()</code> first. We think that basically makes sure it is positive.

13.94.4.12 `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?

13.94.4.13 `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.
------------------	--

13.94.4.14 `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.

## 13.94.4.15 remove()

```
void seq64::triggers::remove (
    midipulse tick )
```

## Parameters

<i>tick</i>	Provides the tick to be examined.
-------------	-----------------------------------

## 13.94.4.16 get\_state()

```
bool seq64::triggers::get_state (
    midipulse tick )
```

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.

## 13.94.4.17 select()

```
bool seq64::triggers::select (
    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.

**13.94.4.18 unselect()**

```
bool seq64::triggers::unselect ( )
```

**Returns**

Always returns false.

**13.94.4.19 intersect()**

```
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.

**13.94.4.20 remove\_selected()**

```
void seq64::triggers::remove_selected ( )
```

**13.94.4.21 copy\_selected()**

```
void seq64::triggers::copy_selected ( )
```

## 13.94.4.22 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.
-------------------	--

## 13.94.4.23 move\_selected()

```
bool seq64::triggers::move_selected (
    midipulse tick,
    bool fixoffset,
    grow_edit_t which = GROW_MOVE )

    mintick][0          1][maxtick
                2
```

The \a 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.
```

## 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 trigger::grow_edit_t 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 perfrill 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.

## 13.94.4.24 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.

**13.94.4.25 `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.

**13.94.4.26 `get_maximum()`**

```
midipulse seq64::triggers::get_maximum ( )
```

**Returns**

Returns the tick-end for the last trigger, if available. Otherwise, 0 is returned.

**13.94.4.27 `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.

**13.94.4.28 `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.

#### 13.94.4.29 clear()

```
void seq64::triggers::clear ( ) [inline]
```

#### 13.94.4.30 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.

#### 13.94.4.31 next\_trigger()

```
trigger seq64::triggers::next_trigger ( )
```

##### Returns

Returns the next trigger. If there is none, a default trigger object is returned.

#### 13.94.4.32 reset\_draw\_trigger\_marker()

```
void seq64::triggers::reset_draw_trigger_marker ( ) [inline]
```

#### 13.94.4.33 set\_trigger\_paste\_tick()

```
void seq64::triggers::set_trigger_paste_tick (
    midipulse tick ) [inline]
```

#### 13.94.4.34 get\_trigger\_paste\_tick()

```
midipulse seq64::triggers::get_trigger_paste_tick ( ) const [inline]
```

#### 13.94.4.35 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.

#### 13.94.4.36 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.

### 13.94.5 Friends And Related Function Documentation

#### 13.94.5.1 midi\_container

```
friend class midi_container [friend]
```

#### 13.94.5.2 midifile

```
friend class midifile [friend]
```

#### 13.94.5.3 sequence

```
friend class sequence [friend]
```

#### 13.94.5.4 Seq24PerfInput

```
friend class Seq24PerfInput [friend]
```

#### 13.94.5.5 FruityPerfInput

```
friend class FruityPerfInput [friend]
```

### 13.94.6 Field Documentation

#### 13.94.6.1 m\_parent

```
sequence& seq64::triggers::m_parent [private]
```

#### 13.94.6.2 m\_triggers

```
List seq64::triggers::m_triggers [private]
```



#### 13.94.6.3 m\_clipboard

`trigger` seq64::triggers::m\_clipboard [private]

#### 13.94.6.4 m\_undo\_stack

`Stack` seq64::triggers::m\_undo\_stack [private]

#### 13.94.6.5 m\_redo\_stack

`Stack` seq64::triggers::m\_redo\_stack [private]

#### 13.94.6.6 m\_iterator\_play\_trigger

`List::iterator` seq64::triggers::m\_iterator\_play\_trigger [private]

#### 13.94.6.7 m\_iterator\_draw\_trigger

`List::iterator` seq64::triggers::m\_iterator\_draw\_trigger [private]

#### 13.94.6.8 m\_trigger\_copied

`bool` seq64::triggers::m\_trigger\_copied [private]

#### 13.94.6.9 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.

#### 13.94.6.10 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.

## 13.94.6.11 m\_length

```
int seq64::triggers::m_length [private]
```

This might change, we're not yet sure.

## 13.95 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.*

## 13.95.1 Detailed Description

Will later make the size adjustable, if it makes sense to do so.

## 13.95.2 Constructor & Destructor Documentation

### 13.95.2.1 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.
-------------	--

### 13.95.2.2 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.
------------	---------------------------------------

## 13.95.3 Member Function Documentation

**13.95.3.1 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.

**13.95.3.2 is\_valid()**

```
bool seq64::user_instrument::is_valid ( ) const [inline]
```

**13.95.3.3 set\_defaults()**

```
void seq64::user_instrument::set_defaults ( )
```

Also invalidates the object.

**13.95.3.4 name()**

```
const std::string& seq64::user_instrument::name ( ) const [inline]
```

**13.95.3.5 controller\_count()**

```
int seq64::user_instrument::controller_count ( ) const [inline]
```

**13.95.3.6 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).

**13.95.3.7 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.

## 13.95.3.8 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.

## 13.95.3.9 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.

## 13.95.3.10 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.
-----------------	--

13.95.3.11 `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.
------------	---

## 13.95.4 Field Documentation

13.95.4.1 `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.

13.95.4.2 `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.

13.95.4.3 `m_instrument_def`

```
user_instrument_t seq64::user_instrument::m_instrument_def [private]
```

13.96 `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.*

### 13.96.1 Field Documentation

#### 13.96.1.1 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.

#### 13.96.1.2 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.

#### 13.96.1.3 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.

## 13.97 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*
- `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.*

### 13.97.1 Detailed Description

Will later make the size adjustable, if it makes sense to do so.

### 13.97.2 Constructor & Destructor Documentation

#### 13.97.2.1 `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.
-------------	--

## 13.97.2.2 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.
------------	---------------------------------------

## 13.97.3 Member Function Documentation

## 13.97.3.1 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.

## 13.97.3.2 is\_valid()

```
bool seq64::user_midi_bus::is_valid ( ) const [inline]
```

## 13.97.3.3 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).

**13.97.3.4 name()**

```
const std::string& seq64::user_midi_bus::name ( ) const [inline]
```

**13.97.3.5 channel\_count()**

```
int seq64::user_midi_bus::channel_count ( ) const [inline]
```

**Returns**

This function returns the number of channels. Basically this value is always the same as that returned by [channel\\_max\(\)](#), but this pair of functions is consistent with the count functions in the [user\\_instrument](#) class.

**13.97.3.6 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).

**13.97.3.7 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.

**13.97.3.8 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.

#### 13.97.3.9 set\_name()

```
void seq64::user_midi_bus::set_name (
    const std::string & name ) [inline], [private]
```

#### 13.97.3.10 copy\_definitions()

```
void seq64::user_midi_bus::copy_definitions (
    const user_midi_bus & rhs ) [private]
```

Does not include the validity flag.

### 13.97.4 Field Documentation

#### 13.97.4.1 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.

#### 13.97.4.2 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.

#### 13.97.4.3 m\_midi\_bus\_def

```
user_midi_bus_t seq64::user_midi_bus::m_midi_bus_def [private]
```

## 13.98 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.*

### 13.98.1 Field Documentation

#### 13.98.1.1 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.

#### 13.98.1.2 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" section in the "user" configuration file.

## 13.99 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.*
- 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*
- 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 [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*
- 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\\_beats\\_per\\_minute](#) () const
  - 'Getter' function for member m\_midi\_beats\_per\_minute*
- 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*
- `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 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*

## Protected Member Functions

- `void grid_brackets (int thickness)`  
*'Getter' function for member m\_grid\_brackets*
- `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 10, inclusive.*
- void [max\\_sets](#) (int value)  
*'Setter' function for member m\_max\_sets This value is not modified unless the value parameter is between 32 and 64, inclusive.*
- 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\\_beats\\_per\\_minute](#) ([midibpm](#) beatsperminute)  
*'Setter' function for member m\_midi\_beats\_minute 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.*
- [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.*
- 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\\_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).*
- int [m\\_midi\\_beats\\_per\\_minute](#)  
*Provides the universal and unambiguous MIDI value for beats per minute (BPM).*
- 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 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 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](#)  
*Area of what? Doesn't look at all like it is based on the size of characters.*
- int [m\\_seqarea\\_seq\\_y](#)
- int [m\\_mainwid\\_x](#)  
*The width of the main pattern/sequence grid, in pixels.*
- int [m\\_mainwid\\_y](#)
- 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.*

## Friends

- class [userfile](#)

### 13.99.1 Detailed Description

These settings will eventually be made part of the "user" settings file.

### 13.99.2 Member Typedef Documentation

#### 13.99.2.1 Busses

```
typedef std::vector<user\_midi\_bus> seq64::user\_settings::Busses [private]
```

Internal type for the container of [user\\_midi\\_bus](#) objects. Sorry about the "confusion" about "bus" versus "buss". See Google for arguments about it.

#### 13.99.2.2 BussIterator

```
typedef std::vector<user\_midi\_bus>::iterator seq64::user\_settings::BussIterator [private]
```

### 13.99.2.3 BussConstIterator

```
typedef std::vector<user_midi_bus>::const_iterator seq64::user_settings::BussConstIterator
[private]
```

### 13.99.2.4 Instruments

```
typedef std::vector<user_instrument> seq64::user_settings::Instruments [private]
```

Internal type for the container of `user_instrument` objects.

### 13.99.2.5 InstrumentIterator

```
typedef std::vector<user_instrument>::iterator seq64::user_settings::InstrumentIterator [private]
```

### 13.99.2.6 InstrumentConstIterator

```
typedef std::vector<user_instrument>::const_iterator seq64::user_settings::InstrumentConst←
Iterator [private]
```

## 13.99.3 Member Enumeration Documentation

### 13.99.3.1 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 value.

## 13.99.4 Constructor & Destructor Documentation

#### 13.99.4.1 `user_settings()` [1/2]

```
seq64::user_settings::user_settings ( )
```

#### 13.99.4.2 `user_settings()` [2/2]

```
seq64::user_settings::user_settings (
    const user_settings & rhs )
```

### 13.99.5 Member Function Documentation

#### 13.99.5.1 `operator=()`

```
user_settings & seq64::user_settings::operator= (
    const user_settings & rhs )
```

#### 13.99.5.2 `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!

#### 13.99.5.3 `normalize()`

```
void seq64::user_settings::normalize ( )
```

Should we normalize the BPM increment values here, in case they are irregular?

#### 13.99.5.4 `add_bus()`

```
bool seq64::user_settings::add_bus (
    const std::string & alias )
```

#### 13.99.5.5 add\_instrument()

```
bool seq64::user_settings::add_instrument (
    const std::string & instname )
```

#### 13.99.5.6 bus()

```
const user_midi_bus& seq64::user_settings::bus (
    int index ) [inline]
```

Cannot append the const specifier.

#### 13.99.5.7 instrument()

```
const user_instrument& seq64::user_settings::instrument (
    int index ) [inline]
```

Cannot append the const specifier.

#### 13.99.5.8 bus\_count()

```
int seq64::user_settings::bus_count ( ) const [inline]
```

#### 13.99.5.9 set\_bus\_instrument()

```
void seq64::user_settings::set_bus_instrument (
    int index,
    int channel,
    int instrum )
```

#### 13.99.5.10 bus\_instrument()

```
int seq64::user_settings::bus_instrument (
    int buss,
    int channel ) [inline]
```

#### 13.99.5.11 bus\_name()

```
const std::string& seq64::user_settings::bus_name (
    int buss ) [inline]
```

**13.99.5.12 instrument\_count()**

```
int seq64::user_settings::instrument_count ( ) const [inline]
```

**13.99.5.13 set\_instrument\_controllers()**

```
void seq64::user_settings::set_instrument_controllers (
    int index,
    int cc,
    const std::string & ccname,
    bool isactive )
```

**13.99.5.14 instrument\_name()** [1/2]

```
const std::string& seq64::user_settings::instrument_name (
    int instrum ) [inline]
```

**13.99.5.15 instrument\_name()** [2/2]

```
const std::string& seq64::user_settings::instrument_name (
    int buss,
    int channel ) [inline]
```

**13.99.5.16 instrument\_controller\_active()**

```
bool seq64::user_settings::instrument_controller_active (
    int instrum,
    int cc ) [inline]
```

**13.99.5.17 controller\_active()**

```
bool seq64::user_settings::controller_active (
    int buss,
    int channel,
    int cc ) [inline]
```

It also has a shorter name.



**13.99.5.18 instrument\_controller\_name()**

```
const std::string& seq64::user_settings::instrument_controller_name (
    int instrum,
    int cc ) [inline]
```

**13.99.5.19 controller\_name()**

```
const std::string& seq64::user_settings::controller_name (
    int buss,
    int channel,
    int cc ) [inline]
```

It also has a shorter name.

**13.99.5.20 grid\_style()** [1/2]

```
int seq64::user_settings::grid_style ( ) const [inline]
```

**13.99.5.21 grid\_is\_normal()**

```
bool seq64::user_settings::grid_is_normal ( ) const [inline]
```

**13.99.5.22 grid\_is\_white()**

```
bool seq64::user_settings::grid_is_white ( ) const [inline]
```

**13.99.5.23 grid\_is\_black()**

```
bool seq64::user_settings::grid_is_black ( ) const [inline]
```

**13.99.5.24 grid\_brackets()** [1/2]

```
int seq64::user_settings::grid_brackets ( ) const [inline]
```

**13.99.5.25 mainwnd\_rows()** [1/2]

```
int seq64::user_settings::mainwnd_rows ( ) const [inline]
```

**13.99.5.26 mainwnd\_cols()** [1/2]

```
int seq64::user_settings::mainwnd_cols ( ) const [inline]
```

**13.99.5.27 seqs\_in\_set()**

```
int seq64::user_settings::seqs_in_set ( ) const [inline]
```

**13.99.5.28 gmute\_tracks()**

```
int seq64::user_settings::gmute_tracks ( ) const [inline]
```

**13.99.5.29 max\_sets()** [1/2]

```
int seq64::user_settings::max_sets ( ) const [inline]
```

**13.99.5.30 max\_sequence()**

```
int seq64::user_settings::max_sequence ( ) const [inline]
```

**13.99.5.31 text\_x()** [1/2]

```
int seq64::user_settings::text_x ( ) const [inline]
```

**13.99.5.32 text\_y()** [1/2]

```
int seq64::user_settings::text_y ( ) const [inline]
```

**13.99.5.33 seqchars\_x()** [1/2]

```
int seq64::user_settings::seqchars_x ( ) const [inline]
```

**13.99.5.34 seqchars\_y()** [1/2]

```
int seq64::user_settings::seqchars_y ( ) const [inline]
```

**13.99.5.35 seqarea\_x()** [1/2]

```
int seq64::user_settings::seqarea_x ( ) const [inline]
```

**13.99.5.36 seqarea\_y()** [1/2]

```
int seq64::user_settings::seqarea_y ( ) const [inline]
```

**13.99.5.37 seqarea\_seq\_x()** [1/2]

```
int seq64::user_settings::seqarea_seq_x ( ) const [inline]
```

**13.99.5.38 seqarea\_seq\_y()** [1/2]

```
int seq64::user_settings::seqarea_seq_y ( ) const [inline]
```

**13.99.5.39 mainwid\_border()** [1/2]

```
int seq64::user_settings::mainwid_border ( ) const [inline]
```

**13.99.5.40 mainwid\_spacing()** [1/2]

```
int seq64::user_settings::mainwid_spacing ( ) const [inline]
```

**13.99.5.41 mainwid\_x()**

```
int seq64::user_settings::mainwid_x ( ) const [inline]
```

**13.99.5.42 mainwid\_y()**

```
int seq64::user_settings::mainwid_y ( ) const [inline]
```

**13.99.5.43 control\_height()** [1/2]

```
int seq64::user_settings::control_height ( ) const [inline]
```

**13.99.5.44 zoom()** [1/2]

```
int seq64::user_settings::zoom ( ) const [inline]
```

**13.99.5.45 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.

**13.99.5.46 global\_seq\_feature()** [1/2]

```
bool seq64::user_settings::global_seq_feature ( ) const [inline]
```

**13.99.5.47 global\_seq\_feature()** [2/2]

```
void seq64::user_settings::global_seq_feature (
    bool flag ) [inline]
```

**13.99.5.48 seqedit\_scale()** [1/2]

```
int seq64::user_settings::seqedit_scale ( ) const [inline]
```

**13.99.5.49 seqedit\_scale()** [2/2]

```
void seq64::user_settings::seqedit_scale (
    int scale ) [inline]
```

**13.99.5.50 seqedit\_key()** [1/2]

```
int seq64::user_settings::seqedit_key ( ) const [inline]
```

**13.99.5.51 seqedit\_key()** [2/2]

```
void seq64::user_settings::seqedit_key (
    int key ) [inline]
```

**13.99.5.52 seqedit\_bgsequence()** [1/2]

```
int seq64::user_settings::seqedit_bgsequence ( ) const [inline]
```

**13.99.5.53 seqedit\_bgsequence()** [2/2]

```
void seq64::user_settings::seqedit_bgsequence (
    int seqnum ) [inline]
```

**13.99.5.54 use\_new\_font()** [1/2]

```
bool seq64::user_settings::use_new_font ( ) const [inline]
```

**13.99.5.55 allow\_two\_perfedits()** [1/2]

```
bool seq64::user_settings::allow_two_perfedits ( ) const [inline]
```

**13.99.5.56 perf\_h\_page\_increment()** [1/2]

```
int seq64::user_settings::perf_h_page_increment ( ) const [inline]
```

**13.99.5.57 perf\_v\_page\_increment()** [1/2]

```
int seq64::user_settings::perf_v_page_increment ( ) const [inline]
```

**13.99.5.58 progress\_bar\_colored()** [1/2]

```
int seq64::user_settings::progress_bar_colored ( ) const [inline]
```

**13.99.5.59 progress\_bar\_thick()** [1/2]

```
bool seq64::user_settings::progress_bar_thick ( ) const [inline]
```

**13.99.5.60 inverse\_colors()** [1/2]

```
bool seq64::user_settings::inverse_colors ( ) const [inline]
```

**13.99.5.61 window\_redraw\_rate()** [1/2]

```
int seq64::user_settings::window_redraw_rate ( ) const [inline]
```

**13.99.5.62 use\_more\_icons()** [1/2]

```
bool seq64::user_settings::use_more_icons ( ) const [inline]
```

**13.99.5.63 save\_user\_config()** [1/2]

```
bool seq64::user_settings::save_user_config ( ) const [inline]
```

**13.99.5.64 save\_user\_config()** [2/2]

```
void seq64::user_settings::save_user_config (
    bool flag ) [inline]
```

**13.99.5.65 grid\_brackets()** [2/2]

```
void seq64::user_settings::grid_brackets (
    int thickness ) [inline], [protected]
```

**13.99.5.66 grid\_style()** [2/2]

```
void seq64::user_settings::grid_style (
    int gridstyle ) [protected]
```

**13.99.5.67 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.

**13.99.5.68 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.

**13.99.5.69 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.

**13.99.5.70 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.

**13.99.5.71 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.

**13.99.5.72 seqchars\_x()** [2/2]

```
void seq64::user_settings::seqchars_x (
    int value ) [protected]
```

**13.99.5.73 seqchars\_y()** [2/2]

```
void seq64::user_settings::seqchars_y (
    int value ) [protected]
```

**13.99.5.74 seqarea\_x()** [2/2]

```
void seq64::user_settings::seqarea_x (
    int value ) [protected]
```

**13.99.5.75 seqarea\_y()** [2/2]

```
void seq64::user_settings::seqarea_y (
    int value ) [protected]
```

**13.99.5.76 seqarea\_seq\_x()** [2/2]

```
void seq64::user_settings::seqarea_seq_x (
    int value ) [protected]
```



**13.99.5.77 seqarea\_seq\_y()** [2/2]

```
void seq64::user_settings::seqarea_seq_y (
    int value ) [protected]
```

**13.99.5.78 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.

**13.99.5.79 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.

**13.99.5.80 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.

**13.99.5.81 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.

**13.99.5.82 midi\_ppqn()** [1/2]

```
int seq64::user_settings::midi_ppqn ( ) const [inline]
```

**13.99.5.83 midi\_beats\_per\_bar()** [1/2]

```
int seq64::user_settings::midi_beats_per_bar ( ) const [inline]
```

**13.99.5.84 midi\_beats\_per\_minute()** [1/2]

```
midibpm seq64::user_settings::midi_beats_per_minute ( ) const [inline]
```

**13.99.5.85 midi\_beat\_width()** [1/2]

```
int seq64::user_settings::midi_beat_width ( ) const [inline]
```

**13.99.5.86 midi\_buss\_override()** [1/2]

```
char seq64::user_settings::midi_buss_override ( ) const [inline]
```

**13.99.5.87 velocity\_override()** [1/2]

```
int seq64::user_settings::velocity_override ( ) const [inline]
```

**13.99.5.88 bpm\_precision()** [1/2]

```
int seq64::user_settings::bpm_precision ( ) const [inline]
```

**13.99.5.89 bpm\_step\_increment()** [1/2]

```
midibpm seq64::user_settings::bpm_step_increment ( ) const [inline]
```

**13.99.5.90 bpm\_page\_increment()** [1/2]

```
midibpm seq64::user_settings::bpm_page_increment ( ) const [inline]
```

**13.99.5.91 min\_zoom()**

```
int seq64::user_settings::min_zoom ( ) const [inline]
```

**13.99.5.92 max\_zoom()**

```
int seq64::user_settings::max_zoom ( ) const [inline]
```

**13.99.5.93 baseline\_ppqn()**

```
int seq64::user_settings::baseline_ppqn ( ) const [inline]
```

**13.99.5.94 use\_new\_font()** [2/2]

```
void seq64::user_settings::use_new_font (
    bool flag ) [inline]
```

**13.99.5.95 allow\_two\_perfedits()** [2/2]

```
void seq64::user_settings::allow_two_perfedits (
    bool flag ) [inline]
```

**13.99.5.96 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)

**13.99.5.97 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)

**13.99.5.98 progress\_bar\_colored()** [2/2]

```
void seq64::user_settings::progress_bar_colored (
    int palcode ) [inline]
```

**13.99.5.99 progress\_bar\_thick()** [2/2]

```
void seq64::user_settings::progress_bar_thick (
    bool flag ) [inline]
```

**13.99.5.100 inverse\_colors()** [2/2]

```
void seq64::user_settings::inverse_colors (
    bool flag ) [inline]
```

**13.99.5.101 window\_redraw\_rate()** [2/2]

```
void seq64::user_settings::window_redraw_rate (
    int ms ) [inline]
```

**13.99.5.102 use\_more\_icons()** [2/2]

```
void seq64::user_settings::use_more_icons (
    bool flag ) [inline]
```

**13.99.5.103 midi\_ppqn()** [2/2]

```
void seq64::user_settings::midi_ppqn (
    int value )
```

The default value is 192.

**13.99.5.104 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".)

**13.99.5.105 velocity\_override()** [2/2]

```
void seq64::user_settings::velocity_override (
    int vel )
```

**13.99.5.106 bpm\_precision()** [2/2]

```
void seq64::user_settings::bpm_precision (
    int precision )
```

**13.99.5.107 bpm\_step\_increment()** [2/2]

```
void seq64::user_settings::bpm_step_increment (
    midibpm increment )
```

**13.99.5.108 bpm\_page\_increment()** [2/2]

```
void seq64::user_settings::bpm_page_increment (
    midibpm increment )
```

**13.99.5.109 midi\_beats\_per\_bar()** [2/2]

```
void seq64::user_settings::midi_beats_per_bar (
    int value ) [protected]
```

The default value is 4.

**13.99.5.110 midi\_beats\_per\_minute()** [2/2]

```
void seq64::user_settings::midi_beats_per_minute (
    midibpm value ) [protected]
```

The default value is 120.

**13.99.5.111 midi\_beat\_width()** [2/2]

```
void seq64::user_settings::midi_beat_width (
    int bw ) [protected]
```

The default value is 4.

**13.99.5.112 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.

**13.99.5.113 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.

**13.99.6 Friends And Related Function Documentation****13.99.6.1 userfile**

```
friend class userfile [friend]
```

**13.99.7 Field Documentation****13.99.7.1 m\_midi\_buses**

```
Busses seq64::user_settings::m_midi_buses [private]
```

Since this object is a vector, its size is adjustable.

**13.99.7.2 m\_instruments**

```
Instruments seq64::user_settings::m_instruments [private]
```

The size is adjustable, and grows as objects are added.

**13.99.7.3 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.

#### 13.99.7.4 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.

#### 13.99.7.5 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.

#### 13.99.7.6 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".

#### 13.99.7.7 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.

#### 13.99.7.8 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.

#### 13.99.7.9 m\_mainwid\_spacing

```
int seq64::user_settings::m_mainwid_spacing [private]
```

#### 13.99.7.10 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.

### 13.99.7.11 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.

### 13.99.7.12 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.

### 13.99.7.13 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`.

### 13.99.7.14 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`.

### 13.99.7.15 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`.



#### 13.99.7.16 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.

#### 13.99.7.17 m\_allow\_two\_perfedits

```
bool seq64::user_settings::m_allow_two_perfedits [private]
```

Defaults to true.

#### 13.99.7.18 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.

#### 13.99.7.19 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.

#### 13.99.7.20 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.

#### 13.99.7.21 m\_progress\_bar\_thick

```
bool seq64::user_settings::m_progress_bar_thick [private]
```

2 pixels. It isn't useful to support anything thicker.

#### 13.99.7.22 m\_inverse\_colors

```
bool seq64::user_settings::m_inverse_colors [private]
```

Not all colors are reversed, though.

### 13.99.7.23 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.

### 13.99.7.24 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.

### 13.99.7.25 m\_text\_x

```
int seq64::user_settings::m_text_x [private]
```

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.

### 13.99.7.26 m\_text\_y

```
int seq64::user_settings::m_text_y [private]
```

### 13.99.7.27 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.

### 13.99.7.28 m\_seqchars\_y

```
int seq64::user_settings::m_seqchars_y [private]
```

### 13.99.7.29 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`.

### 13.99.7.30 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.

### 13.99.7.31 m\_midi\_beats\_per\_minute

```
int 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.

### 13.99.7.32 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.

### 13.99.7.33 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`.

### 13.99.7.34 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.

### 13.99.7.35 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.

### 13.99.7.36 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.

### 13.99.7.37 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.

### 13.99.7.38 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 "user" 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;
```

### 13.99.7.39 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.

### 13.99.7.40 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.

### 13.99.7.41 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;
```

### 13.99.7.42 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.

### 13.99.7.43 m\_seqarea\_y

```
int seq64::user_settings::m_seqarea_y [private]
```

### 13.99.7.44 m\_seqarea\_seq\_x

```
int seq64::user_settings::m_seqarea_seq_x [private]
```

These are used only in the mainwid module.

**13.99.7.45 m\_seqarea\_seq\_y**

```
int seq64::user_settings::m_seqarea_seq_y [private]
```

**13.99.7.46 m\_mainwid\_x**

```
int seq64::user_settings::m_mainwid_x [private]
```

Affected by the `m_mainwid_border` and `m_mainwid_spacing` values.

```
c_mainwid_x =  
(  
    (c_seqarea_x + c_mainwid_spacing) * c_mainwnd_cols -  
    c_mainwid_spacing + c_mainwid_border * 2  
);
```

**13.99.7.47 m\_mainwid\_y**

```
int seq64::user_settings::m_mainwid_y [private]
```

**13.99.7.48 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.

**13.99.7.49 mc\_min\_zoom**

```
const int seq64::user_settings::mc_min_zoom [private]
```

It's value is 1.

**13.99.7.50 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.

## 13.99.7.51 mc\_baseline\_ppqn

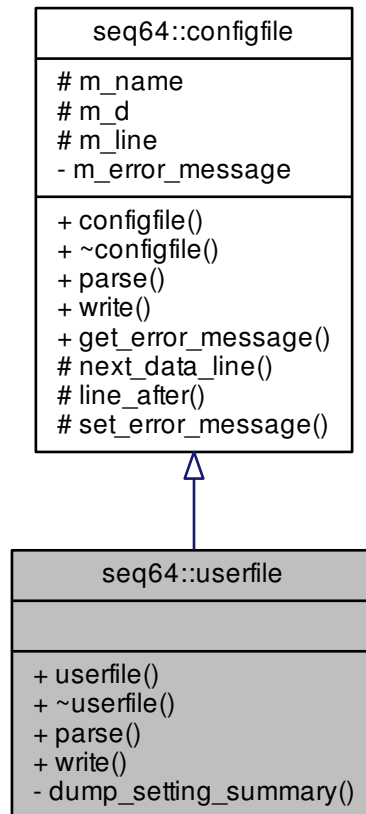
```
const int seq64::user_settings::mc_baseline_ppqn [private]
```

This value is necessary in order to keep user-interface elements stable when different PPQNs are used. It is set to DEFAULT\_PPQN.

## 13.100 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

### 13.100.1 Constructor & Destructor Documentation

#### 13.100.1.1 `userfile()`

```
seq64::userfile::userfile (
    const std::string & name )
```

##### Parameters

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

#### 13.100.1.2 `~userfile()`

```
seq64::userfile::~~userfile ( )
```

### 13.100.2 Member Function Documentation

#### 13.100.2.1 `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](#).

## 13.100.2.2 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](#).

## 13.100.2.3 dump\_setting\_summary()

```
void seq64::userfile::dump_setting_summary ( ) [private]
```

Does work only if PLATFORM\_DEBUG is defined; see the [user\\_settings](#) class.



# Index

- ~AbstractPerfInput
  - seq64::AbstractPerfInput, [134](#)
- ~automutex
  - seq64::automutex, [138](#)
- ~busarray
  - seq64::busarray, [140](#)
- ~businfo
  - seq64::businfo, [150](#)
- ~configfile
  - seq64::configfile, [163](#)
- ~editable\_event
  - seq64::editable\_event, [171](#)
- ~editable\_events
  - seq64::editable\_events, [182](#)
- ~event
  - seq64::event, [193](#)
- ~event\_list
  - seq64::event\_list, [214](#)
- ~eventedit
  - seq64::eventedit, [228](#)
- ~eventslots
  - seq64::eventslots, [244](#)
- ~gui\_assistant
  - seq64::gui\_assistant, [276](#)
- ~gui\_assistant\_gtk2
  - seq64::gui\_assistant\_gtk2, [279](#)
- ~gui\_drawingarea\_gtk2
  - seq64::gui\_drawingarea\_gtk2, [284](#)
- ~gui\_palette\_gtk2
  - seq64::gui\_palette\_gtk2, [302](#)
- ~gui\_window\_gtk2
  - seq64::gui\_window\_gtk2, [312](#)
- ~jack\_assistant
  - seq64::jack\_assistant, [318](#)
- ~keys\_perform
  - seq64::keys\_perform, [352](#)
- ~keys\_perform\_gtk2
  - seq64::keys\_perform\_gtk2, [376](#)
- ~lfownd
  - seq64::lfownd, [392](#)
- ~maintime
  - seq64::maintime, [397](#)
- ~mainwid
  - seq64::mainwid, [404](#)
- ~mainwnd
  - seq64::mainwnd, [421](#)
- ~mastermidibase
  - seq64::mastermidibase, [439](#)
- ~mastermidibus
  - seq64::mastermidibus, [459](#)
- ~midi\_alsa
  - seq64::midi\_alsa, [469](#)
- ~midi\_alsa\_info
  - seq64::midi\_alsa\_info, [478](#)
- ~midi\_api
  - seq64::midi\_api, [484](#)
- ~midi\_container
  - seq64::midi\_container, [494](#)
- ~midi\_in\_jack
  - seq64::midi\_in\_jack, [510](#)
- ~midi\_info
  - seq64::midi\_info, [515](#)
- ~midi\_jack
  - seq64::midi\_jack, [527](#)
- ~midi\_jack\_data
  - seq64::midi\_jack\_data, [537](#)
- ~midi\_jack\_info
  - seq64::midi\_jack\_info, [541](#)
- ~midi\_list
  - seq64::midi\_list, [549](#)
- ~midi\_out\_jack
  - seq64::midi\_out\_jack, [560](#)
- ~midi\_queue
  - seq64::midi\_queue, [566](#)
- ~midi\_splitter
  - seq64::midi\_splitter, [571](#)
- ~midi\_vector
  - seq64::midi\_vector, [580](#)
- ~midibase
  - seq64::midibase, [588](#)
- ~midibus
  - seq64::midibus, [608](#)
- ~midifile
  - seq64::midifile, [620](#)
- ~optionsfile
  - seq64::optionsfile, [647](#)
- ~perfedit
  - seq64::perfedit, [655](#)
- ~perfnames
  - seq64::perfnames, [671](#)
- ~perform
  - seq64::perform, [691](#)
- ~perfroll
  - seq64::perfroll, [769](#)
- ~perftime
  - seq64::perftime, [787](#)
- ~rterror
  - seq64::rterror, [820](#)

- ~rtmidi
  - seq64::rtmidi, [824](#)
- ~rtmidi\_in
  - seq64::rtmidi\_in, [832](#)
- ~rtmidi\_info
  - seq64::rtmidi\_info, [842](#)
- ~rtmidi\_out
  - seq64::rtmidi\_out, [851](#)
- ~seqdata
  - seq64::seqdata, [862](#)
- ~seqedit
  - seq64::seqedit, [879](#)
- ~seqevent
  - seq64::seqevent, [907](#)
- ~seqkeys
  - seq64::seqkeys, [922](#)
- ~seqmenu
  - seq64::seqmenu, [935](#)
- ~seqroll
  - seq64::seqroll, [950](#)
- ~seqtime
  - seq64::seqtime, [979](#)
- ~sequence
  - seq64::sequence, [994](#)
- ~triggers
  - seq64::triggers, [1061](#)
- ~userfile
  - seq64::userfile, [1120](#)
- about\_dialog
  - seq64::mainwnd, [426](#)
- AbstractPerfInput
  - seq64::AbstractPerfInput, [134](#)
- action
  - seq64::midi\_control, [502](#)
- activate
  - seq64::businfo, [152](#)
  - seq64::jack\_assistant, [322](#)
  - seq64::mastermidibase, [450](#)
  - seq64::mastermidibus, [463](#)
  - seq64::perform, [735](#)
- activate\_adding
  - seq64::AbstractPerfInput, [135](#)
  - seq64::FruityPerfInput, [268](#)
  - seq64::Seq24PerfInput, [854](#)
- active
  - seq64::businfo, [150](#)
  - seq64::midi\_control, [503](#)
- add
  - seq64::busarray, [140](#), [141](#)
  - seq64::editable\_events, [185](#)
  - seq64::event\_list, [215](#)
  - seq64::midi\_jack\_info, [545](#)
  - seq64::midi\_port\_info, [562](#)
  - seq64::midi\_queue, [566](#)
  - seq64::triggers, [1063](#)
- add\_bus
  - seq64::midi\_info, [520](#)
  - seq64::rtmidi\_info, [843](#)
  - seq64::user\_settings, [1094](#)
- add\_chord
  - seq64::seqroll, [951](#)
  - seq64::sequence, [1011](#)
- add\_clock
  - seq64::perform, [746](#)
- add\_event
  - seq64::midi\_container, [497](#)
  - seq64::sequence, [1011](#), [1012](#)
- add\_extended\_keys\_page
  - seq64::options, [644](#)
- add\_input
  - seq64::perform, [746](#)
  - seq64::rtmidi\_info, [843](#)
- add\_instrument
  - seq64::user\_settings, [1094](#)
- add\_jack\_sync\_page
  - seq64::options, [644](#)
- add\_keyboard\_page
  - seq64::options, [644](#)
- add\_long
  - seq64::midi\_container, [497](#)
- add\_midi\_clock\_page
  - seq64::options, [643](#)
- add\_midi\_input\_page
  - seq64::options, [644](#)
- add\_mouse\_page
  - seq64::options, [644](#)
- add\_note
  - seq64::seqroll, [951](#)
  - seq64::sequence, [1010](#)
- add\_output
  - seq64::rtmidi\_info, [843](#)
- add\_sequence
  - seq64::perform, [701](#)
- add\_short
  - seq64::midi\_container, [497](#)
- add\_trigger
  - seq64::midifile, [626](#)
  - seq64::sequence, [1013](#)
- add\_variable
  - seq64::midi\_container, [496](#)
- adding
  - seq64::seqroll, [965](#)
- adj\_callback\_bpm
  - seq64::mainwnd, [422](#)
- adj\_callback\_ss
  - seq64::mainwnd, [422](#)
- adjust\_offset
  - seq64::sequence, [1044](#)
  - seq64::triggers, [1071](#)
- adjust\_offsets\_to\_length
  - seq64::triggers, [1064](#)
- adjust\_timestamp
  - seq64::sequence, [1027](#)
- adjust\_trigger\_offsets\_to\_length
  - seq64::sequence, [1044](#)
- adjustment\_dummy

- seq64, [102](#)
- alias
  - seq64::user\_midi\_bus\_t, [1084](#)
- align\_selection
  - seq64::seqroll, [962](#)
- all\_notes\_off
  - seq64::perform, [737](#)
- allocate
  - seq64::midi\_queue, [567](#)
- allow\_click\_edit
  - seq64::rc\_settings, [802](#), [806](#)
- allow\_mod4\_mode
  - seq64::rc\_settings, [802](#), [806](#)
- allow\_snap\_split
  - seq64::rc\_settings, [802](#), [806](#)
- allow\_two\_perfeditis
  - seq64::user\_settings, [1101](#), [1107](#)
- analyze
  - seq64::editable\_event, [176](#)
- any\_group\_unmutes
  - seq64::perform, [732](#)
- any\_selected\_notes
  - seq64::event\_list, [220](#)
  - seq64::sequence, [995](#)
- api\_clock
  - seq64::mastermidibase, [452](#)
  - seq64::midi\_alsa, [473](#)
  - seq64::midi\_api, [488](#)
  - seq64::midi\_jack, [534](#)
  - seq64::midibase, [601](#)
  - seq64::midibus, [612](#), [614](#)
  - seq64::rtmidi, [825](#)
- api\_connect
  - seq64::midi\_api, [485](#)
  - seq64::midi\_info, [517](#)
  - seq64::midi\_jack, [530](#)
  - seq64::midi\_jack\_info, [542](#)
  - seq64::midibus, [612](#)
  - seq64::rtmidi, [824](#)
  - seq64::rtmidi\_info, [847](#)
- api\_continue\_from
  - seq64::mastermidibase, [451](#)
  - seq64::mastermidibus, [463](#)
  - seq64::midi\_alsa, [473](#)
  - seq64::midi\_api, [487](#)
  - seq64::midi\_jack, [533](#)
  - seq64::midibase, [600](#)
  - seq64::midibus, [611](#), [614](#)
  - seq64::rtmidi, [825](#)
- api\_data
  - seq64::rtmidi\_in\_data, [837](#)
- api\_deinit\_in
  - seq64::midi\_alsa, [471](#)
  - seq64::midi\_api, [486](#)
  - seq64::midi\_jack, [532](#)
  - seq64::midibase, [599](#)
  - seq64::midibus, [610](#), [613](#)
  - seq64::rtmidi, [826](#)
- api\_flush
  - seq64::mastermidibase, [452](#)
  - seq64::mastermidibus, [462](#), [464](#)
  - seq64::midi\_alsa, [473](#)
  - seq64::midi\_alsa\_info, [480](#)
  - seq64::midi\_api, [487](#)
  - seq64::midi\_info, [517](#)
  - seq64::midi\_jack, [533](#)
  - seq64::midi\_jack\_info, [543](#)
  - seq64::midibase, [600](#)
  - seq64::midibus, [611](#)
  - seq64::rtmidi, [827](#)
  - seq64::rtmidi\_info, [846](#)
- api\_get\_bus\_name
  - seq64::midi\_api, [488](#)
- api\_get\_midi\_event
  - seq64::mastermidibase, [452](#)
  - seq64::mastermidibus, [460](#), [463](#)
  - seq64::midi\_alsa, [471](#)
  - seq64::midi\_alsa\_info, [478](#)
  - seq64::midi\_api, [486](#)
  - seq64::midi\_in\_jack, [511](#)
  - seq64::midi\_info, [517](#)
  - seq64::midi\_jack, [532](#)
  - seq64::midi\_jack\_info, [542](#)
  - seq64::midibase, [599](#)
  - seq64::midibus, [613](#)
  - seq64::rtmidi, [826](#)
  - seq64::rtmidi\_info, [846](#)
- api\_get\_port\_name
  - seq64::midi\_api, [488](#)
  - seq64::midi\_jack, [535](#)
- api\_init
  - seq64::mastermidibase, [450](#)
  - seq64::mastermidibus, [460](#), [464](#)
- api\_init\_clock
  - seq64::mastermidibase, [451](#)
- api\_init\_in
  - seq64::midi\_alsa, [470](#)
  - seq64::midi\_api, [485](#)
  - seq64::midi\_jack, [531](#)
  - seq64::midibase, [600](#)
  - seq64::midibus, [609](#), [612](#)
  - seq64::rtmidi, [826](#)
- api\_init\_in\_sub
  - seq64::midi\_alsa, [471](#)
  - seq64::midi\_api, [486](#)
  - seq64::midi\_jack, [532](#)
  - seq64::midibase, [599](#)
  - seq64::midibus, [610](#), [613](#)
  - seq64::rtmidi, [826](#)
- api\_init\_out
  - seq64::midi\_alsa, [470](#)
  - seq64::midi\_api, [485](#)
  - seq64::midi\_jack, [530](#)
  - seq64::midibase, [600](#)
  - seq64::midibus, [609](#), [613](#)
  - seq64::rtmidi, [826](#)

- api\_init\_out\_sub
  - seq64::midi\_alsa, 470
  - seq64::midi\_api, 485
  - seq64::midi\_jack, 531
  - seq64::midibase, 599
  - seq64::midibus, 609, 613
  - seq64::rtmidi, 826
- api\_is\_more\_input
  - seq64::mastermidibase, 452
  - seq64::mastermidibus, 460, 463
- api\_play
  - seq64::midi\_alsa, 472
  - seq64::midi\_api, 486
  - seq64::midi\_jack, 533
  - seq64::midibase, 599
  - seq64::midibus, 610, 615
  - seq64::rtmidi, 824
- api\_poll\_for\_midi
  - seq64::mastermidibase, 452
  - seq64::mastermidibus, 460, 463
  - seq64::midi\_alsa, 471
  - seq64::midi\_alsa\_info, 479
  - seq64::midi\_api, 486
  - seq64::midi\_in\_jack, 510
  - seq64::midi\_info, 517
  - seq64::midi\_jack, 532
  - seq64::midi\_jack\_info, 542
  - seq64::midibase, 598
  - seq64::midibus, 614
  - seq64::rtmidi, 827
  - seq64::rtmidi\_info, 846
- api\_port\_start
  - seq64::mastermidibase, 452
  - seq64::mastermidibus, 463, 464
  - seq64::midi\_alsa\_info, 479
  - seq64::midi\_info, 517
  - seq64::midi\_jack\_info, 543
  - seq64::rtmidi\_info, 846
- api\_set\_beats\_per\_minute
  - seq64::mastermidibase, 451
  - seq64::mastermidibus, 462, 464
  - seq64::midi\_alsa, 474
  - seq64::midi\_alsa\_info, 479
  - seq64::midi\_api, 488
  - seq64::midi\_info, 516
  - seq64::midi\_jack, 534
  - seq64::midi\_jack\_info, 543
  - seq64::rtmidi, 825
  - seq64::rtmidi\_info, 846
- api\_set\_ppqn
  - seq64::mastermidibase, 451
  - seq64::mastermidibus, 462, 464
  - seq64::midi\_alsa, 474
  - seq64::midi\_alsa\_info, 479
  - seq64::midi\_api, 488
  - seq64::midi\_info, 516
  - seq64::midi\_jack, 534
  - seq64::midi\_jack\_info, 542
- seq64::rtmidi, 825
- seq64::rtmidi\_info, 846
- api\_start
  - seq64::mastermidibase, 451
  - seq64::mastermidibus, 462
  - seq64::midi\_alsa, 473
  - seq64::midi\_api, 487
  - seq64::midi\_jack, 534
  - seq64::midibase, 601
  - seq64::midibus, 611, 614
  - seq64::rtmidi, 825
- api\_stop
  - seq64::mastermidibase, 451
  - seq64::mastermidibus, 462
  - seq64::midi\_alsa, 473
  - seq64::midi\_api, 487
  - seq64::midi\_jack, 534
  - seq64::midibase, 601
  - seq64::midibus, 612, 614
  - seq64::rtmidi, 825
- api\_sysex
  - seq64::midi\_alsa, 472
  - seq64::midi\_api, 487
  - seq64::midi\_jack, 533
  - seq64::midibase, 600
  - seq64::midibus, 611
  - seq64::rtmidi, 827
- app\_client\_name
  - seq64::rc\_settings, 805
- app\_name
  - seq64::midi\_info, 516
  - seq64::rtmidi\_info, 845
- append
  - seq64::event\_list, 216
- append\_event
  - seq64::sequence, 1012
- append\_sysex
  - seq64::event, 202
- application\_name
  - seq64::rc\_settings, 805
- apply\_length
  - seq64::seqedit, 883
- apply\_song\_transpose
  - seq64::mainwnd, 425
  - seq64::perform, 705
  - seq64::sequence, 1002
- armed\_saved
  - seq64::perform, 709
- array
  - seq64::midi\_message, 555
- at
  - seq64::midi\_message, 554
- at\_bpm\_dn
  - seq64::keys\_perform, 365
- at\_bpm\_up
  - seq64::keys\_perform, 365
- at\_event\_edit
  - seq64::keys\_perform, 368

- at\_fast\_forward
  - seq64::keys\_perform, 368
- at\_follow\_transport
  - seq64::keys\_perform, 368
- at\_group\_learn
  - seq64::keys\_perform, 367
- at\_group\_off
  - seq64::keys\_perform, 367
- at\_group\_on
  - seq64::keys\_perform, 367
- at\_keep\_queue
  - seq64::keys\_perform, 366
- at\_menu\_mode
  - seq64::keys\_perform, 367
- at\_pattern\_edit
  - seq64::keys\_perform, 368
- at\_pause
  - seq64::keys\_perform, 367
- at\_pointer\_position
  - seq64::keys\_perform, 368
- at\_queue
  - seq64::keys\_perform, 366
- at\_replace
  - seq64::keys\_perform, 366
- at\_rewind
  - seq64::keys\_perform, 368
- at\_screenset\_dn
  - seq64::keys\_perform, 366
- at\_screenset\_up
  - seq64::keys\_perform, 366
- at\_set\_playing\_screenset
  - seq64::keys\_perform, 366
- at\_show\_ui\_sequence\_key
  - seq64::keys\_perform, 369
- at\_show\_ui\_sequence\_number
  - seq64::keys\_perform, 369
- at\_snapshot\_1
  - seq64::keys\_perform, 366
- at\_snapshot\_2
  - seq64::keys\_perform, 366
- at\_song\_mode
  - seq64::keys\_perform, 367
- at\_start
  - seq64::keys\_perform, 367
- at\_stop
  - seq64::keys\_perform, 369
- at\_tap\_bpm
  - seq64::keys\_perform, 368
- at\_toggle\_jack
  - seq64::keys\_perform, 367
- at\_toggle\_mutes
  - seq64::keys\_perform, 368
- auto\_option\_save
  - seq64::rc\_settings, 802, 806
- automutex
  - seq64::automutex, 137
- background\_sequence
  - seq64::sequence, 1041, 1042
- baseline\_ppqn
  - seq64::user\_settings, 1107
- beat\_width
  - seq64::midi\_timing, 576
- beats
  - seq64::midi\_measures, 552
- beats\_per\_measure
  - seq64::midi\_timing, 576
- beats\_per\_minute
  - seq64::midi\_timing, 575, 576
- begin
  - seq64::editable\_events, 184
  - seq64::event\_list, 215
- bg\_color
  - seq64::gui\_palette\_gtk2, 305, 306
- black
  - seq64::gui\_palette\_gtk2, 303
- black\_key
  - seq64::gui\_palette\_gtk2, 305
- black\_paint
  - seq64::gui\_palette\_gtk2, 305
- blue
  - seq64::gui\_palette\_gtk2, 305
- bpm
  - seq64::midi\_info, 516
  - seq64::midibase, 590
  - seq64::rtmidi\_info, 846
- bpm\_dn
  - seq64::keys\_perform, 353
- bpm\_from\_tempo\_us
  - seq64, 79
- bpm\_page\_increment
  - seq64::user\_settings, 1106, 1109
- bpm\_precision
  - seq64::user\_settings, 1106, 1108
- bpm\_step\_increment
  - seq64::user\_settings, 1106, 1109
- bpm\_up
  - seq64::keys\_perform, 352
- build\_details
  - seq64, 87
- build\_info\_dialog
  - seq64::mainwnd, 426
- bus
  - seq64::busarray, 141
  - seq64::businfo, 150, 152
  - seq64::user\_settings, 1095
- bus\_container
  - seq64::midi\_info, 520
- bus\_count
  - seq64::user\_settings, 1095
- bus\_instrument
  - seq64::user\_settings, 1095
- bus\_name
  - seq64::midibase, 589, 598
  - seq64::user\_settings, 1095
- busarray
  - seq64::busarray, 140

- seq64::businfo, 153
- businfo
  - seq64::businfo, 149, 150
- BussConstIterator
  - seq64::user\_settings, 1092
- BussIterator
  - seq64::user\_settings, 1092
- bussbyte
  - seq64, 64
- Busses
  - seq64::user\_settings, 1092
- button
  - seq64::click, 158
  - seq64::options, 640
- button\_press
  - seq64::seqroll, 962
- button\_press\_initial
  - seq64::seqroll, 961
- button\_release
  - seq64::seqroll, 962
- c\_backsequence
  - seq64, 117
- c\_bpmtag
  - seq64, 116
- c\_chord\_number
  - seq64, 123
- c\_chord\_size
  - seq64, 123
- c\_chord\_table
  - seq64, 123
- c\_chord\_table\_text
  - seq64, 123
- c\_chord\_text
  - seq64, 123
- c\_controller\_names
  - seq64, 109
- c\_interval\_text
  - seq64, 123
- c\_key\_text
  - seq64, 123
- c\_mainwid\_x
  - seq64, 130
- c\_mainwid\_y
  - seq64, 130
- c\_max\_busses
  - seq64, 124
- c\_max\_instruments
  - seq64, 124
- c\_midi\_control\_16
  - seq64, 120
- c\_midi\_control\_17
  - seq64, 120
- c\_midi\_control\_18
  - seq64, 120
- c\_midi\_control\_19
  - seq64, 120
- c\_midi\_control\_bpm\_dn
  - seq64, 118
- c\_midi\_control\_bpm\_page\_dn
  - seq64, 120
- c\_midi\_control\_bpm\_page\_up
  - seq64, 120
- c\_midi\_control\_bpm\_up
  - seq64, 118
- c\_midi\_control\_mod\_glearn
  - seq64, 119
- c\_midi\_control\_mod\_gmute
  - seq64, 119
- c\_midi\_control\_mod\_queue
  - seq64, 119
- c\_midi\_control\_mod\_replace
  - seq64, 118
- c\_midi\_control\_mod\_snapshot
  - seq64, 118
- c\_midi\_control\_play\_ss
  - seq64, 119
- c\_midi\_control\_playback
  - seq64, 119
- c\_midi\_control\_record
  - seq64, 119
- c\_midi\_control\_solo
  - seq64, 119
- c\_midi\_control\_ss\_dn
  - seq64, 118
- c\_midi\_control\_ss\_up
  - seq64, 118
- c\_midi\_control\_thru
  - seq64, 120
- c\_midi\_controls
  - seq64, 119
- c\_midi\_controls\_extended
  - seq64, 120
- c\_midi\_track\_ctrl
  - seq64, 117
- c\_midibus
  - seq64, 114
- c\_midibus\_input\_size
  - seq64, 114
- c\_midibus\_output\_size
  - seq64, 114
- c\_midibus\_sysex\_chunk
  - seq64, 114
- c\_midich
  - seq64, 115
- c\_midiCLOCKS
  - seq64, 116
- c\_midiCtrl
  - seq64, 116
- c\_music\_scales
  - seq64, 68
- c\_musickey
  - seq64, 117
- c\_musicscale
  - seq64, 117
- c\_mutegroups
  - seq64, 116



- c\_notes
  - seq64, [116](#)
- c\_perf\_bp\_mes
  - seq64, [117](#)
- c\_perf\_bw
  - seq64, [117](#)
- c\_scales\_policy
  - seq64, [121](#)
- c\_scales\_text
  - seq64, [122](#)
- c\_scales\_transpose\_dn
  - seq64, [122](#)
- c\_scales\_transpose\_up
  - seq64, [121](#)
- c\_status\_queue
  - seq64, [129](#)
- c\_status\_replace
  - seq64, [129](#)
- c\_status\_snapshot
  - seq64, [129](#)
- c\_timesig
  - seq64, [116](#)
- c\_transpose
  - seq64, [117](#)
- c\_triggers
  - seq64, [116](#)
- c\_triggers\_new
  - seq64, [116](#)
- calculate\_base\_sizes
  - seq64::mainwid, [409](#)
- cancel\_callback
  - seq64::midi\_api, [490](#)
  - seq64::rtmidi\_in, [833](#)
- category
  - seq64::editable\_event, [173](#)
- category\_string
  - seq64::editable\_event, [173](#)
- category\_t
  - seq64::editable\_event, [169](#)
- cc\_match
  - seq64::event, [200](#)
- change\_event\_data\_lfo
  - seq64::sequence, [1031](#)
- change\_event\_data\_range
  - seq64::sequence, [1030](#)
- change\_focus
  - seq64::eventedit, [232](#)
  - seq64::seqedit, [889](#)
- change\_horz
  - seq64::perfroll, [773](#)
  - seq64::perftime, [788](#)
  - seq64::seqdata, [865](#)
  - seq64::seqevent, [911](#)
  - seq64::seqroll, [960](#)
  - seq64::seqtime, [980](#)
- change\_vert
  - seq64::eventslots, [250](#)
  - seq64::perfnames, [672](#)
  - seq64::perfroll, [773](#)
  - seq64::seqkeys, [925](#)
  - seq64::seqroll, [960](#)
- channel\_count
  - seq64::user\_midi\_bus, [1082](#)
- channel\_match
  - seq64::sequence, [1045](#)
- channel\_max
  - seq64::user\_midi\_bus, [1082](#)
- channel\_string
  - seq64::editable\_event, [176](#)
- char\_height
  - seq64::font, [261](#)
- char\_width
  - seq64::font, [261](#)
- CharList
  - seq64::midi\_list, [548](#)
- CharVector
  - seq64::midi\_vector, [580](#)
- check\_channel
  - seq64::event, [195](#)
- check\_queued\_tick
  - seq64::sequence, [1006](#)
- checklen
  - seq64::midifile, [625](#)
- choose\_file
  - seq64::mainwnd, [428](#)
- choose\_ppqn
  - seq64, [100](#)
- clamp
  - seq64, [105](#), [106](#)
- clamp\_track
  - seq64::perform, [744](#)
- clear
  - seq64::editable\_events, [186](#)
  - seq64::event\_list, [217](#)
  - seq64::midi\_container, [496](#)
  - seq64::midi\_info, [516](#)
  - seq64::midi\_list, [549](#)
  - seq64::midi\_port\_info, [563](#)
  - seq64::midi\_vector, [582](#)
  - seq64::rtmidi\_info, [843](#)
  - seq64::triggers, [1069](#)
- clear\_all
  - seq64::perform, [700](#)
- clear\_flags
  - seq64::seqroll, [963](#)
- clear\_link
  - seq64::event, [203](#)
- clear\_links
  - seq64::event\_list, [219](#)
- clear\_old
  - seq64::seqroll, [963](#)
- clear\_selected
  - seq64::seqroll, [963](#)
- clear\_sequence\_triggers
  - seq64::perform, [703](#)
- clear\_triggers

- seq64::sequence, 1020
- clear\_window
  - seq64::gui\_drawingarea\_gtk2, 285
- click
  - seq64::click, 156
- client
  - seq64::jack\_assistant, 327
- client\_handle
  - seq64::midi\_jack, 527, 528
  - seq64::midi\_jack\_info, 541, 544
- client\_name
  - seq64::jack\_assistant, 327
- client\_open
  - seq64::jack\_assistant, 327
- client\_uuid
  - seq64::jack\_assistant, 327
- clip\_timestamp
  - seq64::sequence, 1028
- clock
  - seq64::busarray, 142
  - seq64::businfo, 153
  - seq64::mastermidibase, 450
  - seq64::midibase, 597
- clock\_callback\_mod
  - seq64::options, 642
- clock\_callback\_off
  - seq64::options, 641
- clock\_callback\_on
  - seq64::options, 642
- clock\_e
  - seq64, 68
- clock\_mod\_callback
  - seq64::options, 642
- clock\_tick\_duration\_bogus
  - seq64, 81
- clock\_ticks\_from\_ppqn
  - seq64, 82
- clocks\_per\_metronome
  - seq64::perform, 694
  - seq64::sequence, 1000
- close\_client
  - seq64::midi\_jack, 529
- close\_out
  - seq64::eventedit, 232
- close\_port
  - seq64::midi\_jack, 529
- collapse
  - seq64::perfedit, 659
  - seq64::perform, 728
- Color
  - seq64::font, 259
  - seq64::gui\_palette\_gtk2, 301
- combine\_bytes
  - seq64::perform, 697
- complete\_paste
  - seq64::seqroll, 955
- condition\_var
  - seq64::condition\_var, 160
- config\_directory
  - seq64::rc\_settings, 805, 810
- config\_filename
  - seq64::rc\_settings, 805, 810
- config\_filename\_alt
  - seq64::rc\_settings, 805, 811
- config\_filespec
  - seq64::rc\_settings, 801
- configfile
  - seq64::configfile, 163
- connect
  - seq64::midi\_jack\_info, 545
- connect\_name
  - seq64::midi\_info, 519
  - seq64::midi\_port\_info, 564
  - seq64::midibase, 589
- connect\_port
  - seq64::midi\_jack, 529
- const\_iterator
  - seq64::editable\_events, 181
  - seq64::event\_list, 214
  - seq64::seqmenu, 934
- container
  - seq64::midi\_message, 554
- continue\_from
  - seq64::busarray, 142
  - seq64::businfo, 153
  - seq64::mastermidibase, 442
  - seq64::midibase, 597
- continue\_sysex
  - seq64::rtmidi\_in\_data, 836
- control\_height
  - seq64::user\_settings, 1100, 1105
- controller\_active
  - seq64::user\_instrument, 1077
  - seq64::user\_settings, 1096
- controller\_count
  - seq64::user\_instrument, 1076
- controller\_max
  - seq64::user\_instrument, 1076
- controller\_name
  - seq64::user\_instrument, 1076
  - seq64::user\_settings, 1097
- controllers
  - seq64::user\_instrument\_t, 1079
- controllers\_active
  - seq64::user\_instrument\_t, 1079
- convert\_drop\_xy
  - seq64::perfrroll, 774
- convert\_sel\_box\_to\_rect
  - seq64::seqroll, 958
- convert\_t
  - seq64::seqevent, 912
- convert\_tn
  - seq64::seqroll, 957
- convert\_tn\_box\_to\_rect
  - seq64::seqroll, 958
- convert\_x

- seq64::perfroll, 772
- seq64::seqdata, 865
- seq64::seqevent, 911
- convert\_xy
  - seq64::perfroll, 772
  - seq64::seqroll, 957
- convert\_y
  - seq64::eventslots, 249
  - seq64::perfnames, 671
  - seq64::seqkeys, 924
- copy
  - seq64::perfedit, 659
  - seq64::perform, 728
  - seq64::triggers, 1068
- copy\_definitions
  - seq64::user\_instrument, 1078
  - seq64::user\_midi\_bus, 1083
- copy\_events
  - seq64::sequence, 1042
- copy\_selected
  - seq64::sequence, 1025
  - seq64::triggers, 1066
- copy\_selected\_trigger
  - seq64::sequence, 1017
- copy\_triggers
  - seq64::perform, 723
  - seq64::sequence, 1019
- count
  - seq64::busarray, 141
  - seq64::editable\_events, 185
  - seq64::event\_list, 215
  - seq64::midi\_message, 555
  - seq64::midi\_queue, 566
  - seq64::midi\_splitter, 572
- count\_selected\_events
  - seq64::event\_list, 221
- count\_selected\_notes
  - seq64::event\_list, 220
- create\_jack\_client
  - seq64, 94
- create\_lash\_driver
  - seq64, 97
- create\_master\_bus
  - seq64::perform, 745
- create\_menu\_image
  - seq64::seqedit, 888
- create\_menus
  - seq64::seqedit, 887
- create\_ringbuffer
  - seq64::midi\_jack, 529
- create\_seqedit
  - seq64::seqmenu, 938
- current\_event
  - seq64::editable\_events, 186
- current\_index
  - seq64::eventslots, 244
- current\_screen\_set\_notepad
  - seq64::perform, 731
- current\_seq
  - seq64::seqmenu, 935
- current\_x
  - seq64::gui\_drawingarea\_gtk2, 284
- current\_y
  - seq64::gui\_drawingarea\_gtk2, 284
- cut\_selected
  - seq64::sequence, 1025
- cut\_selected\_trigger
  - seq64::sequence, 1017
- dark\_blue
  - seq64::gui\_palette\_gtk2, 303
- dark\_cyan
  - seq64::gui\_palette\_gtk2, 304
- dark\_green
  - seq64::gui\_palette\_gtk2, 303
- dark\_grey
  - seq64::gui\_palette\_gtk2, 304
- dark\_magenta
  - seq64::gui\_palette\_gtk2, 304
- dark\_orange
  - seq64::gui\_palette\_gtk2, 303
- dark\_red
  - seq64::gui\_palette\_gtk2, 303
- data
  - seq64::event, 205
  - seq64::midi\_control, 503
- data\_string
  - seq64::editable\_event, 176
- deactivate
  - seq64::businfo, 152
- deallocate
  - seq64::midi\_queue, 567
- decrement\_beats\_per\_minute
  - seq64::perform, 717
- decrement\_bottom
  - seq64::eventslots, 252
- decrement\_current
  - seq64::eventslots, 252
- decrement\_data1
  - seq64::event, 201
- decrement\_data2
  - seq64::event, 201
- decrement\_offset
  - seq64::trigger, 1056
- decrement\_screenset
  - seq64::perform, 718
- decrement\_selected
  - seq64::sequence, 1032
- decrement\_tick\_end
  - seq64::trigger, 1056
- decrement\_tick\_start
  - seq64::trigger, 1055
- decrement\_top
  - seq64::eventslots, 251
- deinit
  - seq64::jack\_assistant, 321
- deinit\_in

- seq64::midibase, 595
- deinit\_jack\_transport
  - seq64::perform, 740
- del\_selected\_trigger
  - seq64::sequence, 1017
- del\_trigger
  - seq64::sequence, 1014
- delete\_api
  - seq64::rtmidi, 829
  - seq64::rtmidi\_info, 848
- delete\_current\_event
  - seq64::eventslots, 247
- delete\_current\_sequence
  - seq64::seqmenu, 937
- delete\_lash\_driver
  - seq64, 98
- delete\_sequence
  - seq64::perform, 702
- delta\_time\_us\_to\_ticks
  - seq64, 80
- device\_ignore
  - seq64::rc\_settings, 804, 808
- device\_ignore\_num
  - seq64::rc\_settings, 804, 808
- disconnect
  - seq64::midi\_jack\_info, 545
- display\_name
  - seq64::midibase, 589, 598
- divisions
  - seq64::midi\_measures, 552
- do\_action
  - seq64::seqedit, 888
- do\_input
  - seq64::rtmidi\_in\_data, 836
- done
  - seq64::midi\_container, 495
  - seq64::midi\_list, 549
  - seq64::midi\_vector, 581
- double\_ticks\_from\_ppqn
  - seq64, 82
- draw\_all
  - seq64::perfroll, 771
- draw\_area
  - seq64::seqkeys, 924
- draw\_background
  - seq64::perftime, 788
  - seq64::seqevent, 909
- draw\_background\_on
  - seq64::perfroll, 773
- draw\_background\_on\_pixmap
  - seq64::seqroll, 953
- draw\_drawable
  - seq64::gui\_drawingarea\_gtk2, 294
- draw\_drawable\_row
  - seq64::perfroll, 773
- draw\_event
  - seq64::eventslots, 249
- draw\_events
  - seq64::eventslots, 250
- draw\_events\_on
  - seq64::seqdata, 865
  - seq64::seqevent, 911
  - seq64::seqroll, 959
- draw\_events\_on\_pixmap
  - seq64::seqdata, 866
  - seq64::seqevent, 909
  - seq64::seqroll, 953
- draw\_key
  - seq64::seqkeys, 924
- draw\_line
  - seq64::gui\_drawingarea\_gtk2, 286–288
- draw\_line\_on\_pixmap
  - seq64::gui\_drawingarea\_gtk2, 286, 287
- draw\_line\_on\_window
  - seq64::seqdata, 863
- draw\_marker\_on\_sequence
  - seq64::mainwid, 406
- draw\_normal\_rectangle\_on\_pixmap
  - seq64::gui\_drawingarea\_gtk2, 293
- draw\_pixmap\_on\_window
  - seq64::mainwid, 405
  - seq64::perftime, 790
  - seq64::seqdata, 866
  - seq64::seqevent, 909
  - seq64::seqtime, 979
- draw\_progress
  - seq64::perfroll, 771
- draw\_progress\_on\_window
  - seq64::perftime, 788
  - seq64::seqroll, 954
  - seq64::seqtime, 979
- draw\_rectangle
  - seq64::gui\_drawingarea\_gtk2, 290–292
- draw\_rectangle\_on\_pixmap
  - seq64::gui\_drawingarea\_gtk2, 292, 293
- draw\_selection\_on\_window
  - seq64::seqevent, 909
  - seq64::seqroll, 954
- draw\_sequence
  - seq64::perfnames, 671
- draw\_sequence\_on
  - seq64::perfroll, 773
- draw\_sequence\_on\_pixmap
  - seq64::mainwid, 407
- draw\_sequence\_pixmap\_on\_window
  - seq64::mainwid, 408
- draw\_sequences
  - seq64::perfedit, 660
  - seq64::perfnames, 671
- draw\_sequences\_on\_pixmap
  - seq64::mainwid, 408
- draw\_type\_t
  - seq64, 69
- dref
  - seq64::editable\_events, 184, 185
  - seq64::event\_list, 218

- drop\_action
  - seq64::seqroll, [966](#)
- drop\_event
  - seq64::seqevent, [910](#)
- drop\_x
  - seq64::gui\_drawingarea\_gtk2, [285](#)
- drop\_y
  - seq64::gui\_drawingarea\_gtk2, [285](#)
- dump\_midi\_input
  - seq64::mastermidibase, [444](#)
- dump\_setting\_summary
  - seq64::userfile, [1121](#)
- dump\_summary
  - seq64::user\_settings, [1105](#)
- EVENT\_AFTERTOUCHE
  - seq64, [110](#)
- EVENT\_ANY
  - seq64, [109](#)
- EVENT\_CHANNEL\_PRESSURE
  - seq64, [110](#)
- EVENT\_CLEAR\_CHAN\_MASK
  - seq64, [113](#)
- EVENT\_CONTROL\_CHANGE
  - seq64, [110](#)
- EVENT\_GET\_CHAN\_MASK
  - seq64, [113](#)
- EVENT\_MIDI\_ACTIVE\_SENS
  - seq64, [113](#)
- EVENT\_MIDI\_CLOCK
  - seq64, [112](#)
- EVENT\_MIDI\_CONTINUE
  - seq64, [112](#)
- EVENT\_MIDI\_META
  - seq64, [113](#)
- EVENT\_MIDI\_QUARTER\_FRAME
  - seq64, [111](#)
- EVENT\_MIDI\_RESET
  - seq64, [113](#)
- EVENT\_MIDI\_SONG\_F4
  - seq64, [111](#)
- EVENT\_MIDI\_SONG\_F5
  - seq64, [111](#)
- EVENT\_MIDI\_SONG\_F9
  - seq64, [112](#)
- EVENT\_MIDI\_SONG\_FD
  - seq64, [113](#)
- EVENT\_MIDI\_SONG\_POS
  - seq64, [111](#)
- EVENT\_MIDI\_SONG\_SELECT
  - seq64, [111](#)
- EVENT\_MIDI\_START
  - seq64, [112](#)
- EVENT\_MIDI\_STOP
  - seq64, [112](#)
- EVENT\_MIDI\_SYSEX\_CONTINUE
  - seq64, [112](#)
- EVENT\_MIDI\_SYSEX\_END
  - seq64, [112](#)
- EVENT\_MIDI\_SYSEX
  - seq64, [111](#)
- EVENT\_MIDI\_TUNE\_SELECT
  - seq64, [112](#)
- EVENT\_NOTE\_OFF
  - seq64, [110](#)
- EVENT\_NOTE\_ON
  - seq64, [110](#)
- EVENT\_NULL\_CHANNEL
  - seq64, [113](#)
- EVENT\_PITCH\_WHEEL
  - seq64, [110](#)
- EVENT\_PROGRAM\_CHANGE
  - seq64, [110](#)
- EVENT\_STATUS\_BIT
  - seq64, [109](#)
- EVENTS\_ALL
  - seq64, [114](#)
- EVENTS\_UNSELECTED
  - seq64, [114](#)
- edit\_action\_t
  - seq64, [69](#)
- edit\_callback\_notepad
  - seq64::mainwnd, [422](#)
- edit\_field\_has\_focus
  - seq64::mainwnd, [429](#)
- editable\_event
  - seq64::editable\_event, [170](#), [171](#)
- editable\_events
  - seq64::editable\_events, [182](#)
  - seq64::event\_list, [222](#)
- empty
  - seq64::event\_list, [215](#)
  - seq64::midi\_message, [555](#)
  - seq64::midi\_queue, [566](#)
- end
  - seq64::editable\_events, [184](#)
  - seq64::event\_list, [215](#)
- enqueue\_draw
  - seq64::eventedit, [229](#)
  - seq64::eventslots, [249](#)
  - seq64::perfedit, [655](#)
  - seq64::perfnames, [671](#)
  - seq64::perfroll, [774](#)
  - seq64::perftime, [788](#)
- enregister
  - seq64::perform, [696](#)
- enregister\_peer
  - seq64::perfedit, [656](#)
- enregister\_perfedits
  - seq64::mainwnd, [424](#)
- errdump
  - seq64::midifile, [633](#)
- error
  - seq64::midi\_api, [489](#)
  - seq64::midi\_info, [519](#)
- error\_is\_fatal
  - seq64::midifile, [622](#)

- error\_message
  - seq64, 88
  - seq64::jack\_assistant, 330
  - seq64::midifile, 622
  - seq64::optionsfile, 649
- event
  - seq64::event, 192
- event\_count
  - seq64::eventslots, 244
  - seq64::sequence, 996
- event\_edit
  - seq64::keys\_perform, 358
- event\_in\_range
  - seq64::sequence, 1042
- event\_key
  - seq64::event\_list::event\_key, 210
- event\_list
  - seq64::event\_list, 214
- event\_name
  - seq64::editable\_event::name\_value\_t, 639
- event\_value
  - seq64::editable\_event::name\_value\_t, 639
- EventStack
  - seq64::sequence, 993
- eventedit
  - seq64::eventedit, 227
  - seq64::eventslots, 255
- Events
  - seq64::editable\_events, 181
  - seq64::event\_list, 213
- events
  - seq64::editable\_events, 184
  - seq64::event\_list, 221
  - seq64::sequence, 995
- EventsPair
  - seq64::editable\_events, 181
  - seq64::event\_list, 213
- eventslots
  - seq64::editable\_events, 187
  - seq64::eventedit, 235
  - seq64::eventslots, 243
- expand
  - seq64::perfedit, 659
  - seq64::perform, 728
- extract\_bus\_name
  - seq64, 84
- extract\_names
  - seq64::midi\_jack\_info, 545
- extract\_port\_name
  - seq64, 85
- extract\_port\_names
  - seq64, 84
- extract\_timing\_numbers
  - seq64, 71
- FF\_RW\_timeout
  - seq64, 105
  - seq64::perform, 698
- FF\_rewind
  - seq64::perform, 698
- fast\_forward
  - seq64::keys\_perform, 360
  - seq64::perfedit, 657
  - seq64::perform, 700
- ff\_rw\_button\_t
  - seq64::perform, 691
- ff\_rw\_type
  - seq64::perform, 699
- fg\_color
  - seq64::gui\_palette\_gtk2, 306
- file\_access
  - seq64, 89
- file\_accessible
  - seq64, 90
- file\_executable
  - seq64, 91
- file\_exists
  - seq64, 89
- file\_exit
  - seq64::mainwnd, 428
- file\_import\_dialog
  - seq64::mainwnd, 426
- file\_is\_directory
  - seq64, 91
- file\_new
  - seq64::mainwnd, 425
- file\_open
  - seq64::mainwnd, 425
- file\_readable
  - seq64, 90
- file\_save
  - seq64::mainwnd, 425
- file\_save\_as
  - seq64::mainwnd, 427
- file\_writable
  - seq64, 90
- filename
  - seq64::rc\_settings, 804, 809
- fill
  - seq64::midi\_container, 494
- fill\_background\_pixmap
  - seq64::perfroll, 770
- fill\_background\_window
  - seq64::mainwid, 405
- fill\_meta\_track\_end
  - seq64::midi\_container, 498
- fill\_proprietary
  - seq64::midi\_container, 498
- fill\_seq\_name
  - seq64::midi\_container, 498
- fill\_seq\_number
  - seq64::midi\_container, 497
- fill\_time\_sig\_and\_tempo
  - seq64::midi\_container, 498
- fill\_top\_bar
  - seq64::seqedit, 886
- filter\_by\_channel

- seq64::mastermidibase, 440
- seq64::perform, 695
- seq64::rc\_settings, 803, 808
- filter\_callback
  - seq64::options, 642
- finish
  - seq64::perform, 703
- first\_message
  - seq64::rtmidi\_in\_data, 836
- flush
  - seq64::mastermidibase, 444
  - seq64::midibase, 596
- follow\_progress
  - seq64::perfroll, 771
  - seq64::seqroll, 955
- follow\_transport
  - seq64::keys\_perform, 360
- font
  - seq64::font, 259
- font\_render
  - seq64, 102
- force\_draw
  - seq64::gui\_drawingarea\_gtk2, 285
  - seq64::seqevent, 910
  - seq64::seqkeys, 923
  - seq64::seqroll, 955
- format\_timestamp
  - seq64::editable\_event, 175
- front
  - seq64::midi\_queue, 567
- FruityPerfInput
  - seq64::FruityPerfInput, 265
  - seq64::perfroll, 778
  - seq64::triggers, 1072
- FruitySeqEventInput
  - seq64::FruitySeqEventInput, 269
  - seq64::seqevent, 915
- FruitySeqRollInput
  - seq64::FruitySeqRollInput, 272
  - seq64::seqkeys, 928
  - seq64::seqroll, 971
- full
  - seq64::midi\_queue, 566
- full\_port\_count
  - seq64::midi\_info, 516
  - seq64::rtmidi, 828
  - seq64::rtmidi\_info, 844
- g\_midi\_control\_limit
  - seq64, 121
- g\_rc\_settings
  - seq64, 130
- g\_user\_settings
  - seq64, 130
- get
  - seq64::midi\_container, 496
  - seq64::midi\_list, 549
  - seq64::midi\_vector, 581
- get\_32nds\_per\_quarter
  - seq64::perform, 694
  - seq64::sequence, 1000
- get\_all\_port\_info
  - seq64::midi\_alsa\_info, 480
  - seq64::midi\_info, 520
  - seq64::midi\_jack\_info, 544
  - seq64::rtmidi\_info, 845
- get\_api
  - seq64::rtmidi, 828, 829
- get\_api\_info
  - seq64::rtmidi\_info, 847
- get\_beat\_width
  - seq64::jack\_assistant, 319
  - seq64::perform, 694
  - seq64::sequence, 1000
- get\_beats\_per\_bar
  - seq64::perform, 693
  - seq64::sequence, 999
- get\_beats\_per\_measure
  - seq64::jack\_assistant, 320
- get\_beats\_per\_minute
  - seq64::jack\_assistant, 320
  - seq64::mastermidibase, 440
  - seq64::perform, 706
- get\_bpm
  - seq64::mastermidibase, 440
- get\_bus\_id
  - seq64::midi\_info, 518
  - seq64::midi\_port\_info, 563
  - seq64::midibase, 589
  - seq64::rtmidi, 827
  - seq64::rtmidi\_info, 843
- get\_bus\_index
  - seq64::midibase, 589
- get\_bus\_name
  - seq64::midi\_info, 518
  - seq64::midi\_port\_info, 563
  - seq64::rtmidi, 827
  - seq64::rtmidi\_info, 844
- get\_channel
  - seq64::event, 195
- get\_client
  - seq64::midi\_alsa, 469
  - seq64::midibus, 609
- get\_clipboard\_box
  - seq64::sequence, 1027
- get\_clock
  - seq64::busarray, 144
  - seq64::mastermidibase, 448
  - seq64::midibase, 591
- get\_clock\_mod
  - seq64::midibase, 594
- get\_compiled\_api
  - seq64::rtmidi\_info, 842
- get\_current\_jack\_position
  - seq64, 96
  - seq64::jack\_assistant, 334
  - seq64::perform, 751



- get\_current\_sequence
  - seq64::seqmenu, 936
- get\_data
  - seq64::event, 201
- get\_editing
  - seq64::sequence, 1002
- get\_error\_message
  - seq64::configfile, 164
- get\_follow\_transport
  - seq64::jack\_assistant, 326
  - seq64::perform, 699
- get\_group\_mute\_state
  - seq64::perform, 711
- get\_hold\_undo
  - seq64::sequence, 997
- get\_input
  - seq64::busarray, 146
  - seq64::mastermidibase, 447
  - seq64::midi\_info, 518
  - seq64::midi\_port\_info, 563
  - seq64::midibase, 592
  - seq64::perform, 747
  - seq64::rtmidi\_info, 844
- get\_jack\_client\_info
  - seq64::jack\_assistant, 328
- get\_jack\_mode
  - seq64::jack\_assistant, 325
- get\_jack\_pos
  - seq64::jack\_assistant, 325
- get\_jack\_stop\_tick
  - seq64::jack\_assistant, 325
- get\_jack\_tick
  - seq64::jack\_assistant, 325
  - seq64::perform, 703
- get\_key\_events
  - seq64::keys\_perform, 362
  - seq64::perform, 713
- get\_key\_events\_rev
  - seq64::keys\_perform, 363
  - seq64::perform, 713
- get\_key\_groups
  - seq64::keys\_perform, 362
  - seq64::perform, 713
- get\_key\_groups\_rev
  - seq64::keys\_perform, 363
  - seq64::perform, 713
- get\_keys
  - seq64::keys\_perform, 352
- get\_last\_tick
  - seq64::sequence, 1004
- get\_left\_tick
  - seq64::perform, 704
- get\_length
  - seq64::sequence, 1004
- get\_linked
  - seq64::event, 203
- get\_max\_trigger
  - seq64::perform, 728
- seq64::sequence, 1019
- get\_maximum
  - seq64::triggers, 1068
- get\_measures
  - seq64::seqedit, 883
  - seq64::sequence, 999
- get\_message
  - seq64::rterror, 821
- get\_midi\_bus
  - seq64::sequence, 1020
- get\_midi\_bus\_name
  - seq64::busarray, 144
- get\_midi\_channel
  - seq64::sequence, 1008
- get\_midi\_event
  - seq64::busarray, 147
  - seq64::mastermidibase, 446
  - seq64::midibase, 594
- get\_midi\_in\_bus\_name
  - seq64::mastermidibase, 445
- get\_midi\_out\_bus\_name
  - seq64::mastermidibase, 445
- get\_minmax\_note\_events
  - seq64::sequence, 1038
- get\_name
  - seq64::sequence, 1002
- get\_next\_event
  - seq64::sequence, 1038, 1039
- get\_next\_note\_event
  - seq64::sequence, 1036
- get\_next\_trigger
  - seq64::sequence, 1039
- get\_note
  - seq64::event, 205
- get\_note\_velocity
  - seq64::event, 206
- get\_num\_in\_buses
  - seq64::mastermidibase, 440
- get\_num\_out\_buses
  - seq64::mastermidibase, 440
- get\_num\_selected\_events
  - seq64::sequence, 1024
- get\_num\_selected\_notes
  - seq64::sequence, 1024
- get\_offset
  - seq64::perform, 712
- get\_playing
  - seq64::sequence, 1005
- get\_playing\_screensets
  - seq64::perform, 726
- get\_port
  - seq64::midi\_alsa, 470
  - seq64::midibus, 609
- get\_port\_count
  - seq64::midi\_info, 517
  - seq64::midi\_port\_info, 563
  - seq64::rtmidi, 828
  - seq64::rtmidi\_info, 844



- get\_port\_id
  - seq64::midi\_info, 518
  - seq64::midi\_port\_info, 563
  - seq64::midibase, 590
  - seq64::rtmidi, 828
  - seq64::rtmidi\_info, 844
- get\_port\_name
  - seq64::midi\_info, 518
  - seq64::midi\_port\_info, 563
  - seq64::rtmidi, 828
  - seq64::rtmidi\_info, 844
- get\_ppqn
  - seq64::jack\_assistant, 319
  - seq64::mastermidibase, 440
  - seq64::sequence, 999
- get\_quantized\_rec
  - seq64::sequence, 1006
- get\_queue\_number
  - seq64::midi\_port\_info, 564
- get\_queued
  - seq64::sequence, 1005
- get\_queued\_tick
  - seq64::sequence, 1005
- get\_raise
  - seq64::sequence, 1003
- get\_rank
  - seq64::event, 207
- get\_recording
  - seq64::sequence, 1006
- get\_right\_tick
  - seq64::perform, 705
- get\_screen\_set\_notepad
  - seq64::perform, 731
- get\_screenshot
  - seq64::perform, 726
- get\_selected\_box
  - seq64::seqroll, 959
  - seq64::sequence, 1026
- get\_selected\_end
  - seq64::triggers, 1068
- get\_selected\_start
  - seq64::triggers, 1067
- get\_sequence
  - seq64::mastermidibase, 441
  - seq64::perform, 719, 720
  - seq64::seqmenu, 936
- get\_song\_mute
  - seq64::sequence, 1001
- get\_start\_tick
  - seq64::perform, 704
- get\_state
  - seq64::triggers, 1065
- get\_status
  - seq64::event, 200
- get\_sysex
  - seq64::event, 202
- get\_sysex\_size
  - seq64::event, 203
- get\_system
  - seq64::midi\_info, 518
  - seq64::midi\_port\_info, 564
  - seq64::rtmidi\_info, 845
- get\_thru
  - seq64::sequence, 1007
- get\_tick
  - seq64::perform, 703
- get\_timestamp
  - seq64::event, 195
- get\_toggle\_jack
  - seq64::perfedit, 656
  - seq64::perform, 697
- get\_transposable
  - seq64::sequence, 1002
- get\_transpose
  - seq64::perform, 706
- get\_trigger\_count
  - seq64::sequence, 995
- get\_trigger\_offset
  - seq64::sequence, 1020
- get\_trigger\_paste\_tick
  - seq64::sequence, 995
  - seq64::triggers, 1071
- get\_trigger\_state
  - seq64::sequence, 1014
- get\_triggers
  - seq64::sequence, 1015
- get\_version
  - seq64::rtmidi\_info, 842
- get\_virtual
  - seq64::midi\_info, 518
  - seq64::midi\_port\_info, 564
  - seq64::rtmidi\_info, 845
- getType
  - seq64::rterror, 821
- global\_queue
  - seq64::midi\_info, 519, 520
  - seq64::rtmidi\_info, 845
- global\_seq\_feature
  - seq64::user\_settings, 1100
- gmute\_tracks
  - seq64::user\_settings, 1098
- green
  - seq64::gui\_palette\_gtk2, 305
- grey
  - seq64::gui\_palette\_gtk2, 304
- grid\_brackets
  - seq64::user\_settings, 1097, 1103
- grid\_is\_black
  - seq64::user\_settings, 1097
- grid\_is\_normal
  - seq64::user\_settings, 1097
- grid\_is\_white
  - seq64::user\_settings, 1097
- grid\_style
  - seq64::user\_settings, 1097, 1103
- group\_learn

- seq64::keys\_perform, 357
- group\_off
  - seq64::keys\_perform, 356, 357
- group\_on
  - seq64::keys\_perform, 356
- grow
  - seq64::perfedit, 659
  - seq64::triggers, 1064
- grow\_edit\_t
  - seq64::triggers, 1060
- grow\_selected
  - seq64::sequence, 1033
- grow\_selected\_notes
  - seq64::seqroll, 961
- grow\_trigger
  - seq64::sequence, 1014
- growing
  - seq64::seqroll, 966
- gs\_mainwid\_pointer
  - seq64, 130
- gs\_perfedit\_pointer\_0
  - seq64, 131
- gs\_perfedit\_pointer\_1
  - seq64, 131
- Gtk, 49
- gtk\_drawarea\_init
  - seq64::gui\_drawingarea\_gtk2, 295
- gui
  - seq64::perform, 695
- gui\_assistant
  - seq64::gui\_assistant, 276
- gui\_assistant\_gtk2
  - seq64::gui\_assistant\_gtk2, 278
- gui\_drawingarea\_gtk2
  - seq64::gui\_drawingarea\_gtk2, 283
- gui\_palette\_gtk2
  - seq64::gui\_palette\_gtk2, 302
- gui\_window\_gtk2
  - seq64::gui\_window\_gtk2, 311
- handle\_cancel
  - seq64::eventedit, 233
- handle\_close
  - seq64::eventedit, 232
  - seq64::seqedit, 889
- handle\_config
  - seq64::lash, 388
- handle\_delete
  - seq64::eventedit, 232
- handle\_event
  - seq64::lash, 388
- handle\_insert
  - seq64::eventedit, 232
- handle\_midi\_control
  - seq64::perform, 730
- handle\_midi\_control\_ex
  - seq64::perform, 731
- handle\_modify
  - seq64::eventedit, 233
- handle\_motion\_key
  - seq64::AbstractPerfInput, 135
  - seq64::FruityPerfInput, 268
  - seq64::Seq24PerfInput, 855
- handle\_save
  - seq64::eventedit, 233
- handle\_signal
  - seq64::mainwnd, 422
- have\_redo
  - seq64::perform, 727
  - seq64::sequence, 997
- have\_undo
  - seq64::perform, 727
  - seq64::sequence, 997
- height
  - seq64::gui\_drawingarea\_gtk2::rect, 818
  - seq64::rect, 818
- help\_check
  - seq64, 85
  - seq64::rc\_settings, 812
- highlight
  - seq64::perform, 719
- home\_config\_directory
  - seq64::rc\_settings, 811
- horizontal\_adjust
  - seq64::perffroll, 774
  - seq64::seqedit, 882
  - seq64::seqroll, 955
- horizontal\_set
  - seq64::perffroll, 775
  - seq64::seqedit, 882
- idle\_progress
  - seq64::maintime, 397
  - seq64::perftime, 790
  - seq64::seqroll, 959
  - seq64::seqtime, 981
- idle\_redraw
  - seq64::seqdata, 863
  - seq64::seqevent, 910
  - seq64::seqroll, 959
- ignore\_flags
  - seq64::rtmidi\_in\_data, 835, 836
- in\_range
  - seq64::midi\_control, 505
- increment
  - seq64::midi\_splitter, 571
- increment\_beats\_per\_minute
  - seq64::perform, 718
- increment\_bottom
  - seq64::eventslots, 253
- increment\_current
  - seq64::eventslots, 252
- increment\_data1
  - seq64::event, 201
- increment\_data2
  - seq64::event, 201
- increment\_offset
  - seq64::trigger, 1056

- increment\_screenset
  - seq64::perform, [719](#)
- increment\_selected
  - seq64::sequence, [1032](#)
- increment\_size
  - seq64::perftroll, [770](#)
  - seq64::perftime, [788](#)
- increment\_tick\_end
  - seq64::trigger, [1055](#)
- increment\_tick\_start
  - seq64::trigger, [1055](#)
- increment\_top
  - seq64::eventslots, [252](#)
- info\_message
  - seq64, [88](#)
  - seq64::jack\_assistant, [330](#)
- init
  - seq64::font, [260](#)
  - seq64::jack\_assistant, [321](#)
  - seq64::lash, [388](#)
  - seq64::mastermidibase, [439](#)
- init\_before\_show
  - seq64::perfedit, [655](#)
  - seq64::perftroll, [770](#)
- init\_clock
  - seq64::busarray, [142](#)
  - seq64::businfo, [151–153](#)
  - seq64::mastermidibase, [443](#)
  - seq64::midibase, [597](#)
- init\_in
  - seq64::midibase, [595](#)
- init\_in\_sub
  - seq64::midibase, [595](#)
- init\_input
  - seq64::businfo, [151](#), [152](#)
- init\_jack\_transport
  - seq64::perform, [739](#)
- init\_out
  - seq64::midibase, [595](#)
- init\_out\_sub
  - seq64::midibase, [595](#)
- initialize
  - seq64::busarray, [141](#)
  - seq64::businfo, [151](#)
  - seq64::midi\_splitter, [571](#)
- initialize\_buses
  - seq64::mastermidibase, [445](#)
- initialized
  - seq64::businfo, [151](#)
- inner\_start
  - seq64::perform, [744](#)
- inner\_stop
  - seq64::perform, [744](#)
- input
  - seq64::mastermidibase, [450](#)
- input\_callback
  - seq64::options, [642](#)
- input\_data
  - seq64::midi\_api, [490](#)
- input\_func
  - seq64::perform, [710](#)
- input\_ports
  - seq64::midi\_info, [515](#)
- input\_thread\_func
  - seq64, [99](#)
  - seq64::perform, [748](#)
- insert\_event
  - seq64::eventslots, [245](#), [246](#)
- install\_sequence
  - seq64::perform, [743](#)
- install\_signal\_handlers
  - seq64::mainwnd, [428](#)
- instrument
  - seq64::user\_instrument\_t, [1079](#)
  - seq64::user\_midi\_bus, [1082](#)
  - seq64::user\_midi\_bus\_t, [1084](#)
  - seq64::user\_settings, [1095](#)
- instrument\_controller\_active
  - seq64::user\_settings, [1096](#)
- instrument\_controller\_name
  - seq64::user\_settings, [1096](#)
- instrument\_count
  - seq64::user\_settings, [1095](#)
- instrument\_name
  - seq64::user\_settings, [1096](#)
- InstrumentConstIterator
  - seq64::user\_settings, [1093](#)
- InstrumentIterator
  - seq64::user\_settings, [1093](#)
- Instruments
  - seq64::user\_settings, [1093](#)
- interaction\_method
  - seq64::rc\_settings, [804](#), [809](#)
- interaction\_method\_t
  - seq64, [68](#)
- intersect
  - seq64::triggers, [1066](#)
- intersect\_events
  - seq64::sequence, [1017](#)
- intersect\_notes
  - seq64::sequence, [1016](#)
- intersect\_triggers
  - seq64::sequence, [1015](#)
- invalid\_key
  - seq64, [97](#)
- inverse\_active
  - seq64::midi\_control, [503](#)
- inverse\_colors
  - seq64::user\_settings, [1102](#), [1108](#)
- is
  - seq64::keystroke, [384](#)
- is\_active
  - seq64::perform, [705](#)
- is\_adding
  - seq64::AbstractPerfInput, [135](#)
- is\_adding\_pressed

- seq64::AbstractPerfInput, 136
- is\_black\_key
  - seq64::seqkeys, 925
- is\_channel\_msg
  - seq64::event, 195
- is\_control\_status
  - seq64::perform, 692
- is\_ctrl\_key
  - seq64, 102–104
- is\_ctrl\_shift\_key
  - seq64, 104
- is\_current\_seq\_active
  - seq64::seqmenu, 936
- is\_current\_seq\_in\_edit
  - seq64::seqmenu, 936
- is\_delete
  - seq64::keystroke, 384
- is\_desired\_cc\_or\_not\_cc
  - seq64::event, 197
- is\_dirty\_edit
  - seq64::perform, 724
  - seq64::sequence, 1007
- is\_dirty\_main
  - seq64::perform, 724
  - seq64::sequence, 1007
- is\_dirty\_names
  - seq64::perform, 725
  - seq64::sequence, 1007
- is\_dirty\_perf
  - seq64::perform, 725
  - seq64::sequence, 1007
- is\_dumping
  - seq64::mastermidibase, 440
- is\_edit\_sequence
  - seq64::perform, 693
  - seq64::seqmenu, 936
- is\_exportable
  - seq64::perform, 725
- is\_group\_learning
  - seq64::perform, 734
- is\_input\_port
  - seq64::midi\_api, 485
  - seq64::midibase, 590, 591
- is\_input\_system\_port
  - seq64::mastermidibase, 448
  - seq64::perform, 747
- is\_inverse
  - seq64::gui\_palette\_gtk2, 303
- is\_jack\_master
  - seq64::perform, 696
- is\_jack\_running
  - seq64::perform, 696
- is\_left
  - seq64::click, 157
- is\_letter
  - seq64::keystroke, 384
- is\_linked
  - seq64::event, 203
- is\_marked
  - seq64::event, 204
- is\_master
  - seq64::jack\_assistant, 319
- is\_middle
  - seq64::click, 157
- is\_modified
  - seq64::event\_list, 216
  - seq64::perform, 692, 740
  - seq64::seqmenu, 935, 936
- is\_more\_input
  - seq64::mastermidibase, 446
- is\_mseq\_valid
  - seq64::perform, 743
- is\_no\_modifier
  - seq64, 103
- is\_note
  - seq64::event, 206
- is\_note\_msg
  - seq64::event, 196
- is\_note\_off
  - seq64::event, 206
- is\_note\_off\_recorded
  - seq64::event, 207
- is\_note\_on
  - seq64::event, 206
- is\_null\_midipulse
  - seq64, 98
- is\_one\_byte\_msg
  - seq64::event, 196
- is\_output\_port
  - seq64::midibase, 591
- is\_painted
  - seq64::event, 204
- is\_pattern\_playing
  - seq64::perform, 696, 742
- is\_port\_open
  - seq64::midi\_api, 488
  - seq64::rtmidi, 827
- is\_press
  - seq64::click, 157
  - seq64::keystroke, 384
- is\_realized
  - seq64::gui\_window\_gtk2, 312
- is\_right
  - seq64::click, 157
- is\_running
  - seq64::jack\_assistant, 319
  - seq64::perform, 696
- is\_save
  - seq64::mainwnd, 428
- is\_screenset\_valid
  - seq64::perform, 741
- is\_selected
  - seq64::event, 204
- is\_seq\_valid
  - seq64::perform, 742
- is\_sequence\_in\_edit

- seq64::perform, 702
- is\_shift\_key
  - seq64, 103, 104
- is\_smf\_0
  - seq64::perform, 719
  - seq64::sequence, 1008
- is\_strict\_note\_msg
  - seq64::event, 197
- is\_super\_key
  - seq64, 104
- is\_sysex\_special\_id
  - seq64::midifile, 634
- is\_system\_port
  - seq64::busarray, 146
  - seq64::midi\_api, 485
  - seq64::midibase, 591
- is\_two\_byte\_msg
  - seq64::event, 196
- is\_valid
  - seq64::user\_instrument, 1076
  - seq64::user\_midi\_bus, 1081
- is\_virtual\_port
  - seq64::midi\_api, 485
  - seq64::midibase, 590
- iterator
  - seq64::editable\_events, 181
  - seq64::event\_list, 214
  - seq64::seqmenu, 934
- jack\_assistant
  - seq64::jack\_assistant, 318
  - seq64::perform, 747
- jack\_data
  - seq64::midi\_jack, 527
- jack\_dialog
  - seq64::mainwnd, 426
- jack\_dummy\_callback
  - seq64, 100
- jack\_frame\_rate
  - seq64::jack\_assistant, 326
- jack\_idle\_connect
  - seq64::gui\_assistant, 276
  - seq64::gui\_assistant\_gtk2, 279
- jack\_message\_bit\_bucket
  - seq64, 109
- jack\_process\_io
  - seq64, 109
  - seq64::midi\_jack\_info, 545
- jack\_process\_rtmidi\_input
  - seq64, 107
- jack\_process\_rtmidi\_output
  - seq64, 108
- jack\_session\_callback
  - seq64, 96
  - seq64::jack\_assistant, 334
- jack\_session\_uuid
  - seq64::rc\_settings, 804, 809
- jack\_shutdown
  - seq64::perform, 749
- jack\_shutdown\_callback
  - seq64, 93
  - seq64::jack\_assistant, 332
- jack\_sync\_callback
  - seq64, 92
  - seq64::jack\_assistant, 332
  - seq64::perform, 749
- jack\_timebase\_callback
  - seq64, 93
  - seq64::jack\_assistant, 333
  - seq64::perform, 750
- jack\_transport\_callback
  - seq64, 94
  - seq64::jack\_assistant, 332
  - seq64::perform, 749
- jf\_bit
  - seq64::jack\_status\_pair\_t, 340
- jf\_meaning
  - seq64::jack\_status\_pair\_t, 341
- js\_clock\_tick
  - seq64::jack\_scratchpad, 339
- js\_current\_tick
  - seq64::jack\_scratchpad, 339
- js\_delta\_tick\_frac
  - seq64::jack\_scratchpad, 340
- js\_dumping
  - seq64::jack\_scratchpad, 339
- js\_init\_clock
  - seq64::jack\_scratchpad, 339
- js\_jack\_stopped
  - seq64::jack\_scratchpad, 339
- js\_looping
  - seq64::jack\_scratchpad, 339
- js\_playback\_mode
  - seq64::jack\_scratchpad, 339
- js\_ticks\_converted
  - seq64::jack\_scratchpad, 340
- js\_ticks\_converted\_last
  - seq64::jack\_scratchpad, 340
- js\_ticks\_delta
  - seq64::jack\_scratchpad, 340
- js\_total\_tick
  - seq64::jack\_scratchpad, 339
- keep\_queue
  - seq64::keys\_perform, 354
- Key
  - seq64::editable\_events, 181
- key
  - seq64::keystroke, 384
- key\_name
  - seq64::keys\_perform, 364
  - seq64::keys\_perform\_gtk2, 376
  - seq64::perform, 712
- key\_press\_event
  - seq64::perftime, 791
- keybindentry
  - seq64::keybindentry, 342
  - seq64::perform, 747

- keys
  - seq64::gui\_assistant, [277](#)
  - seq64::perform, [695](#)
- keys\_perform
  - seq64::keys\_perform, [351](#)
- keys\_perform\_gtk2
  - seq64::keys\_perform\_gtk2, [376](#)
- keystroke
  - seq64::keystroke, [382](#), [383](#)
- keyval\_name
  - seq64, [97](#)
- keyval\_normalize
  - seq64, [97](#)
- kpt\_bpm\_dn
  - seq64::keys\_perform\_transfer, [378](#)
- kpt\_bpm\_up
  - seq64::keys\_perform\_transfer, [378](#)
- kpt\_event\_edit
  - seq64::keys\_perform\_transfer, [380](#)
- kpt\_fast\_forward
  - seq64::keys\_perform\_transfer, [381](#)
- kpt\_follow\_transport
  - seq64::keys\_perform\_transfer, [381](#)
- kpt\_group\_learn
  - seq64::keys\_perform\_transfer, [378](#)
- kpt\_group\_off
  - seq64::keys\_perform\_transfer, [378](#)
- kpt\_group\_on
  - seq64::keys\_perform\_transfer, [378](#)
- kpt\_keep\_queue
  - seq64::keys\_perform\_transfer, [379](#)
- kpt\_menu\_mode
  - seq64::keys\_perform\_transfer, [380](#)
- kpt\_pattern\_edit
  - seq64::keys\_perform\_transfer, [380](#)
- kpt\_pause
  - seq64::keys\_perform\_transfer, [380](#)
- kpt\_pointer\_position
  - seq64::keys\_perform\_transfer, [381](#)
- kpt\_queue
  - seq64::keys\_perform\_transfer, [379](#)
- kpt\_replace
  - seq64::keys\_perform\_transfer, [379](#)
- kpt\_rewind
  - seq64::keys\_perform\_transfer, [381](#)
- kpt\_screenset\_dn
  - seq64::keys\_perform\_transfer, [378](#)
- kpt\_screenset\_up
  - seq64::keys\_perform\_transfer, [378](#)
- kpt\_set\_playing\_screenset
  - seq64::keys\_perform\_transfer, [378](#)
- kpt\_show\_ui\_sequence\_key
  - seq64::keys\_perform\_transfer, [379](#)
- kpt\_show\_ui\_sequence\_number
  - seq64::keys\_perform\_transfer, [380](#)
- kpt\_snapshot\_1
  - seq64::keys\_perform\_transfer, [379](#)
- kpt\_snapshot\_2
  - seq64::keys\_perform\_transfer, [379](#)
- kpt\_song\_mode
  - seq64::keys\_perform\_transfer, [380](#)
- kpt\_start
  - seq64::keys\_perform\_transfer, [379](#)
- kpt\_stop
  - seq64::keys\_perform\_transfer, [379](#)
- kpt\_tap\_bpm
  - seq64::keys\_perform\_transfer, [380](#)
- kpt\_toggle\_jack
  - seq64::keys\_perform\_transfer, [380](#)
- kpt\_toggle\_mutes
  - seq64::keys\_perform\_transfer, [381](#)
- lash
  - seq64::lash, [387](#)
- lash\_driver
  - seq64, [98](#)
- lash\_support
  - seq64::rc\_settings, [802](#), [806](#)
- lash\_support\_callback
  - seq64::options, [643](#)
- lash\_timeout\_connect
  - seq64::gui\_assistant, [276](#)
  - seq64::gui\_assistant\_gtk2, [279](#)
- last\_used\_dir
  - seq64::rc\_settings, [805](#), [809](#)
- launch
  - seq64::perform, [700](#)
- launch\_input\_thread
  - seq64::perform, [739](#)
- launch\_output\_thread
  - seq64::perform, [739](#)
- learn\_toggle
  - seq64::mainwnd, [424](#)
  - seq64::perform, [717](#)
- left\_right\_size
  - seq64::perform, [705](#)
- legacy\_format
  - seq64::rc\_settings, [802](#), [806](#)
- length
  - seq64::trigger, [1054](#)
- lfownd
  - seq64::lfownd, [392](#)
  - seq64::seqdata, [869](#)
- light\_grey
  - seq64::gui\_palette\_gtk2, [304](#)
- line\_after
  - seq64::configfile, [164](#)
- line\_color
  - seq64::gui\_palette\_gtk2, [303](#)
- line\_count
  - seq64::eventslots, [244](#)
- line\_increment
  - seq64::eventslots, [244](#)
- line\_maximum
  - seq64::eventslots, [244](#)
- link
  - seq64::event, [203](#)

- link\_new
  - seq64::event\_list, 218
  - seq64::sequence, 1035
- List
  - seq64::triggers, 1060
- load\_events
  - seq64::editable\_events, 183
  - seq64::eventslots, 245
- load\_inverse\_palette
  - seq64::gui\_palette\_gtk2, 302
- lock
  - seq64::mutex, 638
- log2\_time\_sig\_value
  - seq64, 78
- log\_main\_sequence
  - seq64::midi\_splitter, 571
- long\_options
  - seq64, 124
- lookup\_keyevent\_key
  - seq64::keys\_perform, 363
  - seq64::perform, 714
- lookup\_keyevent\_seq
  - seq64::keys\_perform, 363
  - seq64::perform, 714
- lookup\_keygroup\_group
  - seq64::keys\_perform, 364
  - seq64::perform, 715
- lookup\_keygroup\_key
  - seq64::keys\_perform, 363
  - seq64::perform, 715
- m\_32nds\_per\_quarter
  - seq64::perform, 757
  - seq64::sequence, 1051
- m\_4bar\_offset
  - seq64::perfroll, 781
  - seq64::perftime, 792
- m\_FF\_RW\_button\_type
  - seq64::perform, 752
- m\_active
  - seq64::businfo, 154
  - seq64::midi\_control, 505
- m\_adding
  - seq64::AbstractPerfInput, 136
  - seq64::Seq24SeqEventInput, 858
  - seq64::seqroll, 973
- m\_adding\_pressed
  - seq64::AbstractPerfInput, 136
- m\_adjust\_bpm
  - seq64::mainwnd, 433
- m\_adjust\_load\_offset
  - seq64::mainwnd, 433
- m\_adjust\_ss
  - seq64::mainwnd, 433
- m\_allow\_click\_edit
  - seq64::rc\_settings, 814
- m\_allow\_mod4\_mode
  - seq64::rc\_settings, 813
- m\_allow\_snap\_split
  - seq64::rc\_settings, 813
- m\_allow\_two\_perfedits
  - seq64::user\_settings, 1113
- m\_alsa\_seq
  - seq64::mastermidibus, 465
  - seq64::midi\_alsa\_info, 481
- m\_api\_data
  - seq64::rtmidi\_in\_data, 839
- m\_app\_client\_name
  - seq64::rc\_settings, 817
- m\_app\_name
  - seq64::midi\_info, 522
- m\_application\_name
  - seq64::rc\_settings, 817
- m\_armed\_progress\_color
  - seq64::mainwid, 412
- m\_armed\_saved
  - seq64::perform, 752
- m\_armed\_statuses
  - seq64::perform, 752
- m\_auto\_option\_save
  - seq64::rc\_settings, 813
- m\_b\_on\_c\_pixmap
  - seq64::font, 263
- m\_b\_on\_y\_pixmap
  - seq64::font, 262
- m\_back
  - seq64::midi\_queue, 568
- m\_background
  - seq64::gui\_drawingarea\_gtk2, 296
- m\_background\_sequence
  - seq64::seqroll, 975
  - seq64::sequence, 1052
- m\_background\_x
  - seq64::perfroll, 780
- m\_bar\_width
  - seq64::maintime, 398
- m\_beat\_length
  - seq64::perfroll, 781
- m\_beat\_width
  - seq64::jack\_assistant, 337
  - seq64::maintime, 398
  - seq64::midi\_timing, 577
  - seq64::perform, 756
- m\_beats
  - seq64::midi\_measures, 553
- m\_beats\_per\_bar
  - seq64::perform, 756
- m\_beats\_per\_measure
  - seq64::jack\_assistant, 337
  - seq64::midi\_timing, 577
- m\_beats\_per\_minute
  - seq64::jack\_assistant, 338
  - seq64::mastermidibase, 455
  - seq64::midi\_timing, 577
- m\_bg\_color
  - seq64::gui\_palette\_gtk2, 309
- m\_bgsequence



- seq64::seqedit, [893](#)
- m\_black
  - seq64::gui\_palette\_gtk2, [306](#)
- m\_black\_pixmap
  - seq64::font, [262](#)
- m\_blk\_key
  - seq64::gui\_palette\_gtk2, [308](#)
- m\_blk\_paint
  - seq64::gui\_palette\_gtk2, [308](#)
- m\_blue
  - seq64::gui\_palette\_gtk2, [308](#)
- m\_bottbox
  - seq64::eventedit, [236](#)
- m\_bottom\_iterator
  - seq64::eventslots, [257](#)
- m\_box\_height
  - seq64::maintime, [399](#)
- m\_box\_less\_pill
  - seq64::maintime, [399](#)
- m\_box\_width
  - seq64::maintime, [399](#)
- m\_bpm
  - seq64::midi\_info, [522](#)
  - seq64::midibase, [602](#)
  - seq64::perfedit, [667](#)
  - seq64::perform, [756](#)
- m\_bpm\_page\_increment
  - seq64::user\_settings, [1116](#)
- m\_bpm\_precision
  - seq64::user\_settings, [1116](#)
- m\_bpm\_step\_increment
  - seq64::user\_settings, [1116](#)
- m\_bus
  - seq64::businfo, [153](#)
  - seq64::sequence, [1047](#)
- m\_bus\_announce
  - seq64::mastermidibase, [454](#)
- m\_bus\_container
  - seq64::midi\_info, [521](#)
- m\_bus\_id
  - seq64::midibase, [602](#)
- m\_bus\_index
  - seq64::midibase, [602](#)
- m\_bus\_name
  - seq64::midibase, [603](#)
- m\_button
  - seq64::click, [159](#)
- m\_button\_bpm
  - seq64::perfedit, [666](#)
  - seq64::seqedit, [902](#)
- m\_button\_bus
  - seq64::seqedit, [899](#)
- m\_button\_bw
  - seq64::perfedit, [666](#)
  - seq64::seqedit, [902](#)
- m\_button\_cancel
  - seq64::eventedit, [237](#)
- m\_button\_channel
  - seq64::seqedit, [899](#)
- m\_button\_chord
  - seq64::seqedit, [901](#)
- m\_button\_collapse
  - seq64::perfedit, [665](#)
- m\_button\_copy
  - seq64::perfedit, [665](#)
- m\_button\_data
  - seq64::seqedit, [902](#)
- m\_button\_del
  - seq64::eventedit, [236](#)
- m\_button\_down
  - seq64::mainwid, [412](#)
- m\_button\_expand
  - seq64::perfedit, [665](#)
- m\_button\_follow
  - seq64::perfedit, [666](#)
- m\_button\_grow
  - seq64::perfedit, [665](#)
- m\_button\_ins
  - seq64::eventedit, [236](#)
- m\_button\_jack
  - seq64::mainwnd, [433](#)
  - seq64::perfedit, [665](#)
- m\_button\_jack\_connect
  - seq64::options, [645](#)
- m\_button\_jack\_disconnect
  - seq64::options, [645](#)
- m\_button\_jack\_master
  - seq64::options, [645](#)
- m\_button\_jack\_master\_cond
  - seq64::options, [645](#)
- m\_button\_jack\_transport
  - seq64::options, [645](#)
- m\_button\_key
  - seq64::seqedit, [901](#)
- m\_button\_learn
  - seq64::mainwnd, [432](#)
- m\_button\_length
  - seq64::seqedit, [900](#)
- m\_button\_lfo
  - seq64::seqedit, [898](#)
- m\_button\_loop
  - seq64::perfedit, [665](#)
- m\_button\_modify
  - seq64::eventedit, [236](#)
- m\_button\_note\_length
  - seq64::seqedit, [900](#)
- m\_button\_ok
  - seq64::options, [645](#)
- m\_button\_perfedit
  - seq64::mainwnd, [432](#)
- m\_button\_play
  - seq64::mainwnd, [432](#)
  - seq64::perfedit, [664](#)
- m\_button\_quantize
  - seq64::seqedit, [899](#)
- m\_button\_rec\_vol



- seq64::seqedit, [902](#)
- m\_button\_redo
  - seq64::perfedit, [665](#)
  - seq64::seqedit, [899](#)
- m\_button\_save
  - seq64::eventedit, [237](#)
- m\_button\_scale
  - seq64::seqedit, [901](#)
- m\_button\_sequence
  - seq64::seqedit, [899](#)
- m\_button\_snap
  - seq64::perfedit, [664](#)
  - seq64::seqedit, [900](#)
- m\_button\_stop
  - seq64::mainwnd, [432](#)
  - seq64::perfedit, [664](#)
- m\_button\_tools
  - seq64::seqedit, [899](#)
- m\_button\_undo
  - seq64::perfedit, [665](#)
  - seq64::seqedit, [898](#)
- m\_button\_xpose
  - seq64::perfedit, [664](#)
- m\_button\_zoom
  - seq64::seqedit, [900](#)
- m\_bw
  - seq64::perfedit, [667](#)
- m\_bytes
  - seq64::midi\_message, [555](#)
- m\_c\_on\_b\_pixmap
  - seq64::font, [263](#)
- m\_call\_seq\_edit
  - seq64::mainwnd, [434](#)
- m\_call\_seq\_eventedit
  - seq64::mainwnd, [434](#)
- m\_category
  - seq64::editable\_event, [178](#)
- m\_cc
  - seq64::seqdata, [870](#)
  - seq64::seqevent, [918](#)
  - seq64::seqroll, [976](#)
- m\_cell\_h
  - seq64::font, [261](#)
- m\_cell\_w
  - seq64::font, [261](#)
- m\_channel
  - seq64::event, [208](#)
- m\_channel\_count
  - seq64::user\_midi\_bus, [1083](#)
- m\_channel\_match
  - seq64::sequence, [1047](#)
- m\_char\_list
  - seq64::midi\_list, [550](#)
  - seq64::midifile, [636](#)
- m\_char\_vector
  - seq64::midi\_vector, [582](#)
- m\_char\_w
  - seq64::eventslots, [256](#)
- seq64::perfnames, [675](#)
- m\_chord
  - seq64::seqedit, [893](#)
  - seq64::seqroll, [973](#)
- m\_client
  - seq64::lash, [389](#)
- m\_client\_name
  - seq64::midi\_in\_jack, [511](#)
  - seq64::midi\_port\_info::port\_info\_t, [794](#)
- m\_client\_number
  - seq64::midi\_port\_info::port\_info\_t, [794](#)
- m\_clip\_mask
  - seq64::font, [263](#)
- m\_clipboard
  - seq64::seqmenu, [942](#)
  - seq64::triggers, [1072](#)
- m\_clock\_mod
  - seq64::midibase, [601](#)
- m\_clock\_type
  - seq64::midibase, [602](#)
- m\_clocks\_per\_metronome
  - seq64::perform, [756](#)
  - seq64::sequence, [1051](#)
- m\_cond
  - seq64::condition\_var, [161](#)
- m\_condition\_var
  - seq64::perform, [761](#)
- m\_config\_directory
  - seq64::rc\_settings, [816](#)
- m\_config\_filename
  - seq64::rc\_settings, [816](#)
- m\_config\_filename\_alt
  - seq64::rc\_settings, [816](#)
- m\_connected
  - seq64::midi\_api, [491](#)
- m\_container
  - seq64::busarray, [148](#)
- m\_continue\_sysex
  - seq64::rtmidi\_in\_data, [839](#)
- m\_control\_height
  - seq64::user\_settings, [1111](#)
- m\_control\_status
  - seq64::perform, [759](#)
- m\_controller\_count
  - seq64::user\_instrument, [1078](#)
- m\_current\_event
  - seq64::editable\_events, [187](#)
- m\_current\_index
  - seq64::eventslots, [257](#)
- m\_current\_iterator
  - seq64::eventslots, [257](#)
- m\_current\_seq
  - seq64::seqmenu, [942](#)
- m\_current\_x
  - seq64::FruityPerfInput, [268](#)
  - seq64::gui\_drawingarea\_gtk2, [297](#)
- m\_current\_y
  - seq64::FruityPerfInput, [268](#)

- seq64::gui\_drawingarea\_gtk2, 297
- m\_current\_zoom
  - seq64::user\_settings, 1111
- m\_d
  - seq64::configfile, 165
- m\_data
  - seq64::event, 208
  - seq64::midi\_control, 505
  - seq64::midifile, 635
- m\_dest\_addr\_client
  - seq64::midi\_alsa, 475
  - seq64::midibus, 615
- m\_dest\_addr\_port
  - seq64::midi\_alsa, 475
  - seq64::midibus, 615
- m\_device\_ignore
  - seq64::rc\_settings, 815
- m\_device\_ignore\_num
  - seq64::rc\_settings, 815
- m\_dirty\_edit
  - seq64::sequence, 1049
- m\_dirty\_main
  - seq64::sequence, 1049
- m\_dirty\_names
  - seq64::sequence, 1049
- m\_dirty\_perf
  - seq64::sequence, 1049
- m\_disable\_reported
  - seq64::midifile, 635
- m\_display\_name
  - seq64::midibase, 603
- m\_divisions
  - seq64::midi\_measures, 553
- m\_divs\_per\_beat
  - seq64::perroll, 780
- m\_dk\_blue
  - seq64::gui\_palette\_gtk2, 307
- m\_dk\_cyan
  - seq64::gui\_palette\_gtk2, 307
- m\_dk\_green
  - seq64::gui\_palette\_gtk2, 306
- m\_dk\_grey
  - seq64::gui\_palette\_gtk2, 308
- m\_dk\_magenta
  - seq64::gui\_palette\_gtk2, 307
- m\_dk\_orange
  - seq64::gui\_palette\_gtk2, 307
- m\_dk\_red
  - seq64::gui\_palette\_gtk2, 306
- m\_do\_input
  - seq64::rtmidi\_in\_data, 838
- m\_dont\_reset\_ticks
  - seq64::perform, 759
- m\_drag\_handle
  - seq64::seqdata, 871
- m\_drag\_paste\_start\_pos
  - seq64::FruitySeqRollInput, 274
- m\_dragging
  - seq64::seqdata, 871
- m\_drawing\_background\_seq
  - seq64::seqroll, 975
- m\_drop\_sequence
  - seq64::perroll, 782
- m\_drop\_tick
  - seq64::perroll, 782
- m\_drop\_tick\_trigger\_offset
  - seq64::perroll, 782
- m\_drop\_x
  - seq64::gui\_drawingarea\_gtk2, 297
- m\_drop\_y
  - seq64::gui\_drawingarea\_gtk2, 298
- m\_dumping\_input
  - seq64::mastermidibase, 455
- m\_edit\_sequence
  - seq64::perform, 760
- m\_editbox
  - seq64::eventedit, 236
- m\_editing
  - seq64::sequence, 1049
- m\_editing\_cc
  - seq64::seqedit, 903
- m\_editing\_status
  - seq64::seqedit, 903
- m\_effective\_tick
  - seq64::Seq24PerfInput, 856
- m\_entry\_bpm
  - seq64::perfedit, 666
  - seq64::seqedit, 902
- m\_entry\_bus
  - seq64::seqedit, 899
- m\_entry\_bw
  - seq64::perfedit, 666
  - seq64::seqedit, 902
- m\_entry\_channel
  - seq64::seqedit, 900
- m\_entry\_chord
  - seq64::seqedit, 901
- m\_entry\_data
  - seq64::seqedit, 902
- m\_entry\_ev\_data\_0
  - seq64::eventedit, 238
- m\_entry\_ev\_data\_1
  - seq64::eventedit, 238
- m\_entry\_ev\_name
  - seq64::eventedit, 238
- m\_entry\_ev\_timestamp
  - seq64::eventedit, 238
- m\_entry\_key
  - seq64::seqedit, 901
- m\_entry\_length
  - seq64::seqedit, 901
- m\_entry\_name
  - seq64::seqedit, 903
- m\_entry\_note\_length
  - seq64::seqedit, 900
- m\_entry\_notes

- seq64::mainwnd, [434](#)
- m\_entry\_scale
  - seq64::seqedit, [901](#)
- m\_entry\_sequence
  - seq64::seqedit, [899](#)
- m\_entry\_snap
  - seq64::perfedit, [664](#)
  - seq64::seqedit, [900](#)
- m\_entry\_xpose
  - seq64::perfedit, [664](#)
- m\_entry\_zoom
  - seq64::seqedit, [900](#)
- m\_erase\_painting
  - seq64::FruitySeqRollInput, [274](#)
- m\_error\_callback
  - seq64::midi\_api, [491](#)
- m\_error\_callback\_user\_data
  - seq64::midi\_api, [491](#)
- m\_error\_is\_fatal
  - seq64::midifile, [635](#)
- m\_error\_message
  - seq64::configfile, [165](#)
  - seq64::midifile, [635](#)
- m\_error\_string
  - seq64::midi\_api, [491](#)
  - seq64::midi\_info, [522](#)
- m\_event\_container
  - seq64::eventslots, [255](#)
- m\_event\_count
  - seq64::eventslots, [256](#)
- m\_eventedit
  - seq64::seqmenu, [942](#)
- m\_events
  - seq64::editable\_events, [187](#)
  - seq64::event\_list, [222](#)
  - seq64::sequence, [1046](#)
- m\_events\_clipboard
  - seq64::sequence, [1046](#)
- m\_events\_redo
  - seq64::sequence, [1047](#)
- m\_events\_undo
  - seq64::sequence, [1047](#)
- m\_events\_undo\_hold
  - seq64::sequence, [1046](#)
- m\_eventslots
  - seq64::eventedit, [235](#)
- m\_excell\_FF\_RW
  - seq64::perform, [752](#)
- m\_fg\_color
  - seq64::gui\_palette\_gtk2, [309](#)
- m\_file\_size
  - seq64::midifile, [635](#)
- m\_filename
  - seq64::rc\_settings, [816](#)
- m\_filter\_by\_channel
  - seq64::mastermidibase, [455](#)
  - seq64::rc\_settings, [815](#)
- m\_first\_error\_occurred
  - seq64::midi\_api, [491](#)
- m\_first\_message
  - seq64::rtmidi\_in\_data, [839](#)
- m\_flash\_height
  - seq64::maintime, [399](#)
- m\_flash\_width
  - seq64::maintime, [399](#)
- m\_flash\_x
  - seq64::maintime, [399](#)
- m\_follow\_transport
  - seq64::jack\_assistant, [337](#)
- m\_font\_h
  - seq64::font, [261](#)
- m\_font\_w
  - seq64::font, [261](#)
- m\_foreground
  - seq64::gui\_drawingarea\_gtk2, [297](#)
- m\_format\_timestamp
  - seq64::editable\_event, [178](#)
- m\_front
  - seq64::midi\_queue, [568](#)
- m\_fruity\_interaction
  - seq64::perfroll, [783](#)
  - seq64::seqevent, [916](#)
  - seq64::seqroll, [972](#)
- m\_gc
  - seq64::gui\_drawingarea\_gtk2, [296](#)
- m\_global\_bgsequence
  - seq64::midifile, [636](#)
- m\_global\_queue
  - seq64::midi\_info, [522](#)
- m\_global\_seq\_feature\_save
  - seq64::user\_settings, [1112](#)
- m\_gmute\_tracks
  - seq64::user\_settings, [1117](#)
- m\_green
  - seq64::gui\_palette\_gtk2, [308](#)
- m\_grey
  - seq64::gui\_palette\_gtk2, [308](#)
- m\_grid\_brackets
  - seq64::user\_settings, [1110](#)
- m\_grid\_style
  - seq64::user\_settings, [1110](#)
- m\_grow\_direction
  - seq64::perfroll, [783](#)
- m\_growing
  - seq64::perfroll, [783](#)
  - seq64::seqevent, [917](#)
  - seq64::seqroll, [973](#)
- m\_gui\_support
  - seq64::perform, [762](#)
- m\_h\_page\_increment
  - seq64::perfroll, [779](#)
- m\_h\_perf\_page\_increment
  - seq64::user\_settings, [1113](#)
- m\_hadjust
  - seq64::gui\_drawingarea\_gtk2, [296](#)
  - seq64::perfedit, [663](#)

- seq64::seqedit, [897](#)
- m\_has\_link
  - seq64::event, [208](#)
- m\_have\_button\_press
  - seq64::perfroll, [781](#)
- m\_have\_focus
  - seq64::eventedit, [239](#)
  - seq64::seqedit, [903](#)
- m\_have\_redo
  - seq64::perform, [761](#)
  - seq64::sequence, [1047](#)
- m\_have\_undo
  - seq64::perform, [761](#)
  - seq64::sequence, [1046](#)
- m\_hbox
  - seq64::lfownd, [393](#)
  - seq64::perfedit, [666](#)
  - seq64::seqedit, [898](#)
- m\_hbox2
  - seq64::seqedit, [898](#)
- m\_hint\_key
  - seq64::seqkeys, [929](#)
- m\_hint\_state
  - seq64::seqkeys, [929](#)
- m\_hlbox
  - seq64::perfedit, [666](#)
- m\_horizontal\_adjust
  - seq64::seqroll, [971](#)
- m\_hscroll
  - seq64::perfedit, [663](#)
- m\_hscroll\_new
  - seq64::seqedit, [897](#)
- m\_htopbox
  - seq64::eventedit, [235](#)
- m\_ignore\_flags
  - seq64::rtmidi\_in\_data, [838](#)
- m\_image\_play
  - seq64::mainwnd, [432](#)
  - seq64::perfedit, [664](#)
- m\_image\_transpose
  - seq64::seqedit, [895](#)
- m\_in\_thread
  - seq64::perform, [755](#)
- m\_in\_thread\_launched
  - seq64::perform, [755](#)
- m\_inbus\_array
  - seq64::mastermidibase, [454](#)
- m\_info\_api
  - seq64::rtmidi\_info, [849](#)
- m\_init\_clock
  - seq64::businfo, [154](#)
- m\_init\_input
  - seq64::businfo, [154](#)
- m\_initial\_chord
  - seq64::seqedit, [892](#)
- m\_initial\_note\_length
  - seq64::seqedit, [892](#)
- m\_initial\_snap
  - seq64::seqedit, [892](#)
- m\_initial\_zoom
  - seq64::seqedit, [892](#)
- m\_initialized
  - seq64::businfo, [154](#)
- m\_input
  - seq64::midi\_info, [521](#)
- m\_input\_data
  - seq64::midi\_api, [491](#)
- m\_input\_port\_name
  - seq64::midi\_alsa, [475](#)
  - seq64::midibus, [616](#)
- m\_inputting
  - seq64::midibase, [602](#)
  - seq64::perform, [755](#)
- m\_instrument\_def
  - seq64::user\_instrument, [1078](#)
- m\_instruments
  - seq64::user\_settings, [1110](#)
- m\_interaction
  - seq64::perfroll, [783](#)
- m\_interaction\_method
  - seq64::rc\_settings, [815](#)
- m\_inverse\_active
  - seq64::midi\_control, [505](#)
- m\_inverse\_colors
  - seq64::user\_settings, [1113](#)
- m\_is\_drag\_pasting
  - seq64::FruitySeqEventInput, [271](#)
  - seq64::seqroll, [974](#)
- m\_is\_drag\_pasting\_start
  - seq64::FruitySeqEventInput, [271](#)
  - seq64::seqroll, [974](#)
- m\_is\_input
  - seq64::midi\_port\_info::port\_info\_t, [795](#)
- m\_is\_input\_port
  - seq64::midibase, [603](#)
- m\_is\_inverse
  - seq64::gui\_palette\_gtk2, [306](#)
- m\_is\_lash\_supported
  - seq64::lash, [389](#)
- m\_is\_modified
  - seq64::event\_list, [222](#)
  - seq64::perform, [761](#)
- m\_is\_pattern\_playing
  - seq64::perform, [755](#)
- m\_is\_press
  - seq64::click, [158](#)
  - seq64::keystroke, [386](#)
- m\_is\_realized
  - seq64::gui\_window\_gtk2, [314](#)
- m\_is\_running
  - seq64::mainwnd, [434](#)
  - seq64::perfedit, [667](#)
- m\_is\_system
  - seq64::midi\_port\_info::port\_info\_t, [795](#)
- m\_is\_system\_port
  - seq64::midibase, [603](#)

- m\_is\_valid
  - seq64::user\_instrument, 1078
  - seq64::user\_midi\_bus, 1083
- m\_is\_virtual
  - seq64::midi\_port\_info::port\_info\_t, 795
- m\_is\_virtual\_port
  - seq64::midibase, 603
- m\_iterator\_draw
  - seq64::sequence, 1047
- m\_iterator\_draw\_trigger
  - seq64::triggers, 1073
- m\_iterator\_play\_trigger
  - seq64::triggers, 1073
- m\_jack\_asst
  - seq64::perform, 761
- m\_jack\_buffmessage
  - seq64::midi\_jack\_data, 538
- m\_jack\_buffsize
  - seq64::midi\_jack\_data, 538
- m\_jack\_client
  - seq64::jack\_assistant, 335
  - seq64::midi\_jack\_data, 538
  - seq64::midi\_jack\_info, 546
- m\_jack\_client\_name
  - seq64::jack\_assistant, 335
- m\_jack\_client\_uuid
  - seq64::jack\_assistant, 335
- m\_jack\_data
  - seq64::midi\_jack, 536
- m\_jack\_frame\_current
  - seq64::jack\_assistant, 335
- m\_jack\_frame\_last
  - seq64::jack\_assistant, 335
- m\_jack\_frame\_rate
  - seq64::jack\_assistant, 337
- m\_jack\_info
  - seq64::midi\_jack, 536
- m\_jack\_lasttime
  - seq64::midi\_jack\_data, 538
- m\_jack\_master
  - seq64::jack\_assistant, 336
- m\_jack\_parent
  - seq64::jack\_assistant, 335
- m\_jack\_port
  - seq64::midi\_jack\_data, 538
- m\_jack\_ports
  - seq64::midi\_jack\_info, 546
- m\_jack\_pos
  - seq64::jack\_assistant, 336
- m\_jack\_rtmidiin
  - seq64::midi\_jack\_data, 538
- m\_jack\_running
  - seq64::jack\_assistant, 336
- m\_jack\_session\_uuid
  - seq64::rc\_settings, 816
- m\_jack\_stop\_tick
  - seq64::jack\_assistant, 337
- m\_jack\_tick
  - seq64::jack\_assistant, 336
  - seq64::perform, 758
- m\_jack\_transport\_state
  - seq64::jack\_assistant, 336
- m\_jack\_transport\_state\_last
  - seq64::jack\_assistant, 336
- m\_session\_ev
  - seq64::jack\_assistant, 336
- m\_justselected\_one
  - seq64::FruitySeqEventInput, 271
  - seq64::seqroll, 974
- m\_key
  - seq64::keybindentry, 343
  - seq64::keystroke, 386
  - seq64::seqedit, 893
  - seq64::seqkeys, 929
  - seq64::seqroll, 973
- m\_key\_bpm\_dn
  - seq64::keys\_perform, 370
- m\_key\_bpm\_up
  - seq64::keys\_perform, 370
- m\_key\_event\_edit
  - seq64::keys\_perform, 373
- m\_key\_events
  - seq64::keys\_perform, 370
- m\_key\_events\_rev
  - seq64::keys\_perform, 370
- m\_key\_fast\_forward
  - seq64::keys\_perform, 373
- m\_key\_follow\_transport
  - seq64::keys\_perform, 373
- m\_key\_group\_learn
  - seq64::keys\_perform, 372
- m\_key\_group\_off
  - seq64::keys\_perform, 372
- m\_key\_group\_on
  - seq64::keys\_perform, 372
- m\_key\_groups
  - seq64::keys\_perform, 370
- m\_key\_groups\_rev
  - seq64::keys\_perform, 370
- m\_key\_keep\_queue
  - seq64::keys\_perform, 371
- m\_key\_menu\_mode
  - seq64::keys\_perform, 372
- m\_key\_pattern\_edit
  - seq64::keys\_perform, 373
- m\_key\_pause
  - seq64::keys\_perform, 372
- m\_key\_pointer\_position
  - seq64::keys\_perform, 373
- m\_key\_queue
  - seq64::keys\_perform, 371
- m\_key\_replace
  - seq64::keys\_perform, 371
- m\_key\_rewind
  - seq64::keys\_perform, 373
- m\_key\_screenset\_dn

- seq64::keys\_perform, 371
- m\_key\_screenset\_up
  - seq64::keys\_perform, 371
- m\_key\_set\_playing\_screenset
  - seq64::keys\_perform, 371
- m\_key\_show\_ui\_sequence\_key
  - seq64::keys\_perform, 370
- m\_key\_show\_ui\_sequence\_number
  - seq64::keys\_perform, 370
- m\_key\_snapshot\_1
  - seq64::keys\_perform, 371
- m\_key\_snapshot\_2
  - seq64::keys\_perform, 371
- m\_key\_song\_mode
  - seq64::keys\_perform, 372
- m\_key\_start
  - seq64::keys\_perform, 372
- m\_key\_stop
  - seq64::keys\_perform, 374
- m\_key\_tap\_bpm
  - seq64::keys\_perform, 373
- m\_key\_toggle\_jack
  - seq64::keys\_perform, 372
- m\_key\_toggle\_mutes
  - seq64::keys\_perform, 373
- m\_keying
  - seq64::seqkeys, 929
- m\_keying\_note
  - seq64::seqkeys, 929
- m\_keys\_perform
  - seq64::gui\_assistant, 277
- m\_label\_category
  - seq64::eventedit, 238
- m\_label\_channel
  - seq64::eventedit, 237
- m\_label\_ev\_count
  - seq64::eventedit, 237
- m\_label\_modified
  - seq64::eventedit, 238
- m\_label\_ppqn
  - seq64::eventedit, 237
- m\_label\_right
  - seq64::eventedit, 238
- m\_label\_seq\_name
  - seq64::eventedit, 237
- m\_label\_spacer
  - seq64::eventedit, 237
- m\_label\_time\_fmt
  - seq64::eventedit, 238
- m\_label\_time\_sig
  - seq64::eventedit, 237
- m\_lash\_args
  - seq64::lash, 389
- m\_lash\_support
  - seq64::rc\_settings, 813
- m\_last\_playing
  - seq64::mainwid, 413
- m\_last\_tick
  - seq64::sequence, 1050
- m\_last\_tick\_x
  - seq64::mainwid, 413
- m\_last\_used\_dir
  - seq64::rc\_settings, 816
- m\_lasttick
  - seq64::midibase, 603
- m\_left\_marker\_tick
  - seq64::perftime, 793
- m\_left\_tick
  - seq64::perform, 757
- m\_legacy\_format
  - seq64::rc\_settings, 813
- m\_length
  - seq64::sequence, 1051
  - seq64::triggers, 1073
- m\_lfo\_wnd
  - seq64::seqedit, 898
- m\_line
  - seq64::configfile, 165
- m\_line\_color
  - seq64::gui\_palette\_gtk2, 309
- m\_line\_count
  - seq64::eventslots, 256
- m\_line\_maximum
  - seq64::eventslots, 256
- m\_line\_overlap
  - seq64::eventslots, 257
- m\_linked
  - seq64::event, 208
- m\_local\_addr\_client
  - seq64::midi\_alsa, 475
  - seq64::midibus, 616
- m\_local\_addr\_port
  - seq64::midi\_alsa, 475
  - seq64::midibus, 616
- m\_looping
  - seq64::perform, 756
- m\_lt\_grey
  - seq64::gui\_palette\_gtk2, 308
- m\_main\_cursor
  - seq64::mainwnd, 432
- m\_main\_time
  - seq64::mainwnd, 431
- m\_main\_wid
  - seq64::mainwnd, 431
- m\_mainperf
  - seq64::gui\_drawingarea\_gtk2, 297
  - seq64::gui\_window\_gtk2, 314
  - seq64::options, 644
  - seq64::seqmenu, 942
- m\_mainwid\_border
  - seq64::mainwid, 414
  - seq64::user\_settings, 1111
- m\_mainwid\_spacing
  - seq64::mainwid, 414
  - seq64::user\_settings, 1111
- m\_mainwid\_x

- seq64::mainwid, [414](#)
- seq64::user\_settings, [1118](#)
- m\_mainwid\_y
  - seq64::mainwid, [414](#)
  - seq64::user\_settings, [1118](#)
- m\_mainwnd\_cols
  - seq64::mainwid, [413](#)
  - seq64::user\_settings, [1111](#)
- m\_mainwnd\_rows
  - seq64::mainwid, [413](#)
  - seq64::user\_settings, [1111](#)
- m\_manual\_alsa\_ports
  - seq64::rc\_settings, [815](#)
- m\_marked
  - seq64::event, [209](#)
- m\_master\_bus
  - seq64::perform, [757](#)
- m\_master\_clocks
  - seq64::mastermidibase, [454](#)
  - seq64::perform, [757](#)
- m\_master\_info
  - seq64::midi\_api, [490](#)
  - seq64::midibus, [616](#)
- m\_master\_inputs
  - seq64::mastermidibase, [454](#)
  - seq64::perform, [757](#)
- m\_masterbus
  - seq64::sequence, [1048](#)
- m\_max\_busses
  - seq64::mastermidibase, [454](#)
- m\_max\_sequence
  - seq64::user\_settings, [1117](#)
- m\_max\_sets
  - seq64::mainwid, [414](#)
  - seq64::perform, [760](#)
  - seq64::user\_settings, [1111](#)
- m\_max\_value
  - seq64::midi\_control, [506](#)
- m\_maxbeats
  - seq64::sequence, [1050](#)
- m\_measure\_length
  - seq64::perfroll, [781](#)
  - seq64::perftime, [793](#)
- m\_measures
  - seq64::midi\_measures, [552](#)
  - seq64::seqedit, [893](#)
- m\_menu
  - seq64::seqmenu, [942](#)
- m\_menu\_bpm
  - seq64::perfedit, [667](#)
  - seq64::seqedit, [896](#)
- m\_menu\_bw
  - seq64::perfedit, [667](#)
  - seq64::seqedit, [896](#)
- m\_menu\_chords
  - seq64::seqedit, [896](#)
- m\_menu\_data
  - seq64::seqedit, [896](#)
- m\_menu\_edit
  - seq64::mainwnd, [430](#)
- m\_menu\_file
  - seq64::mainwnd, [430](#)
- m\_menu\_help
  - seq64::mainwnd, [431](#)
- m\_menu\_key
  - seq64::seqedit, [896](#)
- m\_menu\_length
  - seq64::seqedit, [895](#)
- m\_menu\_midibus
  - seq64::seqedit, [895](#)
- m\_menu\_midich
  - seq64::seqedit, [895](#)
- m\_menu\_mode
  - seq64::mainwnd, [434](#)
- m\_menu\_note\_length
  - seq64::seqedit, [895](#)
- m\_menu\_rec\_vol
  - seq64::seqedit, [896](#)
- m\_menu\_scale
  - seq64::seqedit, [896](#)
- m\_menu\_sequences
  - seq64::seqedit, [896](#)
- m\_menu\_snap
  - seq64::perfedit, [663](#)
  - seq64::seqedit, [895](#)
- m\_menu\_tools
  - seq64::seqedit, [894](#)
- m\_menu\_view
  - seq64::mainwnd, [431](#)
- m\_menu\_xpose
  - seq64::perfedit, [664](#)
- m\_menu\_zoom
  - seq64::seqedit, [895](#)
- m\_menubar
  - seq64::mainwnd, [430](#)
  - seq64::seqedit, [894](#)
- m\_message
  - seq64::rterror, [821](#)
  - seq64::rtmidi\_in\_data, [838](#)
- m\_midi\_api
  - seq64::rtmidi, [829](#)
- m\_midi\_beat\_width
  - seq64::user\_settings, [1115](#)
- m\_midi\_beats\_per\_measure
  - seq64::user\_settings, [1115](#)
- m\_midi\_beats\_per\_minute
  - seq64::user\_settings, [1115](#)
- m\_midi\_bus\_def
  - seq64::user\_midi\_bus, [1083](#)
- m\_midi\_buses
  - seq64::user\_settings, [1110](#)
- m\_midi\_buss\_override
  - seq64::user\_settings, [1115](#)
- m\_midi\_cc\_off
  - seq64::perform, [759](#)
- m\_midi\_cc\_on

- seq64::perform, 759
- m\_midi\_cc\_toggle
  - seq64::perform, 759
- m\_midi\_channel
  - seq64::sequence, 1047
- m\_midi\_handle
  - seq64::midi\_info, 522
- m\_midi\_info
  - seq64::rtmidi, 829
- m\_midi\_master
  - seq64::mastermidibus, 465
- m\_midi\_mode\_input
  - seq64::midi\_info, 521
- m\_midi\_parameters
  - seq64::editable\_events, 187
- m\_midi\_ppqn
  - seq64::user\_settings, 1115
- m\_midiclockpos
  - seq64::perform, 759
- m\_midiclockrunning
  - seq64::perform, 758
- m\_midiclocktick
  - seq64::perform, 758
- m\_min\_value
  - seq64::midi\_control, 505
- m\_mode\_group
  - seq64::perform, 753
- m\_mode\_group\_learn
  - seq64::perform, 753
- m\_modified
  - seq64::seqmenu, 943
- m\_modifier
  - seq64::click, 159
  - seq64::keystroke, 386
- m\_move\_delta\_x
  - seq64::seqroll, 974
- m\_move\_delta\_y
  - seq64::seqroll, 974
- m\_move\_snap\_offset\_x
  - seq64::seqevent, 918
  - seq64::seqroll, 974
- m\_moving
  - seq64::mainwid, 412
  - seq64::perfroll, 783
  - seq64::seqevent, 917
  - seq64::seqroll, 973
- m\_moving\_init
  - seq64::seqevent, 917
  - seq64::seqroll, 973
- m\_moving\_seq
  - seq64::mainwid, 412
- m\_multi\_client
  - seq64::midi\_jack, 536
  - seq64::midi\_jack\_info, 546
- m\_musical\_key
  - seq64::sequence, 1052
- m\_musical\_scale
  - seq64::sequence, 1052
- m\_mute\_group
  - seq64::perform, 752
- m\_mute\_group\_selected
  - seq64::perform, 753
- m\_mutex
  - seq64::mastermidibase, 455
  - seq64::midibase, 604
  - seq64::midifile, 635
  - seq64::sequence, 1052
- m\_mutex\_lock
  - seq64::mutex, 638
- m\_name
  - seq64::configfile, 165
  - seq64::midifile, 635
  - seq64::sequence, 1050
- m\_name\_category
  - seq64::editable\_event, 178
- m\_name\_channel
  - seq64::editable\_event, 179
- m\_name\_data
  - seq64::editable\_event, 179
- m\_name\_meta
  - seq64::editable\_event, 178
- m\_name\_seqspect
  - seq64::editable\_event, 179
- m\_name\_status
  - seq64::editable\_event, 178
- m\_name\_timestamp
  - seq64::editable\_event, 178
- m\_namebox\_w
  - seq64::perfnames, 675
- m\_names\_chars
  - seq64::perfnames, 675
- m\_names\_x
  - seq64::perfnames, 675
- m\_names\_y
  - seq64::perfnames, 675
  - seq64::perfroll, 780
- m\_new\_format
  - seq64::midifile, 636
- m\_note\_length
  - seq64::seqedit, 893
  - seq64::seqroll, 972
- m\_note\_off\_margin
  - seq64::sequence, 1053
- m\_note\_off\_velocity
  - seq64::sequence, 1052
- m\_note\_on\_velocity
  - seq64::sequence, 1052
- m\_notebook
  - seq64::options, 645
- m\_notes\_on
  - seq64::sequence, 1048
- m\_notify
  - seq64::perform, 762
- m\_num\_poll\_descriptors
  - seq64::mastermidibus, 465
  - seq64::midi\_alsa\_info, 481



- m\_number\_h
  - seq64::seqdata, 870
- m\_number\_offset\_y
  - seq64::seqdata, 870
- m\_number\_w
  - seq64::seqdata, 870
- m\_numbers
  - seq64::seqdata, 870
- m\_offset
  - seq64::font, 262
  - seq64::perform, 759
  - seq64::trigger, 1057
- m\_old
  - seq64::seqdata, 870
  - seq64::seqevent, 916
  - seq64::seqroll, 971
- m\_old\_progress\_ticks
  - seq64::perfroll, 781
- m\_old\_seq
  - seq64::mainwid, 412
- m\_one\_measure
  - seq64::perform, 757
- m\_options
  - seq64::mainwnd, 432
- m\_optsbox
  - seq64::eventedit, 236
- m\_orange
  - seq64::gui\_palette\_gtk2, 307
- m\_out\_thread
  - seq64::perform, 754
- m\_out\_thread\_launched
  - seq64::perform, 755
- m\_outbus\_array
  - seq64::mastermidibase, 454
- m\_output
  - seq64::midi\_info, 521
- m\_outputing
  - seq64::perform, 755
- m\_padded\_h
  - seq64::font, 262
- m\_page\_factor
  - seq64::perfroll, 779
- m\_pager\_index
  - seq64::eventslots, 257
- m\_painted
  - seq64::event, 209
- m\_painting
  - seq64::seqevent, 917
  - seq64::seqroll, 973
- m\_parent
  - seq64::editable\_event, 178
  - seq64::eventslots, 255
  - seq64::perfnames, 674
  - seq64::perfroll, 779
  - seq64::perftime, 792
  - seq64::sequence, 1046
  - seq64::triggers, 1072
- m\_parent\_bus
  - seq64::midi\_api, 490
- m\_pass\_sysex
  - seq64::rc\_settings, 814
- m\_paste
  - seq64::seqevent, 918
  - seq64::seqroll, 974
- m\_paste\_tick
  - seq64::triggers, 1073
- m\_peer\_perfedit
  - seq64::perfedit, 662
- m\_perf
  - seq64::keybindentry, 344
- m\_perf\_edit
  - seq64::mainwnd, 431
- m\_perf\_edit\_2
  - seq64::mainwnd, 431
- m\_perf\_scale\_x
  - seq64::perfroll, 780
  - seq64::perftime, 793
- m\_perfnames
  - seq64::perfedit, 663
- m\_perform
  - seq64::lash, 389
- m\_perfroll
  - seq64::perfedit, 663
- m\_perftime
  - seq64::perfedit, 663
- m\_phase
  - seq64::lfownd, 394
- m\_pill\_width
  - seq64::maintime, 399
- m\_pixmap
  - seq64::font, 262
  - seq64::gui\_drawingarea\_gtk2, 296
- m\_playback\_mode
  - seq64::perform, 756
- m\_playing
  - seq64::sequence, 1048
- m\_playing\_notes
  - seq64::sequence, 1048
- m\_playing\_screen
  - seq64::perform, 753
- m\_playscreen\_offset
  - seq64::perform, 753
- m\_poll\_descriptors
  - seq64::mastermidibus, 465
  - seq64::midi\_alsa\_info, 481
- m\_port\_container
  - seq64::midi\_port\_info, 564
- m\_port\_count
  - seq64::midi\_port\_info, 564
- m\_port\_id
  - seq64::midibase, 602
- m\_port\_name
  - seq64::midi\_port\_info::port\_info\_t, 794
  - seq64::midibase, 603
- m\_port\_number
  - seq64::midi\_port\_info::port\_info\_t, 794

- m\_pos
  - seq64::midifile, 635
  - seq64::seqroll, 972
- m\_position\_for\_get
  - seq64::midi\_container, 501
- m\_pp\_eighth
  - seq64::seqedit, 894
- m\_pp\_sixteenth
  - seq64::seqedit, 894
- m\_pp\_whole
  - seq64::seqedit, 894
- m\_ppqn
  - seq64::jack\_assistant, 337
  - seq64::maintime, 400
  - seq64::mainwnd, 431
  - seq64::mastermidibase, 455
  - seq64::midi\_info, 522
  - seq64::midi\_splitter, 573
  - seq64::midi\_timing, 577
  - seq64::midibase, 602
  - seq64::midifile, 636
  - seq64::perfedit, 667
  - seq64::perform, 756
  - seq64::perfroll, 779
  - seq64::perftime, 792
  - seq64::seqedit, 894
  - seq64::seqevent, 916
  - seq64::seqroll, 972
  - seq64::seqtime, 982
  - seq64::sequence, 1050
  - seq64::triggers, 1073
- m\_print\_keys
  - seq64::rc\_settings, 815
- m\_priority
  - seq64::rc\_settings, 814
- m\_progress\_bar\_colored
  - seq64::user\_settings, 1113
- m\_progress\_bar\_thick
  - seq64::user\_settings, 1113
- m\_progress\_color
  - seq64::gui\_palette\_gtk2, 309
- m\_progress\_height
  - seq64::mainwid, 415
- m\_progress\_x
  - seq64::seqroll, 974
- m\_quantized\_rec
  - seq64::sequence, 1049
- m\_queue
  - seq64::mastermidibase, 454
  - seq64::midibase, 602
  - seq64::rtmidi\_in\_data, 838
- m\_queue\_number
  - seq64::midi\_port\_info::port\_info\_t, 795
- m\_queued
  - seq64::sequence, 1049
- m\_queued\_tick
  - seq64::sequence, 1050
- m\_raise
  - seq64::sequence, 1050
- m\_range
  - seq64::lfownd, 394
- m\_rank
  - seq64::event\_list::event\_key, 211
- m\_rec\_vol
  - seq64::sequence, 1052
- m\_recording
  - seq64::sequence, 1048
- m\_red
  - seq64::gui\_palette\_gtk2, 307
- m\_redo\_stack
  - seq64::triggers, 1073
- m\_redo\_vect
  - seq64::perform, 761
- m\_redraw\_period\_ms
  - seq64::gui\_window\_gtk2, 314
- m\_remote\_port\_name
  - seq64::midi\_jack, 536
- m\_reposition
  - seq64::perform, 752
- m\_reveal\_alsa\_ports
  - seq64::rc\_settings, 815
- m\_right\_marker\_tick
  - seq64::perftime, 793
- m\_right\_tick
  - seq64::perform, 758
- m\_rightbox
  - seq64::eventedit, 236
- m\_ring
  - seq64::midi\_queue, 568
- m\_ring\_size
  - seq64::midi\_queue, 568
- m\_roll\_length\_ticks
  - seq64::perfroll, 782
- m\_rt\_midi
  - seq64::midibus, 616
- m\_running
  - seq64::perform, 755
- m\_safety\_mutex
  - seq64::automutex, 138
- m\_save\_user\_config
  - seq64::user\_settings, 1118
- m\_scale
  - seq64::seqedit, 893
  - seq64::seqkeys, 929
  - seq64::seqroll, 972
- m\_scale\_phase
  - seq64::lfownd, 393
- m\_scale\_range
  - seq64::lfownd, 393
- m\_scale\_speed
  - seq64::lfownd, 393
- m\_scale\_value
  - seq64::lfownd, 393
- m\_scale\_wave
  - seq64::lfownd, 393
- m\_screen\_set\_notepad

- seq64::perform, 759
- m\_screenset
  - seq64::mainwid, 413
  - seq64::perform, 760
- m\_screenset\_offset
  - seq64::mainwid, 415
- m\_screenset\_slots
  - seq64::mainwid, 415
- m\_scroll\_offset\_key
  - seq64::seqkeys, 928
  - seq64::seqroll, 975
- m\_scroll\_offset\_ticks
  - seq64::seqdata, 869
  - seq64::seqevent, 917
  - seq64::seqroll, 975
  - seq64::seqtime, 982
- m\_scroll\_offset\_x
  - seq64::seqdata, 870
  - seq64::seqevent, 917
  - seq64::seqroll, 975
  - seq64::seqtime, 982
- m\_scroll\_offset\_y
  - seq64::seqkeys, 929
  - seq64::seqroll, 975
- m\_selected
  - seq64::event, 209
  - seq64::seqevent, 916
  - seq64::seqroll, 971
  - seq64::trigger, 1057
- m\_selecting
  - seq64::seqevent, 917
  - seq64::seqroll, 973
- m\_seq
  - seq64::eventedit, 239
  - seq64::eventslots, 255
  - seq64::lfownd, 393
  - seq64::mastermidibase, 455
  - seq64::midi\_alsa, 475
  - seq64::midibus, 615
  - seq64::seqdata, 869
  - seq64::seqedit, 894
  - seq64::seqevent, 916
  - seq64::seqkeys, 928
  - seq64::seqroll, 971
  - seq64::seqtime, 982
- m\_seq24\_interaction
  - seq64::perfroll, 783
  - seq64::seqevent, 916
- m\_seq\_number
  - seq64::sequence, 1050
- m\_seqarea\_seq\_x
  - seq64::mainwid, 413
  - seq64::user\_settings, 1117
- m\_seqarea\_seq\_y
  - seq64::mainwid, 414
  - seq64::user\_settings, 1117
- m\_seqarea\_x
  - seq64::mainwid, 413
- seq64::user\_settings, 1117
- m\_seqarea\_y
  - seq64::mainwid, 413
  - seq64::user\_settings, 1117
- m\_seqchars\_x
  - seq64::user\_settings, 1114
- m\_seqchars\_y
  - seq64::user\_settings, 1114
- m\_seqdata
  - seq64::lfownd, 393
- m\_seqdata\_wid
  - seq64::seqedit, 897
  - seq64::seqevent, 917
- m\_seqedit
  - seq64::seqmenu, 942
- m\_seqedit\_bgsequence
  - seq64::user\_settings, 1112
- m\_seqedit\_key
  - seq64::user\_settings, 1112
- m\_seqedit\_scale
  - seq64::user\_settings, 1112
- m\_seqevent\_wid
  - seq64::seqedit, 897
- m\_seqkeys\_wid
  - seq64::seqedit, 897
  - seq64::seqroll, 972
- m\_seqroll\_wid
  - seq64::seqedit, 898
- m\_seqs
  - seq64::perform, 753
- m\_seqs\_active
  - seq64::perform, 753
- m\_seqs\_in\_set
  - seq64::perfnames, 676
  - seq64::perform, 760
  - seq64::user\_settings, 1116
- m\_seqtime\_wid
  - seq64::seqedit, 897
- m\_sequence
  - seq64::editable\_events, 187
  - seq64::midi\_container, 501
- m\_sequence\_active
  - seq64::perfnames, 676
  - seq64::perfroll, 782
- m\_sequence\_count
  - seq64::perform, 760
- m\_sequence\_high
  - seq64::perform, 760
- m\_sequence\_max
  - seq64::perfnames, 676
  - seq64::perform, 760
  - seq64::perfroll, 782
- m\_sequence\_offset
  - seq64::perfnames, 676
  - seq64::perfroll, 782
- m\_sequence\_state
  - seq64::perform, 754
- m\_setbox\_w

- seq64::eventslots, 256
- seq64::perfnames, 675
- m\_show\_midi
  - seq64::rc\_settings, 814
- m\_show\_octave\_letters
  - seq64::seqkeys, 929
- m\_showbox
  - seq64::eventedit, 236
- m\_sigpipe
  - seq64::mainwnd, 430
- m\_size
  - seq64::midi\_queue, 568
- m\_size\_box\_w
  - seq64::perroll, 780
- m\_slot
  - seq64::keybindentry, 344
- m\_slots\_chars
  - seq64::eventslots, 255
- m\_slots\_x
  - seq64::eventslots, 256
- m\_slots\_y
  - seq64::eventslots, 256
- m\_smf0\_channels
  - seq64::midi\_splitter, 574
- m\_smf0\_channels\_count
  - seq64::midi\_splitter, 573
- m\_smf0\_main\_sequence
  - seq64::midi\_splitter, 574
- m\_smf0\_seq\_number
  - seq64::midi\_splitter, 574
- m\_smf0\_splitter
  - seq64::midifile, 636
- m\_snap
  - seq64::perfedit, 667
  - seq64::perroll, 779
  - seq64::perftime, 793
  - seq64::seqedit, 893
  - seq64::seqevent, 916
  - seq64::seqroll, 972
- m\_snap\_tick
  - seq64::sequence, 1051
- m\_song\_mute
  - seq64::sequence, 1048
- m\_song\_start\_mode
  - seq64::perform, 751
- m\_speed
  - seq64::lfownd, 394
- m\_spinbutton\_bpm
  - seq64::mainwnd, 433
- m\_spinbutton\_load\_offset
  - seq64::mainwnd, 433
- m\_spinbutton\_ss
  - seq64::mainwnd, 433
- m\_standard\_bpm
  - seq64::perfedit, 667
- m\_start\_from\_perfedit
  - seq64::perform, 751
- m\_starting\_tick
  - seq64::perform, 758
- m\_stats
  - seq64::rc\_settings, 814
- m\_status
  - seq64::event, 208
  - seq64::midi\_control, 505
  - seq64::seqdata, 870
  - seq64::seqevent, 918
  - seq64::seqroll, 976
- m\_sysex
  - seq64::event, 208
- m\_sysex\_size
  - seq64::event, 208
- m\_table
  - seq64::eventedit, 235
  - seq64::perfedit, 662
  - seq64::seqedit, 898
- m\_text\_size\_x
  - seq64::mainwid, 414
- m\_text\_size\_y
  - seq64::mainwid, 414
- m\_text\_x
  - seq64::user\_settings, 1114
- m\_text\_y
  - seq64::user\_settings, 1114
- m\_thru
  - seq64::sequence, 1049
- m\_tick
  - seq64::maintime, 400
  - seq64::perform, 758
- m\_tick\_end
  - seq64::trigger, 1057
- m\_tick\_offset
  - seq64::perftime, 792
- m\_tick\_start
  - seq64::trigger, 1057
- m\_ticks\_per\_bar
  - seq64::perroll, 780
- m\_time\_beat\_width
  - seq64::sequence, 1051
- m\_time\_beats\_per\_measure
  - seq64::sequence, 1051
- m\_timearea\_y
  - seq64::perftime, 793
- m\_timeout\_connect
  - seq64::mainwnd, 434
- m\_timestamp
  - seq64::event, 207
  - seq64::event\_list::event\_key, 211
  - seq64::midi\_message, 556
- m\_toggle\_jack
  - seq64::jack\_assistant, 337
- m\_toggle\_play
  - seq64::seqedit, 902
- m\_toggle\_q\_rec
  - seq64::seqedit, 903
- m\_toggle\_record
  - seq64::seqedit, 903

- m\_toggle\_thru
  - seq64::seqedit, [903](#)
- m\_toggle\_transpose
  - seq64::seqedit, [895](#)
- m\_tooltips
  - seq64::mainwnd, [430](#)
  - seq64::options, [644](#)
  - seq64::perfedit, [666](#)
  - seq64::seqedit, [901](#)
- m\_top\_index
  - seq64::eventslots, [257](#)
- m\_top\_iterator
  - seq64::eventslots, [257](#)
- m\_total\_seqs
  - seq64::user\_settings, [1116](#)
- m\_tracks\_mute\_state
  - seq64::perform, [752](#)
- m\_trans\_button\_press
  - seq64::perfroll, [781](#)
  - seq64::seqroll, [975](#)
- m\_transport\_follow
  - seq64::perfroll, [781](#)
  - seq64::seqroll, [975](#)
- m\_transposable
  - seq64::sequence, [1048](#)
- m\_transpose
  - seq64::perform, [754](#)
- m\_trigger\_copied
  - seq64::triggers, [1073](#)
- m\_trigger\_offset
  - seq64::sequence, [1050](#)
- m\_triggers
  - seq64::sequence, [1046](#)
  - seq64::triggers, [1072](#)
- m\_type
  - seq64::keybindentry, [343](#)
  - seq64::rterror, [821](#)
- m\_undo\_stack
  - seq64::triggers, [1073](#)
- m\_undo\_vect
  - seq64::perform, [761](#)
- m\_us\_per\_quarter\_note
  - seq64::perform, [757](#)
  - seq64::sequence, [1051](#)
- m\_use\_default\_ppqn
  - seq64::midi\_splitter, [573](#)
  - seq64::midifile, [636](#)
- m\_use\_jack\_polling
  - seq64::mastermidibus, [466](#)
- m\_use\_more\_icons
  - seq64::user\_settings, [1114](#)
- m\_use\_new\_font
  - seq64::font, [261](#)
  - seq64::user\_settings, [1112](#)
- m\_usemidiclock
  - seq64::perform, [758](#)
- m\_user\_callback
  - seq64::rtmidi\_in\_data, [839](#)
- m\_user\_data
  - seq64::rtmidi\_in\_data, [839](#)
- m\_user\_filename
  - seq64::rc\_settings, [816](#)
- m\_user\_filename\_alt
  - seq64::rc\_settings, [816](#)
- m\_using\_callback
  - seq64::rtmidi\_in\_data, [839](#)
- m\_v\_page\_increment
  - seq64::perfroll, [779](#)
- m\_v\_perf\_page\_increment
  - seq64::user\_settings, [1113](#)
- m\_vadjust
  - seq64::eventedit, [235](#)
  - seq64::gui\_drawingarea\_gtk2, [296](#)
  - seq64::perfedit, [663](#)
  - seq64::seqedit, [897](#)
- m\_value
  - seq64::lfownd, [394](#)
- m\_vbox
  - seq64::seqedit, [898](#)
- m\_vector\_sequence
  - seq64::mastermidibase, [455](#)
- m\_velocity\_override
  - seq64::user\_settings, [1115](#)
- m\_vertical\_adjust
  - seq64::seqroll, [971](#)
- m\_vscroll
  - seq64::eventedit, [235](#)
  - seq64::perfedit, [663](#)
- m\_vscroll\_new
  - seq64::seqedit, [897](#)
- m\_was\_active\_edit
  - seq64::perform, [754](#)
- m\_was\_active\_main
  - seq64::perform, [754](#)
- m\_was\_active\_names
  - seq64::perform, [754](#)
- m\_was\_active\_perf
  - seq64::perform, [754](#)
- m\_was\_playing
  - seq64::sequence, [1048](#)
- m\_wave
  - seq64::lfownd, [394](#)
- m\_wave\_name
  - seq64::lfownd, [394](#)
- m\_white
  - seq64::gui\_palette\_gtk2, [307](#)
- m\_white\_pixmap
  - seq64::font, [262](#)
- m\_wht\_key
  - seq64::gui\_palette\_gtk2, [309](#)
- m\_wht\_paint
  - seq64::gui\_palette\_gtk2, [308](#)
- m\_window
  - seq64::gui\_drawingarea\_gtk2, [296](#)
- m\_window\_redraw\_rate\_ms
  - seq64::user\_settings, [1113](#)

- m\_window\_x
  - seq64::gui\_drawingarea\_gtk2, [297](#)
  - seq64::gui\_window\_gtk2, [314](#)
- m\_window\_y
  - seq64::gui\_drawingarea\_gtk2, [297](#)
  - seq64::gui\_window\_gtk2, [314](#)
- m\_with\_jack\_master
  - seq64::rc\_settings, [814](#)
- m\_with\_jack\_master\_cond
  - seq64::rc\_settings, [814](#)
- m\_with\_jack\_midi
  - seq64::rc\_settings, [815](#)
- m\_with\_jack\_transport
  - seq64::rc\_settings, [814](#)
- m\_x
  - seq64::click, [158](#)
- m\_xy\_offset
  - seq64::perfnames, [675](#)
- m\_y
  - seq64::click, [159](#)
- m\_y\_on\_b\_pixmap
  - seq64::font, [262](#)
- m\_yellow
  - seq64::gui\_palette\_gtk2, [307](#)
- m\_zoom
  - seq64::perfroll, [780](#)
  - seq64::seqdata, [869](#)
  - seq64::seqedit, [893](#)
  - seq64::seqevent, [916](#)
  - seq64::seqroll, [972](#)
  - seq64::seqtime, [982](#)
- maintime
  - seq64::maintime, [397](#)
- mainwid
  - seq64::mainwid, [404](#)
- mainwid\_border
  - seq64::user\_settings, [1099](#), [1105](#)
- mainwid\_grid\_style\_t
  - seq64::user\_settings, [1093](#)
- mainwid\_spacing
  - seq64::user\_settings, [1099](#), [1105](#)
- mainwid\_x
  - seq64::user\_settings, [1099](#)
- mainwid\_y
  - seq64::user\_settings, [1100](#)
- mainwnd
  - seq64::maintime, [398](#)
  - seq64::mainwid, [412](#)
  - seq64::mainwnd, [420](#)
  - seq64::perform, [747](#)
  - seq64::rc\_settings, [812](#)
  - seq64::seqmenu, [941](#)
- mainwnd\_cols
  - seq64::user\_settings, [1098](#), [1103](#)
- mainwnd\_key\_event
  - seq64::perform, [722](#)
- mainwnd\_rows
  - seq64::user\_settings, [1097](#), [1103](#)
- make\_clock
  - seq64::event, [205](#)
- make\_directory
  - seq64, [91](#)
- make\_section\_name
  - seq64, [101](#)
- manual\_alsa\_ports
  - seq64::rc\_settings, [804](#), [808](#)
- mark
  - seq64::event, [204](#)
- mark\_all
  - seq64::event\_list, [220](#)
- mark\_out\_of\_range
  - seq64::event\_list, [219](#)
- mark\_selected
  - seq64::event\_list, [219](#)
  - seq64::sequence, [1034](#)
- master\_bus
  - seq64::perform, [695](#)
- master\_info
  - seq64::midi\_api, [489](#)
- master\_midi\_mode
  - seq64::midi\_api, [489](#)
- mastermidibase
  - seq64::mastermidibase, [439](#)
- mastermidibus
  - seq64::mastermidibus, [459](#)
  - seq64::midibase, [601](#)
  - seq64::midibus, [615](#)
  - seq64::rtmidi\_info, [848](#)
- match
  - seq64::midi\_control, [504](#)
  - seq64::midibase, [590](#)
- max\_active\_set
  - seq64::perform, [739](#)
- max\_sequence
  - seq64::user\_settings, [1098](#)
- max\_sets
  - seq64::user\_settings, [1098](#), [1103](#)
- max\_value
  - seq64::midi\_control, [504](#)
- max\_zoom
  - seq64::user\_settings, [1106](#)
- mc\_baseline\_ppqn
  - seq64::user\_settings, [1118](#)
- mc\_max\_zoom
  - seq64::user\_settings, [1118](#)
- mc\_min\_zoom
  - seq64::user\_settings, [1118](#)
- measures
  - seq64::midi\_measures, [551](#)
- measures\_to\_ticks
  - seq64, [83](#)
  - seq64::sequence, [1000](#)
- measurestring\_to\_pulses
  - seq64, [74](#)
- menu\_mode
  - seq64::keys\_perform, [359](#)

- merge
  - seq64::event\_list, [217](#)
- message
  - seq64::rtmidi\_in\_data, [835](#)
- message\_concatenate
  - seq64, [88](#)
- meta\_string
  - seq64::editable\_event, [176](#)
- midi\_alsa
  - seq64::midi\_alsa, [469](#)
- midi\_alsa\_info
  - seq64::mastermidibase, [453](#)
  - seq64::mastermidibus, [465](#)
  - seq64::midi\_alsa\_info, [477](#)
- midi\_api
  - seq64::midi\_api, [484](#)
- midi\_api\_name
  - seq64, [106](#)
- midi\_beat\_width
  - seq64::user\_settings, [1106](#), [1109](#)
- midi\_beats\_per\_bar
  - seq64::user\_settings, [1105](#), [1109](#)
- midi\_beats\_per\_minute
  - seq64::user\_settings, [1105](#), [1109](#)
- midi\_buss\_override
  - seq64::user\_settings, [1106](#), [1108](#)
- midi\_container
  - seq64::event\_list, [222](#)
  - seq64::midi\_container, [494](#)
  - seq64::triggers, [1071](#)
- midi\_control
  - seq64::midi\_control, [503](#)
- midi\_control\_event
  - seq64::perform, [730](#)
- midi\_control\_off
  - seq64::perform, [729](#)
- midi\_control\_on
  - seq64::perform, [729](#)
- midi\_control\_toggle
  - seq64::perform, [728](#)
- midi\_handle
  - seq64::midi\_info, [515](#), [520](#)
- midi\_in\_alsa
  - seq64::midi\_in\_alsa, [508](#)
- midi\_in\_jack
  - seq64::midi\_in\_jack, [510](#)
- midi\_info
  - seq64::midi\_info, [515](#)
- midi\_input\_callback
  - seq64, [109](#)
- midi\_input\_test
  - seq64, [107](#)
- midi\_jack
  - seq64::midi\_jack, [525](#)
  - seq64::midi\_jack\_info, [545](#)
- midi\_jack\_data
  - seq64::midi\_jack\_data, [537](#)
- midi\_jack\_info
  - seq64::mastermidibus, [465](#)
  - seq64::midi\_jack, [535](#)
  - seq64::midi\_jack\_info, [541](#)
- midi\_list
  - seq64::midi\_list, [548](#)
- midi\_measures
  - seq64::midi\_measures, [551](#)
- midi\_measures\_to\_pulses
  - seq64, [74](#)
- midi\_message
  - seq64::midi\_message, [554](#)
- midi\_mode
  - seq64::midi\_info, [515](#)
  - seq64::rtmidi\_info, [842](#), [843](#)
- midi\_out\_alsa
  - seq64::midi\_out\_alsa, [558](#)
- midi\_out\_jack
  - seq64::midi\_out\_jack, [560](#)
- midi\_port\_info
  - seq64::midi\_port\_info, [562](#)
- midi\_ppqn
  - seq64::user\_settings, [1105](#), [1108](#)
- midi\_probe
  - seq64, [106](#)
- midi\_queue
  - seq64::midi\_queue, [566](#)
- midi\_splitter
  - seq64::event\_list, [222](#)
  - seq64::midi\_splitter, [570](#)
- midi\_timing
  - seq64::midi\_timing, [575](#)
- midi\_vector
  - seq64::midi\_vector, [580](#)
- midibase
  - seq64::midibase, [587](#)
- midibpm
  - seq64, [65](#)
- midibus
  - seq64::midibus, [607](#), [608](#)
  - seq64::rtmidi, [829](#)
  - seq64::rtmidi\_info, [848](#)
- midibyte
  - seq64, [64](#)
- midifile
  - seq64::event\_list, [222](#)
  - seq64::midi\_container, [500](#)
  - seq64::midifile, [619](#)
  - seq64::perform, [747](#)
  - seq64::triggers, [1072](#)
- midilong
  - seq64, [65](#)
- midipulse
  - seq64, [65](#)
- midishort
  - seq64, [64](#)
- millisleep
  - seq64, [98](#)
- min

- seq64, 107
- min\_value
  - seq64::midi\_control, 504
- min\_zoom
  - seq64::user\_settings, 1106
- mod\_control
  - seq64::click, 158
  - seq64::keystroke, 385
- mod\_control\_shift
  - seq64::click, 158
  - seq64::keystroke, 385
- mod\_last\_tick
  - seq64::sequence, 1004
- mod\_super
  - seq64::click, 158
  - seq64::keystroke, 385
- mod\_timestamp
  - seq64::event, 198
- modifier
  - seq64::click, 158
  - seq64::keystroke, 385
- modify
  - seq64::perform, 692
  - seq64::sequence, 996
- modify\_current\_event
  - seq64::eventslots, 247
- motion\_notify
  - seq64::seqroll, 963
- mouse\_action
  - seq64::seqedit, 889
- mouse\_action\_e
  - seq64, 69
- mouse\_click\_edit\_callback
  - seq64::options, 643
- mouse\_fruity\_callback
  - seq64::options, 643
- mouse\_mod4\_callback
  - seq64::options, 643
- mouse\_seq24\_callback
  - seq64::options, 643
- mouse\_snap\_split\_callback
  - seq64::options, 643
- move
  - seq64::triggers, 1068
- move\_selected
  - seq64::triggers, 1067
- move\_selected\_notes
  - seq64::seqroll, 960
  - seq64::sequence, 1029
- move\_selected\_triggers\_to
  - seq64::sequence, 1018
- move\_selection\_box
  - seq64::seqroll, 960
- move\_triggers
  - seq64::perform, 723
  - seq64::sequence, 1019
- moving
  - seq64::seqroll, 966
- multi\_client
  - seq64::midi\_jack, 527
  - seq64::midi\_jack\_info, 541
- multiply\_pattern
  - seq64::sequence, 1041
- musical\_key
  - seq64::sequence, 1041
- musical\_scale
  - seq64::sequence, 1041
- mute\_all\_tracks
  - seq64::perform, 709
  - seq64::seqmenu, 941
- mute\_group\_offset
  - seq64::perform, 742
- mute\_group\_tracks
  - seq64::perform, 732
- mute\_op\_t
  - seq64::perform, 690
- mute\_screenset
  - seq64::perform, 710
- mutex
  - seq64::mutex, 638
- name
  - seq64::sequence, 1002
  - seq64::user\_instrument, 1076
  - seq64::user\_midi\_bus, 1081
- name\_change\_callback
  - seq64::seqedit, 885
- name\_to\_value
  - seq64::editable\_event, 172
- nc\_midi\_port\_info
  - seq64::midi\_info, 520
- new\_current\_sequence
  - seq64::seqmenu, 936
- new\_file
  - seq64::mainwnd, 428
- new\_open\_error\_dialog
  - seq64::mainwnd, 427
- new\_sequence
  - seq64::perform, 701
  - seq64::seqmenu, 937
- next
  - seq64::triggers, 1070
- next\_data\_line
  - seq64::configfile, 163
- next\_trigger
  - seq64::triggers, 1070
- non\_cc\_match
  - seq64::event, 200
- normal\_action
  - seq64::seqroll, 966
- normalize
  - seq64::user\_settings, 1094
- note\_off\_length
  - seq64::seqroll, 951
- note\_off\_margin
  - seq64::sequence, 1042
- number



- seq64::sequence, 996
- off\_playing\_notes
  - seq64::sequence, 1035
- off\_queued
  - seq64::sequence, 1005
- off\_sequences
  - seq64::perform, 736
- offset
  - seq64::trigger, 1056
- on\_button\_press\_event
  - seq64::AbstractPerfInput, 135
  - seq64::FruityPerfInput, 265
  - seq64::FruitySeqEventInput, 270
  - seq64::FruitySeqRollInput, 273
  - seq64::Seq24PerfInput, 853
  - seq64::Seq24SeqEventInput, 857
  - seq64::eventslots, 253
  - seq64::mainwid, 410
  - seq64::perfnames, 673
  - seq64::perfroll, 776
  - seq64::perftime, 791
  - seq64::seqdata, 867
  - seq64::seqevent, 913
  - seq64::seqkeys, 926
  - seq64::seqroll, 967
  - seq64::seqtime, 981
- on\_button\_release\_event
  - seq64::AbstractPerfInput, 135
  - seq64::FruityPerfInput, 266
  - seq64::FruitySeqEventInput, 270
  - seq64::FruitySeqRollInput, 273
  - seq64::Seq24PerfInput, 854
  - seq64::Seq24SeqEventInput, 857
  - seq64::eventslots, 253
  - seq64::mainwid, 410
  - seq64::perfnames, 673
  - seq64::perfroll, 776
  - seq64::perftime, 791
  - seq64::seqdata, 867
  - seq64::seqevent, 913
  - seq64::seqkeys, 926
  - seq64::seqroll, 967
  - seq64::seqtime, 981
- on\_delete\_event
  - seq64::eventedit, 234
  - seq64::mainwnd, 429
  - seq64::perfedit, 662
  - seq64::seqedit, 890
- on\_enter\_notify\_event
  - seq64::seqkeys, 927
  - seq64::seqroll, 970
- on\_expose\_event
  - seq64::eventslots, 253
  - seq64::maintime, 398
  - seq64::mainwid, 409
  - seq64::perfnames, 672
  - seq64::perfroll, 776
  - seq64::perftime, 790
- seq64::seqdata, 866
- seq64::seqevent, 913
- seq64::seqkeys, 926
- seq64::seqroll, 967
- seq64::seqtime, 981
- on\_focus\_in\_event
  - seq64::eventedit, 234
  - seq64::eventslots, 253
  - seq64::mainwid, 411
  - seq64::perfroll, 777
  - seq64::seqedit, 890
  - seq64::seqevent, 914
  - seq64::seqroll, 968
- on\_focus\_out\_event
  - seq64::eventedit, 234
  - seq64::eventslots, 254
  - seq64::lfownd, 392
  - seq64::mainwid, 411
  - seq64::perfroll, 777
  - seq64::seqedit, 890
  - seq64::seqevent, 914
  - seq64::seqroll, 968
- on\_frame\_down
  - seq64::eventslots, 254
- on\_frame\_end
  - seq64::eventslots, 255
- on\_frame\_home
  - seq64::eventslots, 255
- on\_frame\_up
  - seq64::eventslots, 254
- on\_grouplearnchange
  - seq64::mainwnd, 430
  - seq64::performcallback, 764
- on\_key\_press\_event
  - seq64::eventedit, 234
  - seq64::keybindentry, 343
  - seq64::mainwnd, 429
  - seq64::perfedit, 661
  - seq64::perfroll, 778
  - seq64::seqedit, 891
  - seq64::seqevent, 914
  - seq64::seqroll, 969
- on\_key\_release\_event
  - seq64::mainwnd, 429
  - seq64::perfedit, 662
- on\_leave\_notify\_event
  - seq64::seqdata, 868
  - seq64::seqkeys, 927
  - seq64::seqroll, 970
- on\_left\_button\_pressed
  - seq64::FruityPerfInput, 267
- on\_motion\_notify\_event
  - seq64::AbstractPerfInput, 135
  - seq64::FruityPerfInput, 266
  - seq64::FruitySeqEventInput, 271
  - seq64::FruitySeqRollInput, 273
  - seq64::Seq24PerfInput, 854
  - seq64::Seq24SeqEventInput, 857

- seq64::mainwid, [411](#)
- seq64::perfroll, [777](#)
- seq64::seqdata, [867](#)
- seq64::seqevent, [914](#)
- seq64::seqkeys, [927](#)
- seq64::seqroll, [968](#)
- on\_move\_down
  - seq64::eventslots, [254](#)
- on\_move\_up
  - seq64::eventslots, [254](#)
- on\_queued
  - seq64::sequence, [1005](#)
- on\_realize
  - seq64::eventedit, [233](#)
  - seq64::eventslots, [253](#)
  - seq64::gui\_drawingarea\_gtk2, [296](#)
  - seq64::gui\_window\_gtk2, [313](#)
  - seq64::maintime, [398](#)
  - seq64::mainwid, [409](#)
  - seq64::mainwnd, [429](#)
  - seq64::perfedit, [661](#)
  - seq64::perfnames, [672](#)
  - seq64::perfroll, [775](#)
  - seq64::perftime, [790](#)
  - seq64::seqdata, [866](#)
  - seq64::seqedit, [889](#)
  - seq64::seqevent, [912](#)
  - seq64::seqkeys, [925](#)
  - seq64::seqmenu, [941](#)
  - seq64::seqroll, [967](#)
  - seq64::seqtime, [981](#)
- on\_right\_button\_pressed
  - seq64::FruityPerfInput, [267](#)
- on\_scroll\_event
  - seq64::eventslots, [254](#)
  - seq64::perfnames, [674](#)
  - seq64::perfroll, [777](#)
  - seq64::seqdata, [868](#)
  - seq64::seqedit, [890](#)
  - seq64::seqkeys, [927](#)
  - seq64::seqroll, [969](#)
- on\_set\_focus
  - seq64::eventedit, [233](#)
  - seq64::seqedit, [890](#)
- on\_size\_allocate
  - seq64::eventslots, [254](#)
  - seq64::perfnames, [674](#)
  - seq64::perfroll, [777](#)
  - seq64::perftime, [791](#)
  - seq64::seqdata, [868](#)
  - seq64::seqevent, [915](#)
  - seq64::seqkeys, [928](#)
  - seq64::seqroll, [970](#)
  - seq64::seqtime, [981](#)
- on\_size\_request
  - seq64::perfroll, [778](#)
- open\_client
  - seq64::midi\_in\_jack, [511](#)
  - seq64::midi\_jack, [530](#)
  - seq64::midi\_out\_jack, [561](#)
- open\_client\_impl
  - seq64::midi\_jack, [528](#)
- open\_file
  - seq64::mainwnd, [421](#)
- open\_performance\_edit
  - seq64::mainwnd, [424](#)
- open\_performance\_edit\_2
  - seq64::mainwnd, [424](#)
- openmidi\_api
  - seq64::rtmidi\_in, [833](#)
  - seq64::rtmidi\_info, [848](#)
  - seq64::rtmidi\_out, [852](#)
- operator<
  - seq64::event, [194](#)
  - seq64::event\_list::event\_key, [210](#)
  - seq64::trigger, [1054](#)
- operator=
  - seq64::automutex, [138](#)
  - seq64::click, [156](#)
  - seq64::editable\_event, [172](#)
  - seq64::editable\_events, [183](#)
  - seq64::event, [193](#)
  - seq64::event\_list, [214](#)
  - seq64::gui\_drawingarea\_gtk2, [284](#)
  - seq64::keystroke, [383](#)
  - seq64::maintime, [397](#)
  - seq64::rc\_settings, [801](#)
  - seq64::sequence, [994](#)
  - seq64::triggers, [1061](#)
  - seq64::user\_instrument, [1075](#)
  - seq64::user\_midi\_bus, [1081](#)
  - seq64::user\_settings, [1094](#)
- operator[]
  - seq64::midi\_message, [554](#)
- options
  - seq64::keybindentry, [343](#)
  - seq64::keys\_perform, [369](#)
  - seq64::options, [641](#)
  - seq64::perform, [748](#)
  - seq64::rc\_settings, [812](#)
- options\_dialog
  - seq64::mainwnd, [426](#)
- optionsfile
  - seq64::keys\_perform, [369](#)
  - seq64::optionsfile, [647](#)
  - seq64::perform, [747](#)
  - seq64::rc\_settings, [811](#)
- orange
  - seq64::gui\_palette\_gtk2, [304](#)
- output
  - seq64::jack\_assistant, [324](#)
- output\_func
  - seq64::perform, [710](#)
- output\_ports
  - seq64::midi\_info, [515](#)
- output\_thread\_func

- seq64, 99
- seq64::perform, 748
- padded\_height
  - seq64::font, 261
- page\_decrement\_beats\_per\_minute
  - seq64::perform, 718
- page\_increment\_beats\_per\_minute
  - seq64::perform, 718
- page\_movement
  - seq64::eventslots, 251
- page\_topper
  - seq64::eventslots, 251
- pager\_index
  - seq64::eventslots, 245
- paint
  - seq64::event, 203
- parent
  - seq64::editable\_event, 172
  - seq64::jack\_assistant, 319
- parent\_bus
  - seq64::midi\_api, 489
- parse
  - seq64::configfile, 164
  - seq64::midifile, 620
  - seq64::optionsfile, 647
  - seq64::userfile, 1120
- parse\_command\_line\_options
  - seq64, 87
  - seq64::rc\_settings, 812
- parse\_options\_files
  - seq64, 86
- parse\_prop\_header
  - seq64::midifile, 623
- parse\_proprietary\_track
  - seq64::midifile, 624
- parse\_smf\_0
  - seq64::midifile, 622
- parse\_smf\_1
  - seq64::midifile, 623
- partial\_assign
  - seq64::sequence, 994
- pass\_sysex
  - seq64::rc\_settings, 803, 807
- paste
  - seq64::triggers, 1066
- paste\_selected
  - seq64::sequence, 1026
- paste\_trigger
  - seq64::sequence, 1017
- pattern\_edit
  - seq64::keys\_perform, 358
- pause
  - seq64::keys\_perform, 358
  - seq64::sequence, 1036
- pause\_key
  - seq64::perform, 717
- pause\_playing
  - seq64::mainwnd, 423
- seq64::perfedit, 661
- seq64::perform, 716
- perf
  - seq64::gui\_drawingarea\_gtk2, 285
  - seq64::gui\_window\_gtk2, 312
  - seq64::options, 641
- perf\_h\_page\_increment
  - seq64::user\_settings, 1102, 1107
- perf\_modify
  - seq64::eventedit, 231
- perf\_v\_page\_increment
  - seq64::user\_settings, 1102, 1107
- perfedit
  - seq64::perfedit, 654
  - seq64::perfnames, 674
  - seq64::perform, 748
  - seq64::perfroll, 779
  - seq64::perftime, 792
- perfnames
  - seq64::perfnames, 670
- perform
  - seq64::keys\_perform, 369
  - seq64::mastermidibase, 453
  - seq64::perform, 691
  - seq64::sequence, 1045
- perfroll
  - seq64::AbstractPerfInput, 136
  - seq64::FruityPerfInput, 268
  - seq64::Seq24PerfInput, 855
  - seq64::perform, 748
  - seq64::perfroll, 769
- perfroll\_key\_event
  - seq64::perform, 722
- perftime
  - seq64::perftime, 786
- pixel\_to\_tick
  - seq64::perftime, 789
- play
  - seq64::busarray, 143
  - seq64::mastermidibase, 442
  - seq64::midibase, 596
  - seq64::perform, 738
  - seq64::sequence, 1009
  - seq64::triggers, 1063
- play\_change\_callback
  - seq64::seqedit, 885
- play\_note\_off
  - seq64::sequence, 1035
- play\_note\_on
  - seq64::sequence, 1035
- play\_queue
  - seq64::sequence, 1009
- playback\_key\_event
  - seq64::perform, 722
- pointer\_position
  - seq64::keys\_perform, 360, 361
- poll\_for\_midi
  - seq64::busarray, 146

- seq64::mastermidibase, 446
- seq64::midibase, 594
- pop
  - seq64::midi\_queue, 566
- pop\_front
  - seq64::midi\_queue, 567
- pop\_redo
  - seq64::sequence, 998
  - seq64::triggers, 1062
- pop\_trigger\_redo
  - seq64::perform, 724
  - seq64::sequence, 998
- pop\_trigger\_undo
  - seq64::perform, 724
  - seq64::sequence, 998
- pop\_undo
  - seq64::sequence, 998
  - seq64::triggers, 1062
- popup\_event\_menu
  - seq64::seqedit, 887
- popup\_menu
  - seq64::perfedit, 660
  - seq64::seqedit, 887
  - seq64::seqmenu, 937
- popup\_midibus\_menu
  - seq64::seqedit, 887
- popup\_midich\_menu
  - seq64::seqedit, 888
- popup\_sequence\_menu
  - seq64::seqedit, 888
- popup\_tool\_menu
  - seq64::seqedit, 888
- port\_exit
  - seq64::busarray, 145
  - seq64::mastermidibase, 441
- port\_handle
  - seq64::midi\_jack, 528
- port\_list
  - seq64::mastermidibus, 464
  - seq64::midi\_info, 519
  - seq64::rtmidi\_info, 847
- port\_name
  - seq64::midibase, 589, 598
- port\_settings
  - seq64::mastermidibase, 450
- port\_start
  - seq64::mastermidibase, 441
- position
  - seq64::jack\_assistant, 323
  - seq64::midi\_container, 496
- position\_increment
  - seq64::midi\_container, 496
- position\_jack
  - seq64::perform, 736
- position\_reset
  - seq64::midi\_container, 496
- pow2
  - seq64::midifile, 625
- ppqn
  - seq64::mainwnd, 422
  - seq64::midi\_info, 516
  - seq64::midi\_splitter, 572
  - seq64::midi\_timing, 577
  - seq64::midibase, 590
  - seq64::midifile, 622
  - seq64::perform, 692
  - seq64::rtmidi\_info, 845
- ppqn\_is\_valid
  - seq64, 92
- print
  - seq64::busarray, 144
  - seq64::event, 207
  - seq64::event\_list, 221
  - seq64::mastermidibase, 444
  - seq64::midibase, 597
  - seq64::sequence, 1009
  - seq64::triggers, 1062
- print\_keys
  - seq64::rc\_settings, 804, 808
- print\_message
  - seq64::rterror, 821
- print\_triggers
  - seq64::perform, 703
  - seq64::sequence, 1009
- priority
  - seq64::rc\_settings, 802, 807
- private\_bus
  - seq64::user\_settings, 1109
- private\_instrument
  - seq64::user\_settings, 1109
- process\_events
  - seq64::lash, 388
- progress\_bar\_colored
  - seq64::user\_settings, 1102, 1107
- progress\_bar\_thick
  - seq64::user\_settings, 1102, 1107
- progress\_color
  - seq64::gui\_palette\_gtk2, 303
- prop\_item\_size
  - seq64::midifile, 632
- pulse\_length\_us
  - seq64, 79
- pulses\_per\_measure
  - seq64, 82
- pulses\_to\_measurestring
  - seq64, 72
- pulses\_to\_midi\_measures
  - seq64, 73
- pulses\_to\_string
  - seq64, 72
- pulses\_to\_timestring
  - seq64, 73, 74
- push
  - seq64::midi\_message, 555
- push\_back
  - seq64::event\_list, 216

- push\_quantize
  - seq64::sequence, [1040](#)
- push\_trigger\_undo
  - seq64::perform, [723](#)
  - seq64::sequence, [998](#)
- push\_undo
  - seq64::sequence, [997](#)
  - seq64::triggers, [1062](#)
- put
  - seq64::midi\_container, [495](#)
  - seq64::midi\_list, [549](#)
  - seq64::midi\_vector, [581](#)
- put\_event\_on\_bus
  - seq64::sequence, [1043](#)
- q\_rec\_change\_callback
  - seq64::seqedit, [885](#)
- quantize\_events
  - seq64::sequence, [1039](#)
- query\_save\_changes
  - seq64::mainwnd, [427](#)
- queue
  - seq64::keys\_perform, [353](#), [354](#)
  - seq64::rtmidi\_in\_data, [835](#)
- queue\_number
  - seq64::midi\_info, [519](#)
  - seq64::midibase, [592](#)
  - seq64::rtmidi\_info, [845](#)
- quit
  - seq64::gui\_assistant, [276](#)
  - seq64::gui\_assistant\_gtk2, [279](#)
  - seq64::gui\_window\_gtk2, [312](#)
- rc
  - seq64, [99](#)
- rc\_error\_dialog
  - seq64::mainwnd, [421](#)
- rc\_settings
  - seq64::rc\_settings, [800](#)
- read\_byte
  - seq64::midifile, [626](#)
- read\_byte\_array
  - seq64::midifile, [628](#)
- read\_long
  - seq64::midifile, [626](#)
- read\_seq\_number
  - seq64::midifile, [630](#)
- read\_short
  - seq64::midifile, [626](#)
- read\_track\_name
  - seq64::midifile, [629](#)
- read\_varinum
  - seq64::midifile, [627](#)
- record\_change\_callback
  - seq64::seqedit, [885](#)
- red
  - seq64::gui\_palette\_gtk2, [304](#)
- redo
  - seq64::perfedit, [660](#)
- redo\_callback
  - seq64::seqedit, [886](#)
- redraw
  - seq64::mainwid, [406](#)
  - seq64::perfnames, [672](#)
  - seq64::seqdata, [862](#)
  - seq64::seqevent, [908](#)
  - seq64::seqmenu, [939](#)
  - seq64::seqroll, [955](#)
  - seq64::seqtime, [979](#)
- redraw\_dirty\_sequences
  - seq64::perfnames, [671](#)
  - seq64::perfroll, [771](#)
- redraw\_events
  - seq64::seqroll, [955](#)
- redraw\_period\_ms
  - seq64::gui\_window\_gtk2, [312](#)
- redraw\_progress
  - seq64::perfroll, [771](#)
- ref\_midi\_port\_info
  - seq64::midi\_info, [521](#)
- register\_port
  - seq64::midi\_jack, [530](#)
- remote\_port\_name
  - seq64::midi\_jack, [527](#)
- remove
  - seq64::businfo, [150](#)
  - seq64::editable\_events, [186](#)
  - seq64::event\_list, [217](#)
  - seq64::sequence, [1044](#), [1045](#)
  - seq64::triggers, [1065](#)
- remove\_all
  - seq64::sequence, [1045](#)
- remove\_marked
  - seq64::event\_list, [220](#)
  - seq64::sequence, [1034](#)
- remove\_selected
  - seq64::sequence, [1034](#)
  - seq64::triggers, [1066](#)
- remove\_seqedit
  - seq64::seqmenu, [938](#)
- render\_number
  - seq64::seqdata, [866](#)
- render\_string
  - seq64::gui\_drawingarea\_gtk2, [289](#)
- render\_string\_on\_drawable
  - seq64::font, [260](#)
- render\_string\_on\_pixmap
  - seq64::gui\_drawingarea\_gtk2, [289](#)
- replace
  - seq64::editable\_events, [186](#)
  - seq64::keys\_perform, [353](#)
- replacement\_port
  - seq64::busarray, [147](#)
- reposition
  - seq64::perform, [700](#)
- reset
  - seq64::mainwid, [405](#)

- seq64::perftime, 787
- seq64::seqdata, 862
- seq64::seqevent, 908
- seq64::seqkeys, 925
- seq64::seqroll, 954
- seq64::seqtime, 979
- reset\_draw\_marker
  - seq64::sequence, 1036
- reset\_draw\_trigger\_marker
  - seq64::sequence, 1036
  - seq64::triggers, 1070
- reset\_sequences
  - seq64::perform, 737
- restart\_sysex
  - seq64::event, 202
- restore\_playing\_state
  - seq64::perform, 712
- RevSlotMap
  - seq64::keys\_perform, 351
- reveal\_alsa\_ports
  - seq64::rc\_settings, 804, 808
- rewind
  - seq64::keys\_perform, 360
  - seq64::perfedit, 656
  - seq64::perform, 699
- rterror
  - seq64::rterror, 820
- rterror\_callback
  - seq64, 65
- rtmidi
  - seq64::rtmidi, 824
- rtmidi\_api
  - seq64, 70
- rtmidi\_callback\_t
  - seq64, 65
- rtmidi\_in
  - seq64::rtmidi\_in, 832
  - seq64::rtmidi\_info, 849
- rtmidi\_in\_data
  - seq64::rtmidi\_in\_data, 835
- rtmidi\_info
  - seq64::midi\_info, 521
  - seq64::rc\_settings, 812
  - seq64::rtmidi\_info, 842
- rtmidi\_out
  - seq64::rtmidi\_info, 849
  - seq64::rtmidi\_out, 851
- s\_arg\_list
  - seq64, 124
- s\_build\_alsamidi\_support
  - seq64, 125
- s\_build\_chord\_generator
  - seq64, 126
- s\_build\_edit\_highlight
  - seq64, 126
- s\_build\_follow\_progress
  - seq64, 127
- s\_build\_highlight\_empty
  - seq64, 125
- s\_build\_jack\_session
  - seq64, 126
- s\_build\_jack\_support
  - seq64, 126
- s\_build\_lash\_support
  - seq64, 126
- s\_build\_midi\_vector
  - seq64, 127
- s\_build\_pause\_support
  - seq64, 126
- s\_build\_portmidi\_support
  - seq64, 125
- s\_build\_rtmidi\_support
  - seq64, 125
- s\_build\_solid\_grid
  - seq64, 127
- s\_build\_timesig\_tempo
  - seq64, 127
- s\_build\_use\_event\_map
  - seq64, 126
- s\_character\_mapping
  - seq64, 128
- s\_debug\_mode
  - seq64, 128
- s\_event\_editor
  - seq64, 126
- s\_global\_lash\_driver
  - seq64, 128
- s\_handlesize
  - seq64, 130, 131
- s\_help\_1a
  - seq64, 124
- s\_help\_1b
  - seq64, 125
- s\_help\_2
  - seq64, 125
- s\_help\_3
  - seq64, 125
- s\_help\_4
  - seq64, 125
- s\_jitter\_amount
  - seq64, 130
- s\_seq32\_jack\_support
  - seq64, 127
- s\_seq32\_lfo\_support
  - seq64, 128
- s\_seq32\_menu\_buttons
  - seq64, 128
- s\_seq32\_transport
  - seq64, 127
- s\_seq32\_transpose
  - seq64, 128
- s\_statistics\_support
  - seq64, 127
- s\_status\_pairs
  - seq64, 128
- s\_strip\_empty\_mutes

- seq64, 127
- save\_clock
  - seq64::mastermidibase, 453
- save\_events
  - seq64::editable\_events, 183
  - seq64::eventslots, 248
- save\_file
  - seq64::mainwnd, 428
- save\_input
  - seq64::mastermidibase, 453
- save\_playing\_state
  - seq64::perform, 712
- save\_user\_config
  - seq64::user\_settings, 1102, 1103
- scale\_lfo\_change
  - seq64::lflownd, 392
- screenset\_dn
  - seq64::keys\_perform, 355
- screenset\_up
  - seq64::keys\_perform, 355
- scroll\_hadjust
  - seq64::gui\_drawingarea\_gtk2, 294
  - seq64::gui\_window\_gtk2, 312
- scroll\_hset
  - seq64::gui\_drawingarea\_gtk2, 295
  - seq64::gui\_window\_gtk2, 313
- scroll\_offset\_x
  - seq64::seqroll, 963
- scroll\_offset\_y
  - seq64::seqroll, 965
- scroll\_vadjust
  - seq64::gui\_drawingarea\_gtk2, 294
  - seq64::gui\_window\_gtk2, 313
- scroll\_vset
  - seq64::gui\_drawingarea\_gtk2, 295
  - seq64::gui\_window\_gtk2, 313
- select
  - seq64::event, 204
  - seq64::triggers, 1065
- select\_action
  - seq64::seqroll, 966
- select\_action\_e
  - seq64::sequence, 993
- select\_all
  - seq64::event\_list, 221
  - seq64::sequence, 1025
- select\_all\_notes
  - seq64::sequence, 1024
- select\_and\_mute\_group
  - seq64::perform, 733
- select\_even\_or\_odd\_notes
  - seq64::sequence, 1024
- select\_event
  - seq64::eventslots, 248
- select\_event\_handle
  - seq64::sequence, 1023
- select\_events
  - seq64::sequence, 1021, 1022
- select\_fg\_bg\_colors
  - seq64::mainwid, 409
- select\_group\_mute
  - seq64::perform, 734
- select\_linked
  - seq64::sequence, 1023
- select\_note\_events
  - seq64::sequence, 1021
- select\_trigger
  - seq64::sequence, 1015
- selected
  - seq64::trigger, 1056
- selected\_api
  - seq64::rtmidi\_info, 847
- selected\_trigger\_end
  - seq64::sequence, 1018
- selected\_trigger\_start
  - seq64::sequence, 1018
- selecting
  - seq64::seqroll, 966
- send\_message
  - seq64::midi\_out\_jack, 560
- seq
  - seq64::midi\_alsa\_info, 478
- Seq24PerfInput
  - seq64::Seq24PerfInput, 853
  - seq64::perfroll, 778
  - seq64::triggers, 1072
- Seq24SeqEventInput
  - seq64::Seq24SeqEventInput, 856
  - seq64::seqevent, 915
- seq64, 49
  - adjustment\_dummy, 102
  - bpm\_from\_tempo\_us, 79
  - build\_details, 87
  - bussbyte, 64
  - c\_backsequence, 117
  - c\_bpmtag, 116
  - c\_chord\_number, 123
  - c\_chord\_size, 123
  - c\_chord\_table, 123
  - c\_chord\_table\_text, 123
  - c\_chord\_text, 123
  - c\_controller\_names, 109
  - c\_interval\_text, 123
  - c\_key\_text, 123
  - c\_mainwid\_x, 130
  - c\_mainwid\_y, 130
  - c\_max\_busses, 124
  - c\_max\_instruments, 124
  - c\_midi\_control\_16, 120
  - c\_midi\_control\_17, 120
  - c\_midi\_control\_18, 120
  - c\_midi\_control\_19, 120
  - c\_midi\_control\_bpm\_dn, 118
  - c\_midi\_control\_bpm\_page\_dn, 120
  - c\_midi\_control\_bpm\_page\_up, 120
  - c\_midi\_control\_bpm\_up, 118



- [c\\_midi\\_control\\_mod\\_glearn, 119](#)
- [c\\_midi\\_control\\_mod\\_gmute, 119](#)
- [c\\_midi\\_control\\_mod\\_queue, 119](#)
- [c\\_midi\\_control\\_mod\\_replace, 118](#)
- [c\\_midi\\_control\\_mod\\_snapshot, 118](#)
- [c\\_midi\\_control\\_play\\_ss, 119](#)
- [c\\_midi\\_control\\_playback, 119](#)
- [c\\_midi\\_control\\_record, 119](#)
- [c\\_midi\\_control\\_solo, 119](#)
- [c\\_midi\\_control\\_ss\\_dn, 118](#)
- [c\\_midi\\_control\\_ss\\_up, 118](#)
- [c\\_midi\\_control\\_thru, 120](#)
- [c\\_midi\\_controls, 119](#)
- [c\\_midi\\_controls\\_extended, 120](#)
- [c\\_midi\\_track\\_ctrl, 117](#)
- [c\\_midibus, 114](#)
- [c\\_midibus\\_input\\_size, 114](#)
- [c\\_midibus\\_output\\_size, 114](#)
- [c\\_midibus\\_sysex\\_chunk, 114](#)
- [c\\_midich, 115](#)
- [c\\_midiclocks, 116](#)
- [c\\_midictrl, 116](#)
- [c\\_music\\_scales, 68](#)
- [c\\_musickey, 117](#)
- [c\\_musicscale, 117](#)
- [c\\_mutegroups, 116](#)
- [c\\_notes, 116](#)
- [c\\_perf\\_bp\\_mes, 117](#)
- [c\\_perf\\_bw, 117](#)
- [c\\_scales\\_policy, 121](#)
- [c\\_scales\\_text, 122](#)
- [c\\_scales\\_transpose\\_dn, 122](#)
- [c\\_scales\\_transpose\\_up, 121](#)
- [c\\_status\\_queue, 129](#)
- [c\\_status\\_replace, 129](#)
- [c\\_status\\_snapshot, 129](#)
- [c\\_timesig, 116](#)
- [c\\_transpose, 117](#)
- [c\\_triggers, 116](#)
- [c\\_triggers\\_new, 116](#)
- [choose\\_ppqn, 100](#)
- [clamp, 105, 106](#)
- [clock\\_e, 68](#)
- [clock\\_tick\\_duration\\_bogus, 81](#)
- [clock\\_ticks\\_from\\_ppqn, 82](#)
- [create\\_jack\\_client, 94](#)
- [create\\_lash\\_driver, 97](#)
- [delete\\_lash\\_driver, 98](#)
- [delta\\_time\\_us\\_to\\_ticks, 80](#)
- [double\\_ticks\\_from\\_ppqn, 82](#)
- [draw\\_type\\_t, 69](#)
- [EVENT\\_AFTERTOUCH, 110](#)
- [EVENT\\_ANY, 109](#)
- [EVENT\\_CHANNEL\\_PRESSURE, 110](#)
- [EVENT\\_CLEAR\\_CHAN\\_MASK, 113](#)
- [EVENT\\_CONTROL\\_CHANGE, 110](#)
- [EVENT\\_GET\\_CHAN\\_MASK, 113](#)
- [EVENT\\_MIDI\\_ACTIVE\\_SENS, 113](#)
- [EVENT\\_MIDI\\_CLOCK, 112](#)
- [EVENT\\_MIDI\\_CONTINUE, 112](#)
- [EVENT\\_MIDI\\_META, 113](#)
- [EVENT\\_MIDI\\_QUARTER\\_FRAME, 111](#)
- [EVENT\\_MIDI\\_RESET, 113](#)
- [EVENT\\_MIDI\\_SONG\\_F4, 111](#)
- [EVENT\\_MIDI\\_SONG\\_F5, 111](#)
- [EVENT\\_MIDI\\_SONG\\_F9, 112](#)
- [EVENT\\_MIDI\\_SONG\\_FD, 113](#)
- [EVENT\\_MIDI\\_SONG\\_POS, 111](#)
- [EVENT\\_MIDI\\_SONG\\_SELECT, 111](#)
- [EVENT\\_MIDI\\_START, 112](#)
- [EVENT\\_MIDI\\_STOP, 112](#)
- [EVENT\\_MIDI\\_SYSEX\\_CONTINUE, 112](#)
- [EVENT\\_MIDI\\_SYSEX\\_END, 112](#)
- [EVENT\\_MIDI\\_SYSEX, 111](#)
- [EVENT\\_MIDI\\_TUNE\\_SELECT, 112](#)
- [EVENT\\_NOTE\\_OFF, 110](#)
- [EVENT\\_NOTE\\_ON, 110](#)
- [EVENT\\_NULL\\_CHANNEL, 113](#)
- [EVENT\\_PITCH\\_WHEEL, 110](#)
- [EVENT\\_PROGRAM\\_CHANGE, 110](#)
- [EVENT\\_STATUS\\_BIT, 109](#)
- [EVENTS\\_ALL, 114](#)
- [EVENTS\\_UNSELECTED, 114](#)
- [edit\\_action\\_t, 69](#)
- [error\\_message, 88](#)
- [extract\\_bus\\_name, 84](#)
- [extract\\_port\\_name, 85](#)
- [extract\\_port\\_names, 84](#)
- [extract\\_timing\\_numbers, 71](#)
- [FF\\_RW\\_timeout, 105](#)
- [file\\_access, 89](#)
- [file\\_accessible, 90](#)
- [file\\_executable, 91](#)
- [file\\_exists, 89](#)
- [file\\_is\\_directory, 91](#)
- [file\\_readable, 90](#)
- [file\\_writable, 90](#)
- [font\\_render, 102](#)
- [g\\_midi\\_control\\_limit, 121](#)
- [g\\_rc\\_settings, 130](#)
- [g\\_user\\_settings, 130](#)
- [get\\_current\\_jack\\_position, 96](#)
- [gs\\_mainwid\\_pointer, 130](#)
- [gs\\_perfedit\\_pointer\\_0, 131](#)
- [gs\\_perfedit\\_pointer\\_1, 131](#)
- [help\\_check, 85](#)
- [info\\_message, 88](#)
- [input\\_thread\\_func, 99](#)
- [interaction\\_method\\_t, 68](#)
- [invalid\\_key, 97](#)
- [is\\_ctrl\\_key, 102–104](#)
- [is\\_ctrl\\_shift\\_key, 104](#)
- [is\\_no\\_modifier, 103](#)
- [is\\_null\\_midipulse, 98](#)
- [is\\_shift\\_key, 103, 104](#)
- [is\\_super\\_key, 104](#)



jack\_dummy\_callback, 100  
jack\_message\_bit\_bucket, 109  
jack\_process\_io, 109  
jack\_process\_rtmidi\_input, 107  
jack\_process\_rtmidi\_output, 108  
jack\_session\_callback, 96  
jack\_shutdown\_callback, 93  
jack\_sync\_callback, 92  
jack\_timebase\_callback, 93  
jack\_transport\_callback, 94  
keyval\_name, 97  
keyval\_normalize, 97  
lash\_driver, 98  
log2\_time\_sig\_value, 78  
long\_options, 124  
make\_directory, 91  
make\_section\_name, 101  
measures\_to\_ticks, 83  
measurestring\_to\_pulses, 74  
message\_concatenate, 88  
midi\_api\_name, 106  
midi\_input\_callback, 109  
midi\_input\_test, 107  
midi\_measures\_to\_pulses, 74  
midi\_probe, 106  
midibpm, 65  
midibyte, 64  
midilong, 65  
midipulse, 65  
midishort, 64  
millisleep, 98  
min, 107  
mouse\_action\_e, 69  
output\_thread\_func, 99  
parse\_command\_line\_options, 87  
parse\_options\_files, 86  
ppqn\_is\_valid, 92  
pulse\_length\_us, 79  
pulses\_per\_measure, 82  
pulses\_to\_measurestring, 72  
pulses\_to\_midi\_measures, 73  
pulses\_to\_string, 72  
pulses\_to\_timestring, 73, 74  
rc, 99  
rterror\_callback, 65  
rtmidi\_api, 70  
rtmidi\_callback\_t, 65  
s\_arg\_list, 124  
s\_build\_alsamidi\_support, 125  
s\_build\_chord\_generator, 126  
s\_build\_edit\_highlight, 126  
s\_build\_follow\_progress, 127  
s\_build\_highlight\_empty, 125  
s\_build\_jack\_session, 126  
s\_build\_jack\_support, 126  
s\_build\_lash\_support, 126  
s\_build\_midi\_vector, 127  
s\_build\_pause\_support, 126  
s\_build\_portmidi\_support, 125  
s\_build\_rtmidi\_support, 125  
s\_build\_solid\_grid, 127  
s\_build\_timesig\_tempo, 127  
s\_build\_use\_event\_map, 126  
s\_character\_mapping, 128  
s\_debug\_mode, 128  
s\_event\_editor, 126  
s\_global\_lash\_driver, 128  
s\_handlesize, 130, 131  
s\_help\_1a, 124  
s\_help\_1b, 125  
s\_help\_2, 125  
s\_help\_3, 125  
s\_help\_4, 125  
s\_jitter\_amount, 130  
s\_seq32\_jack\_support, 127  
s\_seq32\_lfo\_support, 128  
s\_seq32\_menu\_buttons, 128  
s\_seq32\_transport, 127  
s\_seq32\_transpose, 128  
s\_statistics\_support, 127  
s\_status\_pairs, 128  
s\_strip\_empty\_mutes, 127  
seq\_app\_name, 101  
seq\_client\_name, 101  
seq\_event\_type\_t, 67  
seq\_modifier\_t, 66  
seq\_scroll\_direction\_t, 67  
seq\_version, 101  
shorten\_file\_spec, 76  
show\_jack\_statuses, 96  
silence\_jack\_errors, 106  
silence\_jack\_info, 106  
string\_is\_void, 77  
string\_not\_void, 76  
string\_to\_midibyte, 76  
string\_to\_pulses, 75  
strings\_match, 77  
swap, 70  
tempo\_us\_from\_bpm, 79  
tempo\_us\_to\_bytes, 78  
test\_widget\_click, 105  
ticks\_to\_delta\_time\_us, 81  
timestring\_to\_pulses, 75, 100  
to\_string, 89  
update\_mainwid\_sequences, 105  
update\_perfedit\_sequences, 105  
usr, 99  
versiontext, 124  
wave\_func, 83  
wave\_type\_name, 71  
wave\_type\_t, 66  
write\_options\_files, 87  
zoom\_power\_of\_2, 78  
seq64::AbstractPerfInput, 133  
~AbstractPerfInput, 134  
AbstractPerfInput, 134

- activate\_adding, 135
- handle\_motion\_key, 135
- is\_adding, 135
- is\_adding\_pressed, 136
- m\_adding, 136
- m\_adding\_pressed, 136
- on\_button\_press\_event, 135
- on\_button\_release\_event, 135
- on\_motion\_notify\_event, 135
- perfroll, 136
- set\_adding, 136
- set\_adding\_pressed, 136
- seq64::FruityPerfInput, 263
  - activate\_adding, 268
  - FruityPerfInput, 265
  - handle\_motion\_key, 268
  - m\_current\_x, 268
  - m\_current\_y, 268
  - on\_button\_press\_event, 265
  - on\_button\_release\_event, 266
  - on\_left\_button\_pressed, 267
  - on\_motion\_notify\_event, 266
  - on\_right\_button\_pressed, 267
  - perfroll, 268
  - update\_mouse\_pointer, 266
- seq64::FruitySeqEventInput, 269
  - FruitySeqEventInput, 269
  - m\_is\_drag\_pasting, 271
  - m\_is\_drag\_pasting\_start, 271
  - m\_justselected\_one, 271
  - on\_button\_press\_event, 270
  - on\_button\_release\_event, 270
  - on\_motion\_notify\_event, 271
  - update\_mouse\_pointer, 269
- seq64::FruitySeqRollInput, 272
  - FruitySeqRollInput, 272
  - m\_drag\_paste\_start\_pos, 274
  - m\_erase\_painting, 274
  - on\_button\_press\_event, 273
  - on\_button\_release\_event, 273
  - on\_motion\_notify\_event, 273
  - update\_mouse\_pointer, 272
- seq64::Seq24PerfInput, 852
  - activate\_adding, 854
  - handle\_motion\_key, 855
  - m\_effective\_tick, 856
  - on\_button\_press\_event, 853
  - on\_button\_release\_event, 854
  - on\_motion\_notify\_event, 854
  - perfroll, 855
  - Seq24PerfInput, 853
- seq64::Seq24SeqEventInput, 856
  - m\_adding, 858
  - on\_button\_press\_event, 857
  - on\_button\_release\_event, 857
  - on\_motion\_notify\_event, 857
  - Seq24SeqEventInput, 856
  - set\_adding, 856
- seq64::automutex, 137
  - ~automutex, 138
  - automutex, 137
  - m\_safety\_mutex, 138
  - operator=, 138
- seq64::busarray, 138
  - ~busarray, 140
  - add, 140, 141
  - bus, 141
  - busarray, 140
  - clock, 142
  - continue\_from, 142
  - count, 141
  - get\_clock, 144
  - get\_input, 146
  - get\_midi\_bus\_name, 144
  - get\_midi\_event, 147
  - init\_clock, 142
  - initialize, 141
  - is\_system\_port, 146
  - m\_container, 148
  - play, 143
  - poll\_for\_midi, 146
  - port\_exit, 145
  - print, 144
  - replacement\_port, 147
  - set\_all\_clocks, 143
  - set\_all\_inputs, 146
  - set\_clock, 143
  - set\_input, 145
  - start, 142
  - stop, 142
  - sysex, 143
- seq64::businfo, 148
  - ~businfo, 150
  - activate, 152
  - active, 150
  - bus, 150, 152
  - busarray, 153
  - businfo, 149, 150
  - clock, 153
  - continue\_from, 153
  - deactivate, 152
  - init\_clock, 151–153
  - init\_input, 151, 152
  - initialize, 151
  - initialized, 151
  - m\_active, 154
  - m\_bus, 153
  - m\_init\_clock, 154
  - m\_init\_input, 154
  - m\_initialized, 154
  - remove, 150
  - start, 152
  - stop, 152
  - sysex, 153
- seq64::click, 154
  - button, 158

- click, 156
- is\_left, 157
- is\_middle, 157
- is\_press, 157
- is\_right, 157
- m\_button, 159
- m\_is\_press, 158
- m\_modifier, 159
- m\_x, 158
- m\_y, 159
- mod\_control, 158
- mod\_control\_shift, 158
- mod\_super, 158
- modifier, 158
- operator=, 156
- x, 157
- y, 157
- seq64::condition\_var, 159
  - condition\_var, 160
  - m\_cond, 161
  - signal, 160
  - sm\_cond, 161
  - wait, 160
- seq64::configfile, 161
  - ~configfile, 163
  - configfile, 163
  - get\_error\_message, 164
  - line\_after, 164
  - m\_d, 165
  - m\_error\_message, 165
  - m\_line, 165
  - m\_name, 165
  - next\_data\_line, 163
  - parse, 164
  - set\_error\_message, 165
  - write, 164
- seq64::editable\_event, 166
  - ~editable\_event, 171
  - analyze, 176
  - category, 173
  - category\_string, 173
  - category\_t, 169
  - channel\_string, 176
  - data\_string, 176
  - editable\_event, 170, 171
  - format\_timestamp, 175
  - m\_category, 178
  - m\_format\_timestamp, 178
  - m\_name\_category, 178
  - m\_name\_channel, 179
  - m\_name\_data, 179
  - m\_name\_meta, 178
  - m\_name\_seqspect, 179
  - m\_name\_status, 178
  - m\_name\_timestamp, 178
  - m\_parent, 178
  - meta\_string, 176
  - name\_to\_value, 172
  - operator=, 172
  - parent, 172
  - seqspec\_string, 176
  - set\_status\_from\_string, 175
  - sm\_category\_arrays, 177
  - sm\_category\_names, 177
  - sm\_channel\_event\_names, 177
  - sm\_meta\_event\_names, 177
  - sm\_prop\_event\_names, 177
  - sm\_system\_event\_names, 177
  - status\_string, 175
  - stock\_event\_string, 175
  - time\_as\_measures, 174
  - time\_as\_minutes, 174
  - time\_as\_pulses, 174
  - timestamp, 174
  - timestamp\_format\_t, 170
  - timestamp\_string, 173
  - value\_to\_name, 172
- seq64::editable\_event::name\_value\_t, 638
  - event\_name, 639
  - event\_value, 639
- seq64::editable\_events, 179
  - ~editable\_events, 182
  - add, 185
  - begin, 184
  - clear, 186
  - const\_iterator, 181
  - count, 185
  - current\_event, 186
  - dref, 184, 185
  - editable\_events, 182
  - end, 184
  - Events, 181
  - events, 184
  - EventsPair, 181
  - eventslots, 187
  - iterator, 181
  - Key, 181
  - load\_events, 183
  - m\_current\_event, 187
  - m\_events, 187
  - m\_midi\_parameters, 187
  - m\_sequence, 187
  - operator=, 183
  - remove, 186
  - replace, 186
  - save\_events, 183
  - string\_to\_pulses, 183
  - timing, 183
- seq64::event, 188
  - ~event, 193
  - append\_sysex, 202
  - cc\_match, 200
  - check\_channel, 195
  - clear\_link, 203
  - data, 205
  - decrement\_data1, 201

- decrement\_data2, 201
- event, 192
- get\_channel, 195
- get\_data, 201
- get\_linked, 203
- get\_note, 205
- get\_note\_velocity, 206
- get\_rank, 207
- get\_status, 200
- get\_sysex, 202
- get\_sysex\_size, 203
- get\_timestamp, 195
- increment\_data1, 201
- increment\_data2, 201
- is\_channel\_msg, 195
- is\_desired\_cc\_or\_not\_cc, 197
- is\_linked, 203
- is\_marked, 204
- is\_note, 206
- is\_note\_msg, 196
- is\_note\_off, 206
- is\_note\_off\_recorded, 207
- is\_note\_on, 206
- is\_one\_byte\_msg, 196
- is\_painted, 204
- is\_selected, 204
- is\_strict\_note\_msg, 197
- is\_two\_byte\_msg, 196
- link, 203
- m\_channel, 208
- m\_data, 208
- m\_has\_link, 208
- m\_linked, 208
- m\_marked, 209
- m\_painted, 209
- m\_selected, 209
- m\_status, 208
- m\_sysex, 208
- m\_sysex\_size, 208
- m\_timestamp, 207
- make\_clock, 205
- mark, 204
- mod\_timestamp, 198
- non\_cc\_match, 200
- operator<, 194
- operator=, 193
- paint, 203
- print, 207
- restart\_sysex, 202
- select, 204
- set\_channel, 199
- set\_data, 200
- set\_note, 205
- set\_note\_velocity, 206
- set\_status, 198, 199
- set\_status\_keep\_channel, 199
- set\_sysex\_size, 203
- set\_timestamp, 194
- SysexContainer, 192
- transpose\_note, 205
- unmark, 204
- unpaint, 204
- unselect, 204
- seq64::event\_list, 211
  - ~event\_list, 214
  - add, 215
  - any\_selected\_notes, 220
  - append, 216
  - begin, 215
  - clear, 217
  - clear\_links, 219
  - const\_iterator, 214
  - count, 215
  - count\_selected\_events, 221
  - count\_selected\_notes, 220
  - dref, 218
  - editable\_events, 222
  - empty, 215
  - end, 215
  - event\_list, 214
  - Events, 213
  - events, 221
  - EventsPair, 213
  - is\_modified, 216
  - iterator, 214
  - link\_new, 218
  - m\_events, 222
  - m\_is\_modified, 222
  - mark\_all, 220
  - mark\_out\_of\_range, 219
  - mark\_selected, 219
  - merge, 217
  - midi\_container, 222
  - midi\_splitter, 222
  - midifile, 222
  - operator=, 214
  - print, 221
  - push\_back, 216
  - remove, 217
  - remove\_marked, 220
  - select\_all, 221
  - sequence, 222
  - sort, 218
  - unmark\_all, 220
  - unmodify, 217
  - unpaint\_all, 220
  - unselect\_all, 221
  - verify\_and\_link, 219
- seq64::event\_list::event\_key, 209
  - event\_key, 210
  - m\_rank, 211
  - m\_timestamp, 211
  - operator<, 210
- seq64::eventedit, 223
  - ~eventedit, 228
  - change\_focus, 232

close\_out, 232  
enqueue\_draw, 229  
eventedit, 227  
eventslots, 235  
handle\_cancel, 233  
handle\_close, 232  
handle\_delete, 232  
handle\_insert, 232  
handle\_modify, 233  
handle\_save, 233  
m\_bottbox, 236  
m\_button\_cancel, 237  
m\_button\_del, 236  
m\_button\_ins, 236  
m\_button\_modify, 236  
m\_button\_save, 237  
m\_editbox, 236  
m\_entry\_ev\_data\_0, 238  
m\_entry\_ev\_data\_1, 238  
m\_entry\_ev\_name, 238  
m\_entry\_ev\_timestamp, 238  
m\_eventslots, 235  
m\_have\_focus, 239  
m\_htopbox, 235  
m\_label\_category, 238  
m\_label\_channel, 237  
m\_label\_ev\_count, 237  
m\_label\_modified, 238  
m\_label\_ppqn, 237  
m\_label\_right, 238  
m\_label\_seq\_name, 237  
m\_label\_spacer, 237  
m\_label\_time\_fmt, 238  
m\_label\_time\_sig, 237  
m\_optsbox, 236  
m\_rightbox, 236  
m\_seq, 239  
m\_showbox, 236  
m\_table, 235  
m\_vadjust, 235  
m\_vscroll, 235  
on\_delete\_event, 234  
on\_focus\_in\_event, 234  
on\_focus\_out\_event, 234  
on\_key\_press\_event, 234  
on\_realize, 233  
on\_set\_focus, 233  
perf\_modify, 231  
set\_dirty, 231  
set\_event\_category, 230  
set\_event\_data\_0, 230  
set\_event\_data\_1, 230  
set\_event\_name, 230  
set\_event\_timestamp, 230  
set\_seq\_count, 229  
set\_seq\_ppqn, 229  
set\_seq\_time\_sig, 229  
set\_seq\_title, 229  
v\_adjustment, 231  
seq64::eventslots, 239  
~eventslots, 244  
change\_vert, 250  
convert\_y, 249  
current\_index, 244  
decrement\_bottom, 252  
decrement\_current, 252  
decrement\_top, 251  
delete\_current\_event, 247  
draw\_event, 249  
draw\_events, 250  
enqueue\_draw, 249  
event\_count, 244  
eventedit, 255  
eventslots, 243  
increment\_bottom, 253  
increment\_current, 252  
increment\_top, 252  
insert\_event, 245, 246  
line\_count, 244  
line\_increment, 244  
line\_maximum, 244  
load\_events, 245  
m\_bottom\_iterator, 257  
m\_char\_w, 256  
m\_current\_index, 257  
m\_current\_iterator, 257  
m\_event\_container, 255  
m\_event\_count, 256  
m\_line\_count, 256  
m\_line\_maximum, 256  
m\_line\_overlap, 257  
m\_pager\_index, 257  
m\_parent, 255  
m\_seq, 255  
m\_setbox\_w, 256  
m\_slots\_chars, 255  
m\_slots\_x, 256  
m\_slots\_y, 256  
m\_top\_index, 257  
m\_top\_iterator, 257  
modify\_current\_event, 247  
on\_button\_press\_event, 253  
on\_button\_release\_event, 253  
on\_expose\_event, 253  
on\_focus\_in\_event, 253  
on\_focus\_out\_event, 254  
on\_frame\_down, 254  
on\_frame\_end, 255  
on\_frame\_home, 255  
on\_frame\_up, 254  
on\_move\_down, 254  
on\_move\_up, 254  
on\_realize, 253  
on\_scroll\_event, 254  
on\_size\_allocate, 254  
page\_movement, 251

- page\_topper, 251
- pager\_index, 245
- save\_events, 248
- select\_event, 248
- set\_current\_event, 245
- set\_text, 249
- top\_index, 244
- seq64::font, 258
  - char\_height, 261
  - char\_width, 261
  - Color, 259
  - font, 259
  - init, 260
  - m\_b\_on\_c\_pixmap, 263
  - m\_b\_on\_y\_pixmap, 262
  - m\_black\_pixmap, 262
  - m\_c\_on\_b\_pixmap, 263
  - m\_cell\_h, 261
  - m\_cell\_w, 261
  - m\_clip\_mask, 263
  - m\_font\_h, 261
  - m\_font\_w, 261
  - m\_offset, 262
  - m\_padded\_h, 262
  - m\_pixmap, 262
  - m\_use\_new\_font, 261
  - m\_white\_pixmap, 262
  - m\_y\_on\_b\_pixmap, 262
  - padded\_height, 261
  - render\_string\_on\_drawable, 260
- seq64::gui\_assistant, 274
  - ~gui\_assistant, 276
  - gui\_assistant, 276
  - jack\_idle\_connect, 276
  - keys, 277
  - lash\_timeout\_connect, 276
  - m\_keys\_perform, 277
  - quit, 276
- seq64::gui\_assistant\_gtk2, 277
  - ~gui\_assistant\_gtk2, 279
  - gui\_assistant\_gtk2, 278
  - jack\_idle\_connect, 279
  - lash\_timeout\_connect, 279
  - quit, 279
  - sm\_internal\_keys, 279
- seq64::gui\_drawingarea\_gtk2, 280
  - ~gui\_drawingarea\_gtk2, 284
  - clear\_window, 285
  - current\_x, 284
  - current\_y, 284
  - draw\_drawable, 294
  - draw\_line, 286–288
  - draw\_line\_on\_pixmap, 286, 287
  - draw\_normal\_rectangle\_on\_pixmap, 293
  - draw\_rectangle, 290–292
  - draw\_rectangle\_on\_pixmap, 292, 293
  - drop\_x, 285
  - drop\_y, 285
  - force\_draw, 285
  - gtk\_drawarea\_init, 295
  - gui\_drawingarea\_gtk2, 283
  - m\_background, 296
  - m\_current\_x, 297
  - m\_current\_y, 297
  - m\_drop\_x, 297
  - m\_drop\_y, 298
  - m\_foreground, 297
  - m\_gc, 296
  - m\_hadjust, 296
  - m\_mainperf, 297
  - m\_pixmap, 296
  - m\_vadjust, 296
  - m\_window, 296
  - m\_window\_x, 297
  - m\_window\_y, 297
  - on\_realize, 296
  - operator=, 284
  - perf, 285
  - render\_string, 289
  - render\_string\_on\_pixmap, 289
  - scroll\_hadjust, 294
  - scroll\_hset, 295
  - scroll\_vadjust, 294
  - scroll\_vset, 295
  - set\_current\_drop\_x, 295
  - set\_current\_drop\_y, 295
  - set\_line, 285
  - window\_x, 284
  - window\_y, 284
- seq64::gui\_drawingarea\_gtk2::rect, 818
  - height, 818
  - width, 819
  - x, 818
  - y, 818
- seq64::gui\_palette\_gtk2, 298
  - ~gui\_palette\_gtk2, 302
  - bg\_color, 305, 306
  - black, 303
  - black\_key, 305
  - black\_paint, 305
  - blue, 305
  - Color, 301
  - dark\_blue, 303
  - dark\_cyan, 304
  - dark\_green, 303
  - dark\_grey, 304
  - dark\_magenta, 304
  - dark\_orange, 303
  - dark\_red, 303
  - fg\_color, 306
  - green, 305
  - grey, 304
  - gui\_palette\_gtk2, 302
  - is\_inverse, 303
  - light\_grey, 304
  - line\_color, 303

- load\_inverse\_palette, 302
- m\_bg\_color, 309
- m\_black, 306
- m\_blk\_key, 308
- m\_blk\_paint, 308
- m\_blue, 308
- m\_dk\_blue, 307
- m\_dk\_cyan, 307
- m\_dk\_green, 306
- m\_dk\_grey, 308
- m\_dk\_magenta, 307
- m\_dk\_orange, 307
- m\_dk\_red, 306
- m\_fg\_color, 309
- m\_green, 308
- m\_grey, 308
- m\_is\_inverse, 306
- m\_line\_color, 309
- m\_lt\_grey, 308
- m\_orange, 307
- m\_progress\_color, 309
- m\_red, 307
- m\_white, 307
- m\_wht\_key, 309
- m\_wht\_paint, 308
- m\_yellow, 307
- orange, 304
- progress\_color, 303
- red, 304
- white, 304
- white\_key, 305
- white\_paint, 305
- yellow, 305
- seq64::gui\_window\_gtk2, 310
  - ~gui\_window\_gtk2, 312
  - gui\_window\_gtk2, 311
  - is\_realized, 312
  - m\_is\_realized, 314
  - m\_mainperf, 314
  - m\_redraw\_period\_ms, 314
  - m\_window\_x, 314
  - m\_window\_y, 314
  - on\_realize, 313
  - perf, 312
  - quit, 312
  - redraw\_period\_ms, 312
  - scroll\_hadjust, 312
  - scroll\_hset, 313
  - scroll\_vadjust, 313
  - scroll\_vset, 313
- seq64::jack\_assistant, 314
  - ~jack\_assistant, 318
  - activate, 322
  - client, 327
  - client\_name, 327
  - client\_open, 327
  - client\_uuid, 327
  - deinit, 321
  - error\_message, 330
  - get\_beat\_width, 319
  - get\_beats\_per\_measure, 320
  - get\_beats\_per\_minute, 320
  - get\_current\_jack\_position, 334
  - get\_follow\_transport, 326
  - get\_jack\_client\_info, 328
  - get\_jack\_mode, 325
  - get\_jack\_pos, 325
  - get\_jack\_stop\_tick, 325
  - get\_jack\_tick, 325
  - get\_ppqn, 319
  - info\_message, 330
  - init, 321
  - is\_master, 319
  - is\_running, 319
  - jack\_assistant, 318
  - jack\_frame\_rate, 326
  - jack\_session\_callback, 334
  - jack\_shutdown\_callback, 332
  - jack\_sync\_callback, 332
  - jack\_timebase\_callback, 333
  - jack\_transport\_callback, 332
  - m\_beat\_width, 337
  - m\_beats\_per\_measure, 337
  - m\_beats\_per\_minute, 338
  - m\_follow\_transport, 337
  - m\_jack\_client, 335
  - m\_jack\_client\_name, 335
  - m\_jack\_client\_uuid, 335
  - m\_jack\_frame\_current, 335
  - m\_jack\_frame\_last, 335
  - m\_jack\_frame\_rate, 337
  - m\_jack\_master, 336
  - m\_jack\_parent, 335
  - m\_jack\_pos, 336
  - m\_jack\_running, 336
  - m\_jack\_stop\_tick, 337
  - m\_jack\_tick, 336
  - m\_jack\_transport\_state, 336
  - m\_jack\_transport\_state\_last, 336
  - m\_session\_ev, 336
  - m\_ppqn, 337
  - m\_toggle\_jack, 337
  - output, 324
  - parent, 319
  - position, 323
  - session\_event, 322
  - set\_beat\_width, 319
  - set\_beats\_per\_measure, 320
  - set\_beats\_per\_minute, 320
  - set\_follow\_transport, 326
  - set\_jack\_mode, 325
  - set\_jack\_running, 327
  - set\_jack\_stop\_tick, 325
  - set\_position, 330
  - set\_ppqn, 324
  - set\_start\_from\_perfedit, 326

- show\_position, 328
- sm\_status\_pairs, 335
- song\_start\_mode, 326
- start, 322
- stop, 322
- sync, 329
- tick\_multiplier, 327
- toggle\_follow\_transport, 326
- toggle\_jack\_mode, 325
- toggle\_song\_start\_mode, 326
- transport\_not\_starting, 320
- transport\_state, 320
- seq64::jack\_scratchpad, 338
  - js\_clock\_tick, 339
  - js\_current\_tick, 339
  - js\_delta\_tick\_frac, 340
  - js\_dumping, 339
  - js\_init\_clock, 339
  - js\_jack\_stopped, 339
  - js\_looping, 339
  - js\_playback\_mode, 339
  - js\_ticks\_converted, 340
  - js\_ticks\_converted\_last, 340
  - js\_ticks\_delta, 340
  - js\_total\_tick, 339
- seq64::jack\_status\_pair\_t, 340
  - jf\_bit, 340
  - jf\_meaning, 341
- seq64::keybindentry, 341
  - keybindentry, 342
  - m\_key, 343
  - m\_perf, 344
  - m\_slot, 344
  - m\_type, 343
  - on\_key\_press\_event, 343
  - options, 343
  - set, 343
  - type, 342
- seq64::keys\_perform, 344
  - ~keys\_perform, 352
  - at\_bpm\_dn, 365
  - at\_bpm\_up, 365
  - at\_event\_edit, 368
  - at\_fast\_forward, 368
  - at\_follow\_transport, 368
  - at\_group\_learn, 367
  - at\_group\_off, 367
  - at\_group\_on, 367
  - at\_keep\_queue, 366
  - at\_menu\_mode, 367
  - at\_pattern\_edit, 368
  - at\_pause, 367
  - at\_pointer\_position, 368
  - at\_queue, 366
  - at\_replace, 366
  - at\_rewind, 368
  - at\_screensset\_dn, 366
  - at\_screensset\_up, 366
  - at\_set\_playing\_screensset, 366
  - at\_show\_ui\_sequence\_key, 369
  - at\_show\_ui\_sequence\_number, 369
  - at\_snapshot\_1, 366
  - at\_snapshot\_2, 366
  - at\_song\_mode, 367
  - at\_start, 367
  - at\_stop, 369
  - at\_tap\_bpm, 368
  - at\_toggle\_jack, 367
  - at\_toggle\_mutes, 368
  - bpm\_dn, 353
  - bpm\_up, 352
  - event\_edit, 358
  - fast\_forward, 360
  - follow\_transport, 360
  - get\_key\_events, 362
  - get\_key\_events\_rev, 363
  - get\_key\_groups, 362
  - get\_key\_groups\_rev, 363
  - get\_keys, 352
  - group\_learn, 357
  - group\_off, 356, 357
  - group\_on, 356
  - keep\_queue, 354
  - key\_name, 364
  - keys\_perform, 351
  - lookup\_keyevent\_key, 363
  - lookup\_keyevent\_seq, 363
  - lookup\_keygroup\_group, 364
  - lookup\_keygroup\_key, 363
  - m\_key\_bpm\_dn, 370
  - m\_key\_bpm\_up, 370
  - m\_key\_event\_edit, 373
  - m\_key\_events, 370
  - m\_key\_events\_rev, 370
  - m\_key\_fast\_forward, 373
  - m\_key\_follow\_transport, 373
  - m\_key\_group\_learn, 372
  - m\_key\_group\_off, 372
  - m\_key\_group\_on, 372
  - m\_key\_groups, 370
  - m\_key\_groups\_rev, 370
  - m\_key\_keep\_queue, 371
  - m\_key\_menu\_mode, 372
  - m\_key\_pattern\_edit, 373
  - m\_key\_pause, 372
  - m\_key\_pointer\_position, 373
  - m\_key\_queue, 371
  - m\_key\_replace, 371
  - m\_key\_rewind, 373
  - m\_key\_screensset\_dn, 371
  - m\_key\_screensset\_up, 371
  - m\_key\_set\_playing\_screensset, 371
  - m\_key\_show\_ui\_sequence\_key, 370
  - m\_key\_show\_ui\_sequence\_number, 370
  - m\_key\_snapshot\_1, 371
  - m\_key\_snapshot\_2, 371



- m\_key\_song\_mode, 372
- m\_key\_start, 372
- m\_key\_stop, 374
- m\_key\_tap\_bpm, 373
- m\_key\_toggle\_jack, 372
- m\_key\_toggle\_mutes, 373
- menu\_mode, 359
- options, 369
- optionsfile, 369
- pattern\_edit, 358
- pause, 358
- perform, 369
- pointer\_position, 360, 361
- queue, 353, 354
- replace, 353
- RevSlotMap, 351
- rewind, 360
- screenset\_dn, 355
- screenset\_up, 355
- set\_all\_key\_events, 364
- set\_all\_key\_groups, 364
- set\_key\_event, 365
- set\_key\_group, 365
- set\_keys, 352
- set\_playing\_screenset, 356
- show\_ui\_sequence\_key, 362
- show\_ui\_sequence\_number, 362
- SlotMap, 351
- snapshot\_1, 354
- snapshot\_2, 355
- song\_mode, 359
- start, 357
- stop, 359
- tap\_bpm, 361
- toggle\_jack, 361
- toggle\_mutes, 361
- seq64::keys\_perform\_gtk2, 374
  - ~keys\_perform\_gtk2, 376
  - key\_name, 376
  - keys\_perform\_gtk2, 376
  - set\_all\_key\_events, 376
  - set\_all\_key\_groups, 377
- seq64::keys\_perform\_transfer, 377
  - kpt\_bpm\_dn, 378
  - kpt\_bpm\_up, 378
  - kpt\_event\_edit, 380
  - kpt\_fast\_forward, 381
  - kpt\_follow\_transport, 381
  - kpt\_group\_learn, 378
  - kpt\_group\_off, 378
  - kpt\_group\_on, 378
  - kpt\_keep\_queue, 379
  - kpt\_menu\_mode, 380
  - kpt\_pattern\_edit, 380
  - kpt\_pause, 380
  - kpt\_pointer\_position, 381
  - kpt\_queue, 379
  - kpt\_replace, 379
  - kpt\_rewind, 381
  - kpt\_screenset\_dn, 378
  - kpt\_screenset\_up, 378
  - kpt\_set\_playing\_screenset, 378
  - kpt\_show\_ui\_sequence\_key, 379
  - kpt\_show\_ui\_sequence\_number, 380
  - kpt\_snapshot\_1, 379
  - kpt\_snapshot\_2, 379
  - kpt\_song\_mode, 380
  - kpt\_start, 379
  - kpt\_stop, 379
  - kpt\_tap\_bpm, 380
  - kpt\_toggle\_jack, 380
  - kpt\_toggle\_mutes, 381
- seq64::keystroke, 381
  - is, 384
  - is\_delete, 384
  - is\_letter, 384
  - is\_press, 384
  - key, 384
  - keystroke, 382, 383
  - m\_is\_press, 386
  - m\_key, 386
  - m\_modifier, 386
  - mod\_control, 385
  - mod\_control\_shift, 385
  - mod\_super, 385
  - modifier, 385
  - operator=, 383
  - shift\_lock, 385
- seq64::lash, 386
  - handle\_config, 388
  - handle\_event, 388
  - init, 388
  - lash, 387
  - m\_client, 389
  - m\_is\_lash\_supported, 389
  - m\_lash\_args, 389
  - m\_perform, 389
  - process\_events, 388
  - set\_alsa\_client\_id, 387
  - start, 387
- seq64::lfownd, 390
  - ~lfownd, 392
  - lfownd, 392
  - m\_hbox, 393
  - m\_phase, 394
  - m\_range, 394
  - m\_scale\_phase, 393
  - m\_scale\_range, 393
  - m\_scale\_speed, 393
  - m\_scale\_value, 393
  - m\_scale\_wave, 393
  - m\_seq, 393
  - m\_seqdata, 393
  - m\_speed, 394
  - m\_value, 394
  - m\_wave, 394

- m\_wave\_name, 394
- on\_focus\_out\_event, 392
- scale\_lfo\_change, 392
- toggle\_visible, 392
- seq64::maintime, 395
  - ~maintime, 397
  - idle\_progress, 397
  - m\_bar\_width, 398
  - m\_beat\_width, 398
  - m\_box\_height, 399
  - m\_box\_less\_pill, 399
  - m\_box\_width, 399
  - m\_flash\_height, 399
  - m\_flash\_width, 399
  - m\_flash\_x, 399
  - m\_pill\_width, 399
  - m\_ppqn, 400
  - m\_tick, 400
  - maintime, 397
  - mainwnd, 398
  - on\_expose\_event, 398
  - on\_realize, 398
  - operator=, 397
- seq64::mainwid, 400
  - ~mainwid, 404
  - calculate\_base\_sizes, 409
  - draw\_marker\_on\_sequence, 406
  - draw\_pixmap\_on\_window, 405
  - draw\_sequence\_on\_pixmap, 407
  - draw\_sequence\_pixmap\_on\_window, 408
  - draw\_sequences\_on\_pixmap, 408
  - fill\_background\_window, 405
  - m\_armed\_progress\_color, 412
  - m\_button\_down, 412
  - m\_last\_playing, 413
  - m\_last\_tick\_x, 413
  - m\_mainwid\_border, 414
  - m\_mainwid\_spacing, 414
  - m\_mainwid\_x, 414
  - m\_mainwid\_y, 414
  - m\_mainwnd\_cols, 413
  - m\_mainwnd\_rows, 413
  - m\_max\_sets, 414
  - m\_moving, 412
  - m\_moving\_seq, 412
  - m\_old\_seq, 412
  - m\_progress\_height, 415
  - m\_screenset, 413
  - m\_screenset\_offset, 415
  - m\_screenset\_slots, 415
  - m\_seqarea\_seq\_x, 413
  - m\_seqarea\_seq\_y, 414
  - m\_seqarea\_x, 413
  - m\_seqarea\_y, 413
  - m\_text\_size\_x, 414
  - m\_text\_size\_y, 414
  - mainwid, 404
  - mainwnd, 412
  - on\_button\_press\_event, 410
  - on\_button\_release\_event, 410
  - on\_expose\_event, 409
  - on\_focus\_in\_event, 411
  - on\_focus\_out\_event, 411
  - on\_motion\_notify\_event, 411
  - on\_realize, 409
  - redraw, 406
  - reset, 405
  - select\_fg\_bg\_colors, 409
  - seq\_from\_xy, 408
  - seq\_set\_and\_edit, 406
  - seq\_set\_and\_eventedit, 406
  - set\_screenset, 405
  - timeout, 408
  - update\_mainwid\_sequences, 412
  - update\_markers, 407
  - update\_sequences\_on\_window, 405
  - valid\_sequence, 407
- seq64::mainwnd, 415
  - ~mainwnd, 421
  - about\_dialog, 426
  - adj\_callback\_bpm, 422
  - adj\_callback\_ss, 422
  - apply\_song\_transpose, 425
  - build\_info\_dialog, 426
  - choose\_file, 428
  - edit\_callback\_notepad, 422
  - edit\_field\_has\_focus, 429
  - enregister\_perfedits, 424
  - file\_exit, 428
  - file\_import\_dialog, 426
  - file\_new, 425
  - file\_open, 425
  - file\_save, 425
  - file\_save\_as, 427
  - handle\_signal, 422
  - install\_signal\_handlers, 428
  - is\_save, 428
  - jack\_dialog, 426
  - learn\_toggle, 424
  - m\_adjust\_bpm, 433
  - m\_adjust\_load\_offset, 433
  - m\_adjust\_ss, 433
  - m\_button\_jack, 433
  - m\_button\_learn, 432
  - m\_button\_perfedits, 432
  - m\_button\_play, 432
  - m\_button\_stop, 432
  - m\_call\_seq\_edit, 434
  - m\_call\_seq\_eventedit, 434
  - m\_entry\_notes, 434
  - m\_image\_play, 432
  - m\_is\_running, 434
  - m\_main\_cursor, 432
  - m\_main\_time, 431
  - m\_main\_wid, 431
  - m\_menu\_edit, 430

- m\_menu\_file, 430
- m\_menu\_help, 431
- m\_menu\_mode, 434
- m\_menu\_view, 431
- m\_menubar, 430
- m\_options, 432
- m\_perf\_edit, 431
- m\_perf\_edit\_2, 431
- m\_ppqn, 431
- m\_sigpipe, 430
- m\_spinbutton\_bpm, 433
- m\_spinbutton\_load\_offset, 433
- m\_spinbutton\_ss, 433
- m\_timeout\_connect, 434
- m\_tooltips, 430
- mainwnd, 420
- new\_file, 428
- new\_open\_error\_dialog, 427
- on\_delete\_event, 429
- on\_grouplearncchange, 430
- on\_key\_press\_event, 429
- on\_key\_release\_event, 429
- on\_realize, 429
- open\_file, 421
- open\_performance\_edit, 424
- open\_performance\_edit\_2, 424
- options\_dialog, 426
- pause\_playing, 423
- ppqn, 422
- query\_save\_changes, 427
- rc\_error\_dialog, 421
- save\_file, 428
- sequence\_key, 425
- set\_play\_image, 423
- set\_song\_mute, 426
- set\_sONGLIVE\_image, 423
- signal\_action, 428
- start\_playing, 423
- stop\_playing, 424
- timer\_callback, 422
- toLower, 425
- toggle\_playing, 424
- update\_window\_title, 425
- seq64::mastermidibase, 435
  - ~mastermidibase, 439
  - activate, 450
  - api\_clock, 452
  - api\_continue\_from, 451
  - api\_flush, 452
  - api\_get\_midi\_event, 452
  - api\_init, 450
  - api\_init\_clock, 451
  - api\_is\_more\_input, 452
  - api\_poll\_for\_midi, 452
  - api\_port\_start, 452
  - api\_set\_beats\_per\_minute, 451
  - api\_set\_ppqn, 451
  - api\_start, 451
  - api\_stop, 451
  - clock, 450
  - continue\_from, 442
  - dump\_midi\_input, 444
  - filter\_by\_channel, 440
  - flush, 444
  - get\_beats\_per\_minute, 440
  - get\_bpm, 440
  - get\_clock, 448
  - get\_input, 447
  - get\_midi\_event, 446
  - get\_midi\_in\_bus\_name, 445
  - get\_midi\_out\_bus\_name, 445
  - get\_num\_in\_buses, 440
  - get\_num\_out\_buses, 440
  - get\_ppqn, 440
  - get\_sequence, 441
  - init, 439
  - init\_clock, 443
  - initialize\_buses, 445
  - input, 450
  - is\_dumping, 440
  - is\_input\_system\_port, 448
  - is\_more\_input, 446
  - m\_beats\_per\_minute, 455
  - m\_bus\_announce, 454
  - m\_dumping\_input, 455
  - m\_filter\_by\_channel, 455
  - m\_inbus\_array, 454
  - m\_master\_clocks, 454
  - m\_master\_inputs, 454
  - m\_max\_busses, 454
  - m\_mutex, 455
  - m\_outbus\_array, 454
  - m\_ppqn, 455
  - m\_queue, 454
  - m\_seq, 455
  - m\_vector\_sequence, 455
  - mastermidibase, 439
  - midi\_alsa\_info, 453
  - perform, 453
  - play, 442
  - poll\_for\_midi, 446
  - port\_exit, 441
  - port\_settings, 450
  - port\_start, 441
  - print, 444
  - save\_clock, 453
  - save\_input, 453
  - set\_beats\_per\_minute, 450
  - set\_clock, 443, 446
  - set\_input, 447
  - set\_ppqn, 448
  - set\_sequence\_input, 444
  - start, 441
  - stop, 441
  - sysex, 443
- seq64::mastermidibus, 456

- ~mastermidibus, 459
- activate, 463
- api\_continue\_from, 463
- api\_flush, 462, 464
- api\_get\_midi\_event, 460, 463
- api\_init, 460, 464
- api\_is\_more\_input, 460, 463
- api\_poll\_for\_midi, 460, 463
- api\_port\_start, 463, 464
- api\_set\_beats\_per\_minute, 462, 464
- api\_set\_ppqn, 462, 464
- api\_start, 462
- api\_stop, 462
- m\_alsa\_seq, 465
- m\_midi\_master, 465
- m\_num\_poll\_descriptors, 465
- m\_poll\_descriptors, 465
- m\_use\_jack\_polling, 466
- mastermidibus, 459
- midi\_alsa\_info, 465
- midi\_jack\_info, 465
- port\_list, 464
- seq64::midi\_alsa, 466
  - ~midi\_alsa, 469
  - api\_clock, 473
  - api\_continue\_from, 473
  - api\_deinit\_in, 471
  - api\_flush, 473
  - api\_get\_midi\_event, 471
  - api\_init\_in, 470
  - api\_init\_in\_sub, 471
  - api\_init\_out, 470
  - api\_init\_out\_sub, 470
  - api\_play, 472
  - api\_poll\_for\_midi, 471
  - api\_set\_beats\_per\_minute, 474
  - api\_set\_ppqn, 474
  - api\_start, 473
  - api\_stop, 473
  - api\_sysex, 472
  - get\_client, 469
  - get\_port, 470
  - m\_dest\_addr\_client, 475
  - m\_dest\_addr\_port, 475
  - m\_input\_port\_name, 475
  - m\_local\_addr\_client, 475
  - m\_local\_addr\_port, 475
  - m\_seq, 475
  - midi\_alsa, 469
  - set\_virtual\_name, 474
- seq64::midi\_alsa\_info, 476
  - ~midi\_alsa\_info, 478
  - api\_flush, 480
  - api\_get\_midi\_event, 478
  - api\_poll\_for\_midi, 479
  - api\_port\_start, 479
  - api\_set\_beats\_per\_minute, 479
  - api\_set\_ppqn, 479
  - get\_all\_port\_info, 480
  - m\_alsa\_seq, 481
  - m\_num\_poll\_descriptors, 481
  - m\_poll\_descriptors, 481
  - midi\_alsa\_info, 477
  - seq, 478
  - sm\_input\_caps, 481
  - sm\_output\_caps, 481
- seq64::midi\_api, 482
  - ~midi\_api, 484
  - api\_clock, 488
  - api\_connect, 485
  - api\_continue\_from, 487
  - api\_deinit\_in, 486
  - api\_flush, 487
  - api\_get\_bus\_name, 488
  - api\_get\_midi\_event, 486
  - api\_get\_port\_name, 488
  - api\_init\_in, 485
  - api\_init\_in\_sub, 486
  - api\_init\_out, 485
  - api\_init\_out\_sub, 485
  - api\_play, 486
  - api\_poll\_for\_midi, 486
  - api\_set\_beats\_per\_minute, 488
  - api\_set\_ppqn, 488
  - api\_start, 487
  - api\_stop, 487
  - api\_sysex, 487
  - cancel\_callback, 490
  - error, 489
  - input\_data, 490
  - is\_input\_port, 485
  - is\_port\_open, 488
  - is\_system\_port, 485
  - is\_virtual\_port, 485
  - m\_connected, 491
  - m\_error\_callback, 491
  - m\_error\_callback\_user\_data, 491
  - m\_error\_string, 491
  - m\_first\_error\_occurred, 491
  - m\_input\_data, 491
  - m\_master\_info, 490
  - m\_parent\_bus, 490
  - master\_info, 489
  - master\_midi\_mode, 489
  - midi\_api, 484
  - parent\_bus, 489
  - set\_port\_open, 490
  - user\_callback, 490
- seq64::midi\_container, 492
  - ~midi\_container, 494
  - add\_event, 497
  - add\_long, 497
  - add\_short, 497
  - add\_variable, 496
  - clear, 496
  - done, 495

- fill, 494
- fill\_meta\_track\_end, 498
- fill\_proprietary, 498
- fill\_seq\_name, 498
- fill\_seq\_number, 497
- fill\_time\_sig\_and\_tempo, 498
- get, 496
- m\_position\_for\_get, 501
- m\_sequence, 501
- midi\_container, 494
- midifile, 500
- position, 496
- position\_increment, 496
- position\_reset, 496
- put, 495
- size, 495
- song\_fill\_seq\_event, 500
- song\_fill\_seq\_trigger, 500
- seq64::midi\_control, 501
  - action, 502
  - active, 503
  - data, 503
  - in\_range, 505
  - inverse\_active, 503
  - m\_active, 505
  - m\_data, 505
  - m\_inverse\_active, 505
  - m\_max\_value, 506
  - m\_min\_value, 505
  - m\_status, 505
  - match, 504
  - max\_value, 504
  - midi\_control, 503
  - min\_value, 504
  - set, 504
  - status, 503
- seq64::midi\_in\_alsa, 506
  - midi\_in\_alsa, 508
- seq64::midi\_in\_jack, 508
  - ~midi\_in\_jack, 510
  - api\_get\_midi\_event, 511
  - api\_poll\_for\_midi, 510
  - m\_client\_name, 511
  - midi\_in\_jack, 510
  - open\_client, 511
- seq64::midi\_info, 512
  - ~midi\_info, 515
  - add\_bus, 520
  - api\_connect, 517
  - api\_flush, 517
  - api\_get\_midi\_event, 517
  - api\_poll\_for\_midi, 517
  - api\_port\_start, 517
  - api\_set\_beats\_per\_minute, 516
  - api\_set\_ppqn, 516
  - app\_name, 516
  - bpm, 516
  - bus\_container, 520
  - clear, 516
  - connect\_name, 519
  - error, 519
  - full\_port\_count, 516
  - get\_all\_port\_info, 520
  - get\_bus\_id, 518
  - get\_bus\_name, 518
  - get\_input, 518
  - get\_port\_count, 517
  - get\_port\_id, 518
  - get\_port\_name, 518
  - get\_system, 518
  - get\_virtual, 518
  - global\_queue, 519, 520
  - input\_ports, 515
  - m\_app\_name, 522
  - m\_bpm, 522
  - m\_bus\_container, 521
  - m\_error\_string, 522
  - m\_global\_queue, 522
  - m\_input, 521
  - m\_midi\_handle, 522
  - m\_midi\_mode\_input, 521
  - m\_output, 521
  - m\_ppqn, 522
  - midi\_handle, 515, 520
  - midi\_info, 515
  - midi\_mode, 515
  - nc\_midi\_port\_info, 520
  - output\_ports, 515
  - port\_list, 519
  - ppqn, 516
  - queue\_number, 519
  - ref\_midi\_port\_info, 521
  - rtmidi\_info, 521
- seq64::midi\_jack, 523
  - ~midi\_jack, 527
  - api\_clock, 534
  - api\_connect, 530
  - api\_continue\_from, 533
  - api\_deinit\_in, 532
  - api\_flush, 533
  - api\_get\_midi\_event, 532
  - api\_get\_port\_name, 535
  - api\_init\_in, 531
  - api\_init\_in\_sub, 532
  - api\_init\_out, 530
  - api\_init\_out\_sub, 531
  - api\_play, 533
  - api\_poll\_for\_midi, 532
  - api\_set\_beats\_per\_minute, 534
  - api\_set\_ppqn, 534
  - api\_start, 534
  - api\_stop, 534
  - api\_sysex, 533
  - client\_handle, 527, 528
  - close\_client, 529
  - close\_port, 529

- connect\_port, 529
- create\_ringbuffer, 529
- jack\_data, 527
- m\_jack\_data, 536
- m\_jack\_info, 536
- m\_multi\_client, 536
- m\_remote\_port\_name, 536
- midi\_jack, 525
- midi\_jack\_info, 535
- multi\_client, 527
- open\_client, 530
- open\_client\_impl, 528
- port\_handle, 528
- register\_port, 530
- remote\_port\_name, 527
- set\_virtual\_name, 535
- seq64::midi\_jack\_data, 536
  - ~midi\_jack\_data, 537
  - m\_jack\_buffmessage, 538
  - m\_jack\_buffsize, 538
  - m\_jack\_client, 538
  - m\_jack\_lasttime, 538
  - m\_jack\_port, 538
  - m\_jack\_rtmidiin, 538
  - midi\_jack\_data, 537
  - valid\_buffer, 537
- seq64::midi\_jack\_info, 539
  - ~midi\_jack\_info, 541
  - add, 545
  - api\_connect, 542
  - api\_flush, 543
  - api\_get\_midi\_event, 542
  - api\_poll\_for\_midi, 542
  - api\_port\_start, 543
  - api\_set\_beats\_per\_minute, 543
  - api\_set\_ppqn, 542
  - client\_handle, 541, 544
  - connect, 545
  - disconnect, 545
  - extract\_names, 545
  - get\_all\_port\_info, 544
  - jack\_process\_io, 545
  - m\_jack\_client, 546
  - m\_jack\_ports, 546
  - m\_multi\_client, 546
  - midi\_jack, 545
  - midi\_jack\_info, 541
  - multi\_client, 541
- seq64::midi\_list, 546
  - ~midi\_list, 549
  - CharList, 548
  - clear, 549
  - done, 549
  - get, 549
  - m\_char\_list, 550
  - midi\_list, 548
  - put, 549
  - size, 549
- seq64::midi\_measures, 550
  - beats, 552
  - divisions, 552
  - m\_beats, 553
  - m\_divisions, 553
  - m\_measures, 552
  - measures, 551
  - midi\_measures, 551
- seq64::midi\_message, 553
  - array, 555
  - at, 554
  - container, 554
  - count, 555
  - empty, 555
  - m\_bytes, 555
  - m\_timestamp, 556
  - midi\_message, 554
  - operator[], 554
  - push, 555
  - timestamp, 555
- seq64::midi\_out\_alsa, 556
  - midi\_out\_alsa, 558
- seq64::midi\_out\_jack, 558
  - ~midi\_out\_jack, 560
  - midi\_out\_jack, 560
  - open\_client, 561
  - send\_message, 560
- seq64::midi\_port\_info, 561
  - add, 562
  - clear, 563
  - connect\_name, 564
  - get\_bus\_id, 563
  - get\_bus\_name, 563
  - get\_input, 563
  - get\_port\_count, 563
  - get\_port\_id, 563
  - get\_port\_name, 563
  - get\_queue\_number, 564
  - get\_system, 564
  - get\_virtual, 564
  - m\_port\_container, 564
  - m\_port\_count, 564
  - midi\_port\_info, 562
- seq64::midi\_port\_info::port\_info\_t, 793
  - m\_client\_name, 794
  - m\_client\_number, 794
  - m\_is\_input, 795
  - m\_is\_system, 795
  - m\_is\_virtual, 795
  - m\_port\_name, 794
  - m\_port\_number, 794
  - m\_queue\_number, 795
- seq64::midi\_queue, 565
  - ~midi\_queue, 566
  - add, 566
  - allocate, 567
  - count, 566
  - deallocate, 567

- empty, 566
- front, 567
- full, 566
- m\_back, 568
- m\_front, 568
- m\_ring, 568
- m\_ring\_size, 568
- m\_size, 568
- midi\_queue, 566
- pop, 566
- pop\_front, 567
- seq64::midi\_splitter, 568
  - ~midi\_splitter, 571
  - count, 572
  - increment, 571
  - initialize, 571
  - log\_main\_sequence, 571
  - m\_ppqn, 573
  - m\_smf0\_channels, 574
  - m\_smf0\_channels\_count, 573
  - m\_smf0\_main\_sequence, 574
  - m\_smf0\_seq\_number, 574
  - m\_use\_default\_ppqn, 573
  - midi\_splitter, 570
  - ppqn, 572
  - split, 572
  - split\_channel, 572
- seq64::midi\_timing, 574
  - beat\_width, 576
  - beats\_per\_measure, 576
  - beats\_per\_minute, 575, 576
  - m\_beat\_width, 577
  - m\_beats\_per\_measure, 577
  - m\_beats\_per\_minute, 577
  - m\_ppqn, 577
  - midi\_timing, 575
  - ppqn, 577
- seq64::midi\_vector, 578
  - ~midi\_vector, 580
  - CharVector, 580
  - clear, 582
  - done, 581
  - get, 581
  - m\_char\_vector, 582
  - midi\_vector, 580
  - put, 581
  - size, 581
- seq64::midibase, 582
  - ~midibase, 588
  - api\_clock, 601
  - api\_continue\_from, 600
  - api\_deinit\_in, 599
  - api\_flush, 600
  - api\_get\_midi\_event, 599
  - api\_init\_in, 600
  - api\_init\_in\_sub, 599
  - api\_init\_out, 600
  - api\_init\_out\_sub, 599
  - api\_play, 599
  - api\_poll\_for\_midi, 598
  - api\_start, 601
  - api\_stop, 601
  - api\_sysex, 600
  - bpm, 590
  - bus\_name, 589, 598
  - clock, 597
  - connect\_name, 589
  - continue\_from, 597
  - deinit\_in, 595
  - display\_name, 589, 598
  - flush, 596
  - get\_bus\_id, 589
  - get\_bus\_index, 589
  - get\_clock, 591
  - get\_clock\_mod, 594
  - get\_input, 592
  - get\_midi\_event, 594
  - get\_port\_id, 590
  - init\_clock, 597
  - init\_in, 595
  - init\_in\_sub, 595
  - init\_out, 595
  - init\_out\_sub, 595
  - is\_input\_port, 590, 591
  - is\_output\_port, 591
  - is\_system\_port, 591
  - is\_virtual\_port, 590
  - m\_bpm, 602
  - m\_bus\_id, 602
  - m\_bus\_index, 602
  - m\_bus\_name, 603
  - m\_clock\_mod, 601
  - m\_clock\_type, 602
  - m\_display\_name, 603
  - m\_inputting, 602
  - m\_is\_input\_port, 603
  - m\_is\_system\_port, 603
  - m\_is\_virtual\_port, 603
  - m\_lasttick, 603
  - m\_mutex, 604
  - m\_port\_id, 602
  - m\_port\_name, 603
  - m\_ppqn, 602
  - m\_queue, 602
  - mastermidibus, 601
  - match, 590
  - midibase, 587
  - play, 596
  - poll\_for\_midi, 594
  - port\_name, 589, 598
  - ppqn, 590
  - print, 597
  - queue\_number, 592
  - set\_alt\_name, 593
  - set\_bus\_id, 592
  - set\_clock, 591

- set\_clock\_mod, 594
- set\_clock\_status, 591
- set\_input, 598
- set\_input\_status, 592
- set\_multi\_name, 593
- set\_name, 592
- set\_port\_id, 598
- set\_system\_port\_flag, 591
- show\_bus\_values, 589
- start, 596
- stop, 596
- sysex, 596
- seq64::midibus, 604
  - ~midibus, 608
  - api\_clock, 612, 614
  - api\_connect, 612
  - api\_continue\_from, 611, 614
  - api\_deinit\_in, 610, 613
  - api\_flush, 611
  - api\_get\_midi\_event, 613
  - api\_init\_in, 609, 612
  - api\_init\_in\_sub, 610, 613
  - api\_init\_out, 609, 613
  - api\_init\_out\_sub, 609, 613
  - api\_play, 610, 615
  - api\_poll\_for\_midi, 614
  - api\_start, 611, 614
  - api\_stop, 612, 614
  - api\_sysex, 611
  - get\_client, 609
  - get\_port, 609
  - m\_dest\_addr\_client, 615
  - m\_dest\_addr\_port, 615
  - m\_input\_port\_name, 616
  - m\_local\_addr\_client, 616
  - m\_local\_addr\_port, 616
  - m\_master\_info, 616
  - m\_rt\_midi, 616
  - m\_seq, 615
  - mastermidibus, 615
  - midibus, 607, 608
  - set\_virtual\_name, 612
- seq64::midifile, 616
  - ~midifile, 620
  - add\_trigger, 626
  - checklen, 625
  - errdump, 633
  - error\_is\_fatal, 622
  - error\_message, 622
  - is\_sysex\_special\_id, 634
  - m\_char\_list, 636
  - m\_data, 635
  - m\_disable\_reported, 635
  - m\_error\_is\_fatal, 635
  - m\_error\_message, 635
  - m\_file\_size, 635
  - m\_global\_bgsequence, 636
  - m\_mutex, 635
  - m\_name, 635
  - m\_new\_format, 636
  - m\_pos, 635
  - m\_ppqn, 636
  - m\_smf0\_splitter, 636
  - m\_use\_default\_ppqn, 636
  - midifile, 619
  - parse, 620
  - parse\_prop\_header, 623
  - parse\_proprietary\_track, 624
  - parse\_smf\_0, 622
  - parse\_smf\_1, 623
  - pow2, 625
  - ppqn, 622
  - prop\_item\_size, 632
  - read\_byte, 626
  - read\_byte\_array, 628
  - read\_long, 626
  - read\_seq\_number, 630
  - read\_short, 626
  - read\_track\_name, 629
  - read\_varinum, 627
  - seq\_number\_size, 634
  - track\_end\_size, 634
  - track\_name\_size, 633
  - varinum\_size, 632
  - write, 621
  - write\_byte, 628
  - write\_header, 630
  - write\_long, 627
  - write\_prop\_header, 630
  - write\_proprietary\_track, 631
  - write\_seq\_number, 629
  - write\_short, 627
  - write\_song, 622
  - write\_track, 634
  - write\_track\_end, 630
  - write\_track\_name, 629
  - write\_varinum, 628
- seq64::mutex, 637
  - lock, 638
  - m\_mutex\_lock, 638
  - mutex, 638
  - sm\_recursive\_mutex, 638
  - unlock, 638
- seq64::options, 639
  - add\_extended\_keys\_page, 644
  - add\_jack\_sync\_page, 644
  - add\_keyboard\_page, 644
  - add\_midi\_clock\_page, 643
  - add\_midi\_input\_page, 644
  - add\_mouse\_page, 644
  - button, 640
  - clock\_callback\_mod, 642
  - clock\_callback\_off, 641
  - clock\_callback\_on, 642
  - clock\_mod\_callback, 642
  - filter\_callback, 642



- input\_callback, 642
- lash\_support\_callback, 643
- m\_button\_jack\_connect, 645
- m\_button\_jack\_disconnect, 645
- m\_button\_jack\_master, 645
- m\_button\_jack\_master\_cond, 645
- m\_button\_jack\_transport, 645
- m\_button\_ok, 645
- m\_mainperf, 644
- m\_notebook, 645
- m\_tooltips, 644
- mouse\_click\_edit\_callback, 643
- mouse\_fruity\_callback, 643
- mouse\_mod4\_callback, 643
- mouse\_seq24\_callback, 643
- mouse\_snap\_split\_callback, 643
- options, 641
- perf, 641
- transport\_callback, 642
- seq64::optionsfile, 646
  - ~optionsfile, 647
  - error\_message, 649
  - optionsfile, 647
  - parse, 647
  - write, 649
- seq64::perfedit, 650
  - ~perfedit, 655
  - collapse, 659
  - copy, 659
  - draw\_sequences, 660
  - enqueue\_draw, 655
  - enregister\_peer, 656
  - expand, 659
  - fast\_forward, 657
  - get\_toggle\_jack, 656
  - grow, 659
  - init\_before\_show, 655
  - m\_bpm, 667
  - m\_button\_bpm, 666
  - m\_button\_bw, 666
  - m\_button\_collapse, 665
  - m\_button\_copy, 665
  - m\_button\_expand, 665
  - m\_button\_follow, 666
  - m\_button\_grow, 665
  - m\_button\_jack, 665
  - m\_button\_loop, 665
  - m\_button\_play, 664
  - m\_button\_redo, 665
  - m\_button\_snap, 664
  - m\_button\_stop, 664
  - m\_button\_undo, 665
  - m\_button\_xpose, 664
  - m\_bw, 667
  - m\_entry\_bpm, 666
  - m\_entry\_bw, 666
  - m\_entry\_snap, 664
  - m\_entry\_xpose, 664
  - m\_hadjust, 663
  - m\_hbox, 666
  - m\_hlbox, 666
  - m\_hscroll, 663
  - m\_image\_play, 664
  - m\_is\_running, 667
  - m\_menu\_bpm, 667
  - m\_menu\_bw, 667
  - m\_menu\_snap, 663
  - m\_menu\_xpose, 664
  - m\_peer\_perfeddit, 662
  - m\_perfnames, 663
  - m\_perfroll, 663
  - m\_perftime, 663
  - m\_ppqn, 667
  - m\_snap, 667
  - m\_standard\_bpm, 667
  - m\_table, 662
  - m\_tooltips, 666
  - m\_vadjust, 663
  - m\_vscroll, 663
  - on\_delete\_event, 662
  - on\_key\_press\_event, 661
  - on\_key\_release\_event, 662
  - on\_realize, 661
  - pause\_playing, 661
  - perfedit, 654
  - popup\_menu, 660
  - redo, 660
  - rewind, 656
  - set\_beat\_width, 658
  - set\_beats\_per\_bar, 658
  - set\_follow\_transport, 657
  - set\_guides, 659
  - set\_image, 660
  - set\_jack\_mode, 657
  - set\_looped, 659
  - set\_snap, 658
  - set\_transpose, 657
  - set\_zoom, 656
  - start\_playing, 661
  - stop\_playing, 661
  - timeout, 660
  - toggle\_follow\_transport, 657
  - toggle\_jack, 656
  - toggle\_playing, 661
  - transpose\_button\_callback, 658
  - undo, 660
  - update\_perfeddit\_sequences, 662
  - zoom\_check, 655
- seq64::perfnames, 668
  - ~perfnames, 671
  - change\_vert, 672
  - convert\_y, 671
  - draw\_sequence, 671
  - draw\_sequences, 671
  - enqueue\_draw, 671
  - m\_char\_w, 675

- m\_namebox\_w, 675
- m\_names\_chars, 675
- m\_names\_x, 675
- m\_names\_y, 675
- m\_parent, 674
- m\_seqs\_in\_set, 676
- m\_sequence\_active, 676
- m\_sequence\_max, 676
- m\_sequence\_offset, 676
- m\_setbox\_w, 675
- m\_xy\_offset, 675
- on\_button\_press\_event, 673
- on\_button\_release\_event, 673
- on\_expose\_event, 672
- on\_realize, 672
- on\_scroll\_event, 674
- on\_size\_allocate, 674
- perfedit, 674
- perfnames, 670
- redraw, 672
- redraw\_dirty\_sequences, 671
- seq64::perform, 676
- ~perform, 691
- activate, 735
- add\_clock, 746
- add\_input, 746
- add\_sequence, 701
- all\_notes\_off, 737
- any\_group\_unmutes, 732
- apply\_song\_transpose, 705
- armed\_saved, 709
- clamp\_track, 744
- clear\_all, 700
- clear\_sequence\_triggers, 703
- clocks\_per\_metronome, 694
- collapse, 728
- combine\_bytes, 697
- copy, 728
- copy\_triggers, 723
- create\_master\_bus, 745
- current\_screen\_set\_notepad, 731
- decrement\_beats\_per\_minute, 717
- decrement\_screensets, 718
- deinit\_jack\_transport, 740
- delete\_sequence, 702
- enregister, 696
- expand, 728
- FF\_RW\_timeout, 698
- FF\_rewind, 698
- fast\_forward, 700
- ff\_rw\_button\_t, 691
- ff\_rw\_type, 699
- filter\_by\_channel, 695
- finish, 703
- get\_32nds\_per\_quarter, 694
- get\_beat\_width, 694
- get\_beats\_per\_bar, 693
- get\_beats\_per\_minute, 706
- get\_current\_jack\_position, 751
- get\_follow\_transport, 699
- get\_group\_mute\_state, 711
- get\_input, 747
- get\_jack\_tick, 703
- get\_key\_events, 713
- get\_key\_events\_rev, 713
- get\_key\_groups, 713
- get\_key\_groups\_rev, 713
- get\_left\_tick, 704
- get\_max\_trigger, 728
- get\_offset, 712
- get\_playing\_screensets, 726
- get\_right\_tick, 705
- get\_screen\_set\_notepad, 731
- get\_screensets, 726
- get\_sequence, 719, 720
- get\_start\_tick, 704
- get\_tick, 703
- get\_toggle\_jack, 697
- get\_transpose, 706
- gui, 695
- handle\_midi\_control, 730
- handle\_midi\_control\_ex, 731
- have\_redo, 727
- have\_undo, 727
- highlight, 719
- increment\_beats\_per\_minute, 718
- increment\_screensets, 719
- init\_jack\_transport, 739
- inner\_start, 744
- inner\_stop, 744
- input\_func, 710
- input\_thread\_func, 748
- install\_sequence, 743
- is\_active, 705
- is\_control\_status, 692
- is\_dirty\_edit, 724
- is\_dirty\_main, 724
- is\_dirty\_names, 725
- is\_dirty\_perf, 725
- is\_edit\_sequence, 693
- is\_exportable, 725
- is\_group\_learning, 734
- is\_input\_system\_port, 747
- is\_jack\_master, 696
- is\_jack\_running, 696
- is\_modified, 692, 740
- is\_mseq\_valid, 743
- is\_pattern\_playing, 696, 742
- is\_running, 696
- is\_screensets\_valid, 741
- is\_seq\_valid, 742
- is\_sequence\_in\_edit, 702
- is\_smf\_0, 719
- jack\_assistant, 747
- jack\_shutdown, 749
- jack\_sync\_callback, 749

jack\_timebase\_callback, 750  
jack\_transport\_callback, 749  
key\_name, 712  
keybindentry, 747  
keys, 695  
launch, 700  
launch\_input\_thread, 739  
launch\_output\_thread, 739  
learn\_toggle, 717  
left\_right\_size, 705  
lookup\_keyevent\_key, 714  
lookup\_keyevent\_seq, 714  
lookup\_keygroup\_group, 715  
lookup\_keygroup\_key, 715  
m\_32nds\_per\_quarter, 757  
m\_FF\_RW\_button\_type, 752  
m\_armed\_saved, 752  
m\_armed\_statuses, 752  
m\_beat\_width, 756  
m\_beats\_per\_bar, 756  
m\_bpm, 756  
m\_clocks\_per\_metronome, 756  
m\_condition\_var, 761  
m\_control\_status, 759  
m\_dont\_reset\_ticks, 759  
m\_edit\_sequence, 760  
m\_excell\_FF\_RW, 752  
m\_gui\_support, 762  
m\_have\_redo, 761  
m\_have\_undo, 761  
m\_in\_thread, 755  
m\_in\_thread\_launched, 755  
m\_inputting, 755  
m\_is\_modified, 761  
m\_is\_pattern\_playing, 755  
m\_jack\_asst, 761  
m\_jack\_tick, 758  
m\_left\_tick, 757  
m\_looping, 756  
m\_master\_bus, 757  
m\_master\_clocks, 757  
m\_master\_inputs, 757  
m\_max\_sets, 760  
m\_midi\_cc\_off, 759  
m\_midi\_cc\_on, 759  
m\_midi\_cc\_toggle, 759  
m\_midiclockpos, 759  
m\_midiclockrunning, 758  
m\_midiclocktick, 758  
m\_mode\_group, 753  
m\_mode\_group\_learn, 753  
m\_mute\_group, 752  
m\_mute\_group\_selected, 753  
m\_notify, 762  
m\_offset, 759  
m\_one\_measure, 757  
m\_out\_thread, 754  
m\_out\_thread\_launched, 755  
m\_outputting, 755  
m\_playback\_mode, 756  
m\_playing\_screen, 753  
m\_playscreen\_offset, 753  
m\_ppqn, 756  
m\_redo\_vect, 761  
m\_reposition, 752  
m\_right\_tick, 758  
m\_running, 755  
m\_screen\_set\_notepad, 759  
m\_screenset, 760  
m\_seqs, 753  
m\_seqs\_active, 753  
m\_seqs\_in\_set, 760  
m\_sequence\_count, 760  
m\_sequence\_high, 760  
m\_sequence\_max, 760  
m\_sequence\_state, 754  
m\_song\_start\_mode, 751  
m\_start\_from\_perfeddit, 751  
m\_starting\_tick, 758  
m\_tick, 758  
m\_tracks\_mute\_state, 752  
m\_transpose, 754  
m\_undo\_vect, 761  
m\_us\_per\_quarter\_note, 757  
m\_usemidiclock, 758  
m\_was\_active\_edit, 754  
m\_was\_active\_main, 754  
m\_was\_active\_names, 754  
m\_was\_active\_perf, 754  
mainwnd, 747  
mainwnd\_key\_event, 722  
master\_bus, 695  
max\_active\_set, 739  
midi\_control\_event, 730  
midi\_control\_off, 729  
midi\_control\_on, 729  
midi\_control\_toggle, 728  
midifile, 747  
modify, 692  
move\_triggers, 723  
mute\_all\_tracks, 709  
mute\_group\_offset, 742  
mute\_group\_tracks, 732  
mute\_op\_t, 690  
mute\_screenset, 710  
new\_sequence, 701  
off\_sequences, 736  
options, 748  
optionsfile, 747  
output\_func, 710  
output\_thread\_func, 748  
page\_decrement\_beats\_per\_minute, 718  
page\_increment\_beats\_per\_minute, 718  
pause\_key, 717  
pause\_playing, 716  
perfeddit, 748

perform, 691  
 perfroll, 748  
 perfroll\_key\_event, 722  
 play, 738  
 playback\_key\_event, 722  
 pop\_trigger\_redo, 724  
 pop\_trigger\_undo, 724  
 position\_jack, 736  
 ppqn, 692  
 print\_triggers, 703  
 push\_trigger\_undo, 723  
 reposition, 700  
 reset\_sequences, 737  
 restore\_playing\_state, 712  
 rewind, 699  
 save\_playing\_state, 712  
 select\_and\_mute\_group, 733  
 select\_group\_mute, 734  
 seq\_in\_playing\_screen, 740  
 sequence\_count, 692  
 sequence\_key, 720  
 sequence\_label, 720  
 sequence\_max, 692  
 sequence\_playing\_change, 708  
 sequence\_playing\_off, 709  
 sequence\_playing\_on, 708  
 sequence\_playing\_toggle, 708  
 set\_32nds\_per\_quarter, 694  
 set\_active, 737  
 set\_and\_copy\_mute\_group, 734  
 set\_beat\_width, 694  
 set\_beats\_per\_bar, 693  
 set\_beats\_per\_minute, 738  
 set\_clock, 746  
 set\_clock\_bus, 721  
 set\_edit\_sequence, 692  
 set\_follow\_transport, 699  
 set\_group\_mute\_state, 711  
 set\_have\_redo, 727  
 set\_have\_undo, 727  
 set\_input, 746  
 set\_input\_bus, 721  
 set\_jack\_mode, 697  
 set\_jack\_stop\_tick, 697  
 set\_jack\_tick, 703  
 set\_key\_event, 745  
 set\_key\_group, 745  
 set\_left\_tick, 704  
 set\_looping, 739  
 set\_mode\_group\_learn, 734  
 set\_mode\_group\_mute, 733  
 set\_offset, 712  
 set\_orig\_ticks, 738  
 set\_playback\_mode, 742  
 set\_playing\_screensets, 732  
 set\_reposition, 699  
 set\_right\_tick, 704  
 set\_running, 741  
 set\_screen\_set\_notepad, 731, 732  
 set\_screensets, 726  
 set\_sequence\_control\_status, 706  
 set\_song\_mute, 733  
 set\_start\_tick, 704  
 set\_tick, 703  
 set\_transpose, 706  
 set\_was\_active, 737  
 show\_ui\_sequence\_key, 713  
 show\_ui\_sequence\_number, 713, 714  
 sm\_mc\_dummy, 751  
 song\_start\_mode, 696  
 split\_trigger, 728  
 start, 735  
 start\_from\_perfeddit, 698  
 start\_jack, 736  
 start\_key, 717  
 start\_playing, 715  
 stop, 735  
 stop\_jack, 736  
 stop\_key, 717  
 stop\_playing, 717  
 toggle\_all\_tracks, 709  
 toggle\_follow\_transport, 699  
 toggle\_jack\_mode, 697  
 toggle\_other\_names, 727  
 toggle\_other\_seqs, 726  
 toggle\_playing\_tracks, 709  
 toggle\_song\_start\_mode, 696  
 unset\_edit\_sequence, 693  
 unset\_mode\_group\_learn, 734  
 unset\_mode\_group\_mute, 733  
 unset\_sequence\_control\_status, 706  
 us\_per\_quarter\_note, 694, 695  
 valid\_midi\_control\_seq, 741  
 seq64::performcallback, 762  
   on\_grouplearnchange, 764  
 seq64::perfroll, 764  
   ~perfroll, 769  
   change\_horz, 773  
   change\_vert, 773  
   convert\_drop\_xy, 774  
   convert\_x, 772  
   convert\_xy, 772  
   draw\_all, 771  
   draw\_background\_on, 773  
   draw\_drawable\_row, 773  
   draw\_progress, 771  
   draw\_sequence\_on, 773  
   enqueue\_draw, 774  
   fill\_background\_pixmap, 770  
   follow\_progress, 771  
   FruityPerfInput, 778  
   horizontal\_adjust, 774  
   horizontal\_set, 775  
   increment\_size, 770  
   init\_before\_show, 770  
   m\_4bar\_offset, 781

- m\_background\_x, 780
- m\_beat\_length, 781
- m\_divs\_per\_beat, 780
- m\_drop\_sequence, 782
- m\_drop\_tick, 782
- m\_drop\_tick\_trigger\_offset, 782
- m\_fruity\_interaction, 783
- m\_grow\_direction, 783
- m\_growing, 783
- m\_h\_page\_increment, 779
- m\_have\_button\_press, 781
- m\_interaction, 783
- m\_measure\_length, 781
- m\_moving, 783
- m\_names\_y, 780
- m\_old\_progress\_ticks, 781
- m\_page\_factor, 779
- m\_parent, 779
- m\_perf\_scale\_x, 780
- m\_ppqn, 779
- m\_roll\_length\_ticks, 782
- m\_seq24\_interaction, 783
- m\_sequence\_active, 782
- m\_sequence\_max, 782
- m\_sequence\_offset, 782
- m\_size\_box\_w, 780
- m\_snap, 779
- m\_ticks\_per\_bar, 780
- m\_trans\_button\_press, 781
- m\_transport\_follow, 781
- m\_v\_page\_increment, 779
- m\_zoom, 780
- on\_button\_press\_event, 776
- on\_button\_release\_event, 776
- on\_expose\_event, 776
- on\_focus\_in\_event, 777
- on\_focus\_out\_event, 777
- on\_key\_press\_event, 778
- on\_motion\_notify\_event, 777
- on\_realize, 775
- on\_scroll\_event, 777
- on\_size\_allocate, 777
- on\_size\_request, 778
- perfedit, 779
- perffroll, 769
- redraw\_dirty\_sequences, 771
- redraw\_progress, 771
- Seq24PerfInput, 778
- set\_guides, 769
- set\_ppqn, 771
- set\_zoom, 774
- snap\_x, 773
- split\_trigger, 774
- update\_sizes, 770
- vertical\_adjust, 775
- vertical\_set, 775
- seq64::perftime, 784
- ~perftime, 787
- change\_horz, 788
- draw\_background, 788
- draw\_pixmap\_on\_window, 790
- draw\_progress\_on\_window, 788
- enqueue\_draw, 788
- idle\_progress, 790
- increment\_size, 788
- key\_press\_event, 791
- m\_4bar\_offset, 792
- m\_left\_marker\_tick, 793
- m\_measure\_length, 793
- m\_parent, 792
- m\_perf\_scale\_x, 793
- m\_ppqn, 792
- m\_right\_marker\_tick, 793
- m\_snap, 793
- m\_tick\_offset, 792
- m\_timearea\_y, 793
- on\_button\_press\_event, 791
- on\_button\_release\_event, 791
- on\_expose\_event, 790
- on\_realize, 790
- on\_size\_allocate, 791
- perfedit, 792
- perftime, 786
- pixel\_to\_tick, 789
- reset, 787
- set\_guides, 787
- set\_ppqn, 788
- set\_scale, 787
- set\_zoom, 788
- tick\_offset, 789
- tick\_to\_pixel, 789
- update\_pixmap, 790
- update\_sizes, 790
- seq64::rc\_settings, 795
- allow\_click\_edit, 802, 806
- allow\_mod4\_mode, 802, 806
- allow\_snap\_split, 802, 806
- app\_client\_name, 805
- application\_name, 805
- auto\_option\_save, 802, 806
- config\_directory, 805, 810
- config\_filename, 805, 810
- config\_filename\_alt, 805, 811
- config\_filespec, 801
- device\_ignore, 804, 808
- device\_ignore\_num, 804, 808
- filename, 804, 809
- filter\_by\_channel, 803, 808
- help\_check, 812
- home\_config\_directory, 811
- interaction\_method, 804, 809
- jack\_session\_uuid, 804, 809
- lash\_support, 802, 806
- last\_used\_dir, 805, 809
- legacy\_format, 802, 806
- m\_allow\_click\_edit, 814

- m\_allow\_mod4\_mode, 813
- m\_allow\_snap\_split, 813
- m\_app\_client\_name, 817
- m\_application\_name, 817
- m\_auto\_option\_save, 813
- m\_config\_directory, 816
- m\_config\_filename, 816
- m\_config\_filename\_alt, 816
- m\_device\_ignore, 815
- m\_device\_ignore\_num, 815
- m\_filename, 816
- m\_filter\_by\_channel, 815
- m\_interaction\_method, 815
- m\_jack\_session\_uuid, 816
- m\_lash\_support, 813
- m\_last\_used\_dir, 816
- m\_legacy\_format, 813
- m\_manual\_alsa\_ports, 815
- m\_pass\_sysex, 814
- m\_print\_keys, 815
- m\_priority, 814
- m\_reveal\_alsa\_ports, 815
- m\_show\_midi, 814
- m\_stats, 814
- m\_user\_filename, 816
- m\_user\_filename\_alt, 816
- m\_with\_jack\_master, 814
- m\_with\_jack\_master\_cond, 814
- m\_with\_jack\_midi, 815
- m\_with\_jack\_transport, 814
- mainwnd, 812
- manual\_alsa\_ports, 804, 808
- operator=, 801
- options, 812
- optionsfile, 811
- parse\_command\_line\_options, 812
- pass\_sysex, 803, 807
- print\_keys, 804, 808
- priority, 802, 807
- rc\_settings, 800
- reveal\_alsa\_ports, 804, 808
- rtmidi\_info, 812
- set\_config\_files, 810
- set\_defaults, 801
- show\_midi, 802, 806
- stats, 803, 807
- user\_filename, 805, 810
- user\_filename\_alt, 805, 811
- user\_filespec, 801
- with\_jack, 803
- with\_jack\_master, 803, 807
- with\_jack\_master\_cond, 803, 807
- with\_jack\_midi, 803, 807
- with\_jack\_transport, 803, 807
- seq64::rect, 817
- height, 818
- width, 818
- x, 817
- y, 817
- seq64::rterror, 819
  - ~rterror, 820
  - get\_message, 821
  - getType, 821
  - m\_message, 821
  - m\_type, 821
  - print\_message, 821
  - rterror, 820
  - Type, 820
  - what, 821
- seq64::rtmidi, 822
  - ~rtmidi, 824
  - api\_clock, 825
  - api\_connect, 824
  - api\_continue\_from, 825
  - api\_deinit\_in, 826
  - api\_flush, 827
  - api\_get\_midi\_event, 826
  - api\_init\_in, 826
  - api\_init\_in\_sub, 826
  - api\_init\_out, 826
  - api\_init\_out\_sub, 826
  - api\_play, 824
  - api\_poll\_for\_midi, 827
  - api\_set\_beats\_per\_minute, 825
  - api\_set\_ppqn, 825
  - api\_start, 825
  - api\_stop, 825
  - api\_sysex, 827
  - delete\_api, 829
  - full\_port\_count, 828
  - get\_api, 828, 829
  - get\_bus\_id, 827
  - get\_bus\_name, 827
  - get\_port\_count, 828
  - get\_port\_id, 828
  - get\_port\_name, 828
  - is\_port\_open, 827
  - m\_midi\_api, 829
  - m\_midi\_info, 829
  - midibus, 829
  - rtmidi, 824
  - set\_api, 829
- seq64::rtmidi\_in, 830
  - ~rtmidi\_in, 832
  - cancel\_callback, 833
  - openmidi\_api, 833
  - rtmidi\_in, 832
  - user\_callback, 833
- seq64::rtmidi\_in\_data, 834
  - api\_data, 837
  - continue\_sysex, 836
  - do\_input, 836
  - first\_message, 836
  - ignore\_flags, 835, 836
  - m\_api\_data, 839
  - m\_continue\_sysex, 839

- m\_do\_input, 838
- m\_first\_message, 839
- m\_ignore\_flags, 838
- m\_message, 838
- m\_queue, 838
- m\_user\_callback, 839
- m\_user\_data, 839
- m\_using\_callback, 839
- message, 835
- queue, 835
- rtmidi\_in\_data, 835
- test\_ignore\_flags, 835
- user\_callback, 838
- user\_data, 837, 838
- using\_callback, 837
- seq64::rtmidi\_info, 839
  - ~rtmidi\_info, 842
  - add\_bus, 843
  - add\_input, 843
  - add\_output, 843
  - api\_connect, 847
  - api\_flush, 846
  - api\_get\_midi\_event, 846
  - api\_poll\_for\_midi, 846
  - api\_port\_start, 846
  - api\_set\_beats\_per\_minute, 846
  - api\_set\_ppqn, 846
  - app\_name, 845
  - bpm, 846
  - clear, 843
  - delete\_api, 848
  - full\_port\_count, 844
  - get\_all\_port\_info, 845
  - get\_api\_info, 847
  - get\_bus\_id, 843
  - get\_bus\_name, 844
  - get\_compiled\_api, 842
  - get\_input, 844
  - get\_port\_count, 844
  - get\_port\_id, 844
  - get\_port\_name, 844
  - get\_system, 845
  - get\_version, 842
  - get\_virtual, 845
  - global\_queue, 845
  - m\_info\_api, 849
  - mastermidibus, 848
  - midi\_mode, 842, 843
  - midibus, 848
  - openmidi\_api, 848
  - port\_list, 847
  - ppqn, 845
  - queue\_number, 845
  - rtmidi\_in, 849
  - rtmidi\_info, 842
  - rtmidi\_out, 849
  - selected\_api, 847
  - set\_api\_info, 847
  - sm\_selected\_api, 849
- seq64::rtmidi\_out, 850
  - ~rtmidi\_out, 851
  - openmidi\_api, 852
  - rtmidi\_out, 851
- seq64::seqdata, 858
  - ~seqdata, 862
  - change\_horz, 865
  - convert\_x, 865
  - draw\_events\_on, 865
  - draw\_events\_on\_pixmap, 866
  - draw\_line\_on\_window, 863
  - draw\_pixmap\_on\_window, 866
  - idle\_redraw, 863
  - lfownd, 869
  - m\_cc, 870
  - m\_drag\_handle, 871
  - m\_dragging, 871
  - m\_number\_h, 870
  - m\_number\_offset\_y, 870
  - m\_number\_w, 870
  - m\_numbers, 870
  - m\_old, 870
  - m\_scroll\_offset\_ticks, 869
  - m\_scroll\_offset\_x, 870
  - m\_seq, 869
  - m\_status, 870
  - m\_zoom, 869
  - on\_button\_press\_event, 867
  - on\_button\_release\_event, 867
  - on\_expose\_event, 866
  - on\_leave\_notify\_event, 868
  - on\_motion\_notify\_event, 867
  - on\_realize, 866
  - on\_scroll\_event, 868
  - on\_size\_allocate, 868
  - redraw, 862
  - render\_number, 866
  - reset, 862
  - seqdata, 861
  - seqevent, 869
  - seqroll, 869
  - set\_data\_type, 863
  - set\_zoom, 862
  - update\_pixmap, 863
  - update\_sizes, 863
  - xy\_to\_rect, 863
- seq64::seqedit, 871
  - ~seqedit, 879
  - apply\_length, 883
  - change\_focus, 889
  - create\_menu\_image, 888
  - create\_menus, 887
  - do\_action, 888
  - fill\_top\_bar, 886
  - get\_measures, 883
  - handle\_close, 889
  - horizontal\_adjust, 882

horizontal\_set, 882  
 m\_bgsequence, 893  
 m\_button\_bpm, 902  
 m\_button\_bus, 899  
 m\_button\_bw, 902  
 m\_button\_channel, 899  
 m\_button\_chord, 901  
 m\_button\_data, 902  
 m\_button\_key, 901  
 m\_button\_length, 900  
 m\_button\_lfo, 898  
 m\_button\_note\_length, 900  
 m\_button\_quantize, 899  
 m\_button\_rec\_vol, 902  
 m\_button\_redo, 899  
 m\_button\_scale, 901  
 m\_button\_sequence, 899  
 m\_button\_snap, 900  
 m\_button\_tools, 899  
 m\_button\_undo, 898  
 m\_button\_zoom, 900  
 m\_chord, 893  
 m\_editing\_cc, 903  
 m\_editing\_status, 903  
 m\_entry\_bpm, 902  
 m\_entry\_bus, 899  
 m\_entry\_bw, 902  
 m\_entry\_channel, 900  
 m\_entry\_chord, 901  
 m\_entry\_data, 902  
 m\_entry\_key, 901  
 m\_entry\_length, 901  
 m\_entry\_name, 903  
 m\_entry\_note\_length, 900  
 m\_entry\_scale, 901  
 m\_entry\_sequence, 899  
 m\_entry\_snap, 900  
 m\_entry\_zoom, 900  
 m\_hadjust, 897  
 m\_have\_focus, 903  
 m\_hbox, 898  
 m\_hbox2, 898  
 m\_hscroll\_new, 897  
 m\_image\_transpose, 895  
 m\_initial\_chord, 892  
 m\_initial\_note\_length, 892  
 m\_initial\_snap, 892  
 m\_initial\_zoom, 892  
 m\_key, 893  
 m\_lfo\_wnd, 898  
 m\_measures, 893  
 m\_menu\_bpm, 896  
 m\_menu\_bw, 896  
 m\_menu\_chords, 896  
 m\_menu\_data, 896  
 m\_menu\_key, 896  
 m\_menu\_length, 895  
 m\_menu\_midibus, 895  
 m\_menu\_midich, 895  
 m\_menu\_note\_length, 895  
 m\_menu\_rec\_vol, 896  
 m\_menu\_scale, 896  
 m\_menu\_sequences, 896  
 m\_menu\_snap, 895  
 m\_menu\_tools, 894  
 m\_menu\_zoom, 895  
 m\_menubar, 894  
 m\_note\_length, 893  
 m\_pp\_eighth, 894  
 m\_pp\_sixteenth, 894  
 m\_pp\_whole, 894  
 m\_ppqn, 894  
 m\_scale, 893  
 m\_seq, 894  
 m\_seqdata\_wid, 897  
 m\_seqevent\_wid, 897  
 m\_seqkeys\_wid, 897  
 m\_seqroll\_wid, 898  
 m\_seqtime\_wid, 897  
 m\_snap, 893  
 m\_table, 898  
 m\_toggle\_play, 902  
 m\_toggle\_q\_rec, 903  
 m\_toggle\_record, 903  
 m\_toggle\_thru, 903  
 m\_toggle\_transpose, 895  
 m\_tooltips, 901  
 m\_vadjust, 897  
 m\_vbox, 898  
 m\_vscroll\_new, 897  
 m\_zoom, 893  
 mouse\_action, 889  
 name\_change\_callback, 885  
 on\_delete\_event, 890  
 on\_focus\_in\_event, 890  
 on\_focus\_out\_event, 890  
 on\_key\_press\_event, 891  
 on\_realize, 889  
 on\_scroll\_event, 890  
 on\_set\_focus, 890  
 play\_change\_callback, 885  
 popup\_event\_menu, 887  
 popup\_menu, 887  
 popup\_midibus\_menu, 887  
 popup\_midich\_menu, 888  
 popup\_sequence\_menu, 888  
 popup\_tool\_menu, 888  
 q\_rec\_change\_callback, 885  
 record\_change\_callback, 885  
 redo\_callback, 886  
 seqedit, 879  
 seqmenu, 892  
 set\_background\_sequence, 884  
 set\_beat\_width, 881  
 set\_beats\_per\_bar, 880  
 set\_chord, 884



- set\_data\_type, 886
- set\_key, 884
- set\_measures, 883
- set\_midi\_bus, 884
- set\_midi\_channel, 883
- set\_note\_length, 880
- set\_rec\_vol, 881
- set\_scale, 884
- set\_snap, 880
- set\_transpose\_image, 881
- set\_zoom, 879
- start\_playing, 889
- stop\_playing, 889
- thru\_change\_callback, 885
- timeout, 888
- transpose\_change\_callback, 885
- undo\_callback, 886
- update\_all\_windows, 886
- vertical\_adjust, 882
- vertical\_set, 882
- seq64::seqevent, 904
  - ~seqevent, 907
  - change\_horz, 911
  - convert\_t, 912
  - convert\_x, 911
  - draw\_background, 909
  - draw\_events\_on, 911
  - draw\_events\_on\_pixmap, 909
  - draw\_pixmap\_on\_window, 909
  - draw\_selection\_on\_window, 909
  - drop\_event, 910
  - force\_draw, 910
  - FruitySeqEventInput, 915
  - idle\_redraw, 910
  - m\_cc, 918
  - m\_fruity\_interaction, 916
  - m\_growing, 917
  - m\_move\_snap\_offset\_x, 918
  - m\_moving, 917
  - m\_moving\_init, 917
  - m\_old, 916
  - m\_painting, 917
  - m\_paste, 918
  - m\_ppqn, 916
  - m\_scroll\_offset\_ticks, 917
  - m\_scroll\_offset\_x, 917
  - m\_selected, 916
  - m\_selecting, 917
  - m\_seq, 916
  - m\_seq24\_interaction, 916
  - m\_seqdata\_wid, 917
  - m\_snap, 916
  - m\_status, 918
  - m\_zoom, 916
  - on\_button\_press\_event, 913
  - on\_button\_release\_event, 913
  - on\_expose\_event, 913
  - on\_focus\_in\_event, 914
  - on\_focus\_out\_event, 914
  - on\_key\_press\_event, 914
  - on\_motion\_notify\_event, 914
  - on\_realize, 912
  - on\_size\_allocate, 915
  - redraw, 908
  - reset, 908
  - Seq24SeqEventInput, 915
  - seqevent, 907
  - set\_data\_type, 908
  - set\_snap, 908
  - set\_zoom, 908
  - snap\_x, 912
  - snap\_y, 912
  - start\_paste, 911
  - update\_pixmap, 909
  - update\_sizes, 909
  - x\_to\_w, 910
- seq64::seqkeys, 918
  - ~seqkeys, 922
  - change\_vert, 925
  - convert\_y, 924
  - draw\_area, 924
  - draw\_key, 924
  - force\_draw, 923
  - FruitySeqRollInput, 928
  - is\_black\_key, 925
  - m\_hint\_key, 929
  - m\_hint\_state, 929
  - m\_key, 929
  - m\_keying, 929
  - m\_keying\_note, 929
  - m\_scale, 929
  - m\_scroll\_offset\_key, 928
  - m\_scroll\_offset\_y, 929
  - m\_seq, 928
  - m\_show\_octave\_letters, 929
  - on\_button\_press\_event, 926
  - on\_button\_release\_event, 926
  - on\_enter\_notify\_event, 927
  - on\_expose\_event, 926
  - on\_leave\_notify\_event, 927
  - on\_motion\_notify\_event, 927
  - on\_realize, 925
  - on\_scroll\_event, 927
  - on\_size\_allocate, 928
  - reset, 925
  - seqkeys, 921
  - seqroll, 928
  - set\_hint\_key, 922
  - set\_hint\_state, 923
  - set\_key, 922
  - set\_listen\_button\_press, 923
  - set\_listen\_button\_release, 923
  - set\_listen\_motion\_notify, 923
  - set\_scale, 922
  - update\_pixmap, 924
  - update\_sizes, 925

- seq64::seqmenu, 930
  - ~seqmenu, 935
  - const\_iterator, 934
  - create\_seqedit, 938
  - current\_seq, 935
  - delete\_current\_sequence, 937
  - get\_current\_sequence, 936
  - get\_sequence, 936
  - is\_current\_seq\_active, 936
  - is\_current\_seq\_in\_edit, 936
  - is\_edit\_sequence, 936
  - is\_modified, 935, 936
  - iterator, 934
  - m\_clipboard, 942
  - m\_current\_seq, 942
  - m\_eventedit, 942
  - m\_mainperf, 942
  - m\_menu, 942
  - m\_modified, 943
  - m\_seqedit, 942
  - mainwnd, 941
  - mute\_all\_tracks, 941
  - new\_current\_sequence, 936
  - new\_sequence, 937
  - on\_realize, 941
  - popup\_menu, 937
  - redraw, 939
  - remove\_seqedit, 938
  - seq\_clear\_perf, 940
  - seq\_copy, 939
  - seq\_cut, 939
  - seq\_edit, 937
  - seq\_event\_edit, 937
  - seq\_new, 939
  - seq\_paste, 940
  - seq\_set\_and\_edit, 938
  - seq\_set\_and\_eventedit, 939
  - seqedit, 941
  - SeqeditMap, 934
  - SeqeditPair, 934
  - seqmenu, 934
  - set\_bus\_and\_midi\_channel, 940
  - set\_edit\_sequence, 935
  - set\_transposable, 940
  - sm\_seqedit\_list, 942
  - toggle\_all\_tracks, 941
  - toggle\_current\_sequence, 937
  - unmute\_all\_tracks, 941
  - unset\_edit\_sequence, 935
- seq64::seqroll, 943
  - ~seqroll, 950
  - add\_chord, 951
  - add\_note, 951
  - adding, 965
  - align\_selection, 962
  - button\_press, 962
  - button\_press\_initial, 961
  - button\_release, 962
  - change\_horz, 960
  - change\_vert, 960
  - clear\_flags, 963
  - clear\_old, 963
  - clear\_selected, 963
  - complete\_paste, 955
  - convert\_sel\_box\_to\_rect, 958
  - convert\_tn, 957
  - convert\_tn\_box\_to\_rect, 958
  - convert\_xy, 957
  - draw\_background\_on\_pixmap, 953
  - draw\_events\_on, 959
  - draw\_events\_on\_pixmap, 953
  - draw\_progress\_on\_window, 954
  - draw\_selection\_on\_window, 954
  - drop\_action, 966
  - follow\_progress, 955
  - force\_draw, 955
  - FruitySeqRollInput, 971
  - get\_selected\_box, 959
  - grow\_selected\_notes, 961
  - growing, 966
  - horizontal\_adjust, 955
  - idle\_progress, 959
  - idle\_redraw, 959
  - m\_adding, 973
  - m\_background\_sequence, 975
  - m\_cc, 976
  - m\_chord, 973
  - m\_drawing\_background\_seq, 975
  - m\_fruity\_interaction, 972
  - m\_growing, 973
  - m\_horizontal\_adjust, 971
  - m\_is\_drag\_pasting, 974
  - m\_is\_drag\_pasting\_start, 974
  - m\_justselected\_one, 974
  - m\_key, 973
  - m\_move\_delta\_x, 974
  - m\_move\_delta\_y, 974
  - m\_move\_snap\_offset\_x, 974
  - m\_moving, 973
  - m\_moving\_init, 973
  - m\_note\_length, 972
  - m\_old, 971
  - m\_painting, 973
  - m\_paste, 974
  - m\_pos, 972
  - m\_ppqn, 972
  - m\_progress\_x, 974
  - m\_scale, 972
  - m\_scroll\_offset\_key, 975
  - m\_scroll\_offset\_ticks, 975
  - m\_scroll\_offset\_x, 975
  - m\_scroll\_offset\_y, 975
  - m\_selected, 971
  - m\_selecting, 973
  - m\_seq, 971
  - m\_seqkeys\_wid, 972

- m\_snap, 972
- m\_status, 976
- m\_trans\_button\_press, 975
- m\_transport\_follow, 975
- m\_vertical\_adjust, 971
- m\_zoom, 972
- motion\_notify, 963
- move\_selected\_notes, 960
- move\_selection\_box, 960
- moving, 966
- normal\_action, 966
- note\_off\_length, 951
- on\_button\_press\_event, 967
- on\_button\_release\_event, 967
- on\_enter\_notify\_event, 970
- on\_expose\_event, 967
- on\_focus\_in\_event, 968
- on\_focus\_out\_event, 968
- on\_key\_press\_event, 969
- on\_leave\_notify\_event, 970
- on\_motion\_notify\_event, 968
- on\_realize, 967
- on\_scroll\_event, 969
- on\_size\_allocate, 970
- redraw, 955
- redraw\_events, 955
- reset, 954
- scroll\_offset\_x, 963
- scroll\_offset\_y, 965
- select\_action, 966
- selecting, 966
- seqroll, 949
- set\_adding, 961
- set\_background\_sequence, 952
- set\_chord, 952
- set\_current\_offset\_x\_y, 965
- set\_data\_type, 952
- set\_key, 952
- set\_note\_length, 951
- set\_scale, 952
- set\_snap, 950
- set\_zoom, 950
- snap\_x, 956
- snap\_y, 956
- start\_paste, 955
- update\_and\_draw, 954
- update\_background, 953
- update\_mouse\_pointer, 961
- update\_pixmap, 953
- update\_sizes, 953
- vertical\_adjust, 956
- xy\_to\_rect, 957
- seq64::seqtime, 976
  - ~seqtime, 979
  - change\_horz, 980
  - draw\_pixmap\_on\_window, 979
  - draw\_progress\_on\_window, 979
  - idle\_progress, 981
  - m\_ppqn, 982
  - m\_scroll\_offset\_ticks, 982
  - m\_scroll\_offset\_x, 982
  - m\_seq, 982
  - m\_zoom, 982
  - on\_button\_press\_event, 981
  - on\_button\_release\_event, 981
  - on\_expose\_event, 981
  - on\_realize, 981
  - on\_size\_allocate, 981
  - redraw, 979
  - reset, 979
  - seqtime, 978
  - set\_zoom, 979
  - update\_pixmap, 980
  - update\_sizes, 980
- seq64::sequence, 982
  - ~sequence, 994
  - add\_chord, 1011
  - add\_event, 1011, 1012
  - add\_note, 1010
  - add\_trigger, 1013
  - adjust\_offset, 1044
  - adjust\_timestamp, 1027
  - adjust\_trigger\_offsets\_to\_length, 1044
  - any\_selected\_notes, 995
  - append\_event, 1012
  - apply\_song\_transpose, 1002
  - background\_sequence, 1041, 1042
  - change\_event\_data\_lfo, 1031
  - change\_event\_data\_range, 1030
  - channel\_match, 1045
  - check\_queued\_tick, 1006
  - clear\_triggers, 1020
  - clip\_timestamp, 1028
  - clocks\_per\_metronome, 1000
  - copy\_events, 1042
  - copy\_selected, 1025
  - copy\_selected\_trigger, 1017
  - copy\_triggers, 1019
  - cut\_selected, 1025
  - cut\_selected\_trigger, 1017
  - decrement\_selected, 1032
  - del\_selected\_trigger, 1017
  - del\_trigger, 1014
  - event\_count, 996
  - event\_in\_range, 1042
  - EventStack, 993
  - events, 995
  - get\_32nds\_per\_quarter, 1000
  - get\_beat\_width, 1000
  - get\_beats\_per\_bar, 999
  - get\_clipboard\_box, 1027
  - get\_editing, 1002
  - get\_hold\_undo, 997
  - get\_last\_tick, 1004
  - get\_length, 1004
  - get\_max\_trigger, 1019

[get\\_measures](#), 999  
[get\\_midi\\_bus](#), 1020  
[get\\_midi\\_channel](#), 1008  
[get\\_minmax\\_note\\_events](#), 1038  
[get\\_name](#), 1002  
[get\\_next\\_event](#), 1038, 1039  
[get\\_next\\_note\\_event](#), 1036  
[get\\_next\\_trigger](#), 1039  
[get\\_num\\_selected\\_events](#), 1024  
[get\\_num\\_selected\\_notes](#), 1024  
[get\\_playing](#), 1005  
[get\\_ppqn](#), 999  
[get\\_quantized\\_rec](#), 1006  
[get\\_queued](#), 1005  
[get\\_queued\\_tick](#), 1005  
[get\\_raise](#), 1003  
[get\\_recording](#), 1006  
[get\\_selected\\_box](#), 1026  
[get\\_song\\_mute](#), 1001  
[get\\_thru](#), 1007  
[get\\_transposable](#), 1002  
[get\\_trigger\\_count](#), 995  
[get\\_trigger\\_offset](#), 1020  
[get\\_trigger\\_paste\\_tick](#), 995  
[get\\_trigger\\_state](#), 1014  
[get\\_triggers](#), 1015  
[grow\\_selected](#), 1033  
[grow\\_trigger](#), 1014  
[have\\_redo](#), 997  
[have\\_undo](#), 997  
[increment\\_selected](#), 1032  
[intersect\\_events](#), 1017  
[intersect\\_notes](#), 1016  
[intersect\\_triggers](#), 1015  
[is\\_dirty\\_edit](#), 1007  
[is\\_dirty\\_main](#), 1007  
[is\\_dirty\\_names](#), 1007  
[is\\_dirty\\_perf](#), 1007  
[is\\_smf\\_0](#), 1008  
[link\\_new](#), 1035  
[m\\_32nds\\_per\\_quarter](#), 1051  
[m\\_background\\_sequence](#), 1052  
[m\\_bus](#), 1047  
[m\\_channel\\_match](#), 1047  
[m\\_clocks\\_per\\_metronome](#), 1051  
[m\\_dirty\\_edit](#), 1049  
[m\\_dirty\\_main](#), 1049  
[m\\_dirty\\_names](#), 1049  
[m\\_dirty\\_perf](#), 1049  
[m\\_editing](#), 1049  
[m\\_events](#), 1046  
[m\\_events\\_clipboard](#), 1046  
[m\\_events\\_redo](#), 1047  
[m\\_events\\_undo](#), 1047  
[m\\_events\\_undo\\_hold](#), 1046  
[m\\_have\\_redo](#), 1047  
[m\\_have\\_undo](#), 1046  
[m\\_iterator\\_draw](#), 1047  
[m\\_last\\_tick](#), 1050  
[m\\_length](#), 1051  
[m\\_masterbus](#), 1048  
[m\\_maxbeats](#), 1050  
[m\\_midi\\_channel](#), 1047  
[m\\_musical\\_key](#), 1052  
[m\\_musical\\_scale](#), 1052  
[m\\_mutex](#), 1052  
[m\\_name](#), 1050  
[m\\_note\\_off\\_margin](#), 1053  
[m\\_note\\_off\\_velocity](#), 1052  
[m\\_note\\_on\\_velocity](#), 1052  
[m\\_notes\\_on](#), 1048  
[m\\_parent](#), 1046  
[m\\_playing](#), 1048  
[m\\_playing\\_notes](#), 1048  
[m\\_ppqn](#), 1050  
[m\\_quantized\\_rec](#), 1049  
[m\\_queued](#), 1049  
[m\\_queued\\_tick](#), 1050  
[m\\_raise](#), 1050  
[m\\_rec\\_vol](#), 1052  
[m\\_recording](#), 1048  
[m\\_seq\\_number](#), 1050  
[m\\_snap\\_tick](#), 1051  
[m\\_song\\_mute](#), 1048  
[m\\_thru](#), 1049  
[m\\_time\\_beat\\_width](#), 1051  
[m\\_time\\_beats\\_per\\_measure](#), 1051  
[m\\_transposable](#), 1048  
[m\\_trigger\\_offset](#), 1050  
[m\\_triggers](#), 1046  
[m\\_us\\_per\\_quarter\\_note](#), 1051  
[m\\_was\\_playing](#), 1048  
[mark\\_selected](#), 1034  
[measures\\_to\\_ticks](#), 1000  
[mod\\_last\\_tick](#), 1004  
[modify](#), 996  
[move\\_selected\\_notes](#), 1029  
[move\\_selected\\_triggers\\_to](#), 1018  
[move\\_triggers](#), 1019  
[multiply\\_pattern](#), 1041  
[musical\\_key](#), 1041  
[musical\\_scale](#), 1041  
[name](#), 1002  
[note\\_off\\_margin](#), 1042  
[number](#), 996  
[off\\_playing\\_notes](#), 1035  
[off\\_queued](#), 1005  
[on\\_queued](#), 1005  
[operator=](#), 994  
[partial\\_assign](#), 994  
[paste\\_selected](#), 1026  
[paste\\_trigger](#), 1017  
[pause](#), 1036  
[perform](#), 1045  
[play](#), 1009  
[play\\_note\\_off](#), 1035

play\_note\_on, 1035  
play\_queue, 1009  
pop\_redo, 998  
pop\_trigger\_redo, 998  
pop\_trigger\_undo, 998  
pop\_undo, 998  
print, 1009  
print\_triggers, 1009  
push\_quantize, 1040  
push\_trigger\_undo, 998  
push\_undo, 997  
put\_event\_on\_bus, 1043  
quantize\_events, 1039  
remove, 1044, 1045  
remove\_all, 1045  
remove\_marked, 1034  
remove\_selected, 1034  
reset\_draw\_marker, 1036  
reset\_draw\_trigger\_marker, 1036  
select\_action\_e, 993  
select\_all, 1025  
select\_all\_notes, 1024  
select\_even\_or\_odd\_notes, 1024  
select\_event\_handle, 1023  
select\_events, 1021, 1022  
select\_linked, 1023  
select\_note\_events, 1021  
select\_trigger, 1015  
selected\_trigger\_end, 1018  
selected\_trigger\_start, 1018  
sequence, 994  
set\_32nds\_per\_quarter, 1000  
set\_beat\_width, 999  
set\_beats\_per\_bar, 999  
set\_dirty, 1008  
set\_dirty\_mp, 1008  
set\_editing, 1002  
set\_have\_redo, 997  
set\_have\_undo, 997  
set\_hold\_undo, 996  
set\_last\_tick, 1004  
set\_length, 1003  
set\_master\_midi\_bus, 1020  
set\_measures, 999  
set\_midi\_bus, 1020  
set\_midi\_channel, 1008  
set\_name, 998  
set\_parent, 1043  
set\_playing, 1004  
set\_quantized\_rec, 1006  
set\_raise, 1003  
set\_rec\_vol, 1001  
set\_recording, 1006  
set\_snap\_tick, 1006  
set\_song\_mute, 1001  
set\_thru, 1006  
set\_transposable, 1002  
set\_trigger\_offset, 1044  
set\_trigger\_paste\_tick, 995  
shift\_notes, 1041  
show\_events, 1042  
sort\_events, 1013  
split\_trigger, 1013  
stop, 1036  
stream\_event, 1029  
stretch\_selected, 1033  
toggle\_playing, 1005  
toggle\_queued, 1005  
toggle\_song\_mute, 1001  
transpose\_notes, 1040  
triggerlist, 995  
triggers, 1046  
trim\_timestamp, 1028  
unpaint\_all, 1034  
unselect, 1034  
unselect\_triggers, 1015  
us\_per\_quarter\_note, 1001  
verify\_and\_link, 1034  
zero\_markers, 1035  
seq64::trigger, 1053  
    decrement\_offset, 1056  
    decrement\_tick\_end, 1056  
    decrement\_tick\_start, 1055  
    increment\_offset, 1056  
    increment\_tick\_end, 1055  
    increment\_tick\_start, 1055  
    length, 1054  
    m\_offset, 1057  
    m\_selected, 1057  
    m\_tick\_end, 1057  
    m\_tick\_start, 1057  
    offset, 1056  
    operator<, 1054  
    selected, 1056  
    tick\_end, 1055  
    tick\_start, 1055  
    trigger, 1054  
seq64::triggers, 1057  
    ~triggers, 1061  
    add, 1063  
    adjust\_offset, 1071  
    adjust\_offsets\_to\_length, 1064  
    clear, 1069  
    copy, 1068  
    copy\_selected, 1066  
    FruityPerfInput, 1072  
    get\_maximum, 1068  
    get\_selected\_end, 1068  
    get\_selected\_start, 1067  
    get\_state, 1065  
    get\_trigger\_paste\_tick, 1071  
    grow, 1064  
    grow\_edit\_t, 1060  
    intersect, 1066  
    List, 1060  
    m\_clipboard, 1072

- m\_iterator\_draw\_trigger, 1073
- m\_iterator\_play\_trigger, 1073
- m\_length, 1073
- m\_parent, 1072
- m\_paste\_tick, 1073
- m\_ppqn, 1073
- m\_redo\_stack, 1073
- m\_trigger\_copied, 1073
- m\_triggers, 1072
- m\_undo\_stack, 1073
- midi\_container, 1071
- midifile, 1072
- move, 1068
- move\_selected, 1067
- next, 1070
- next\_trigger, 1070
- operator=, 1061
- paste, 1066
- play, 1063
- pop\_redo, 1062
- pop\_undo, 1062
- print, 1062
- push\_undo, 1062
- remove, 1065
- remove\_selected, 1066
- reset\_draw\_trigger\_marker, 1070
- select, 1065
- Seq24PerfInput, 1072
- sequence, 1072
- set\_length, 1062
- set\_ppqn, 1062
- set\_trigger\_paste\_tick, 1070
- split, 1064, 1071
- Stack, 1060
- triggerlist, 1062
- triggers, 1061
- unselect, 1066
- seq64::user\_instrument, 1074
  - controller\_active, 1077
  - controller\_count, 1076
  - controller\_max, 1076
  - controller\_name, 1076
  - copy\_definitions, 1078
  - is\_valid, 1076
  - m\_controller\_count, 1078
  - m\_instrument\_def, 1078
  - m\_is\_valid, 1078
  - name, 1076
  - operator=, 1075
  - set\_controller, 1077
  - set\_defaults, 1076
  - set\_name, 1077
  - user\_instrument, 1075
- seq64::user\_instrument\_t, 1078
  - controllers, 1079
  - controllers\_active, 1079
  - instrument, 1079
- seq64::user\_midi\_bus, 1079
  - channel\_count, 1082
  - channel\_max, 1082
  - copy\_definitions, 1083
  - instrument, 1082
  - is\_valid, 1081
  - m\_channel\_count, 1083
  - m\_is\_valid, 1083
  - m\_midi\_bus\_def, 1083
  - name, 1081
  - operator=, 1081
  - set\_defaults, 1081
  - set\_instrument, 1082
  - set\_name, 1083
  - user\_midi\_bus, 1080, 1081
- seq64::user\_midi\_bus\_t, 1084
  - alias, 1084
  - instrument, 1084
- seq64::user\_settings, 1084
  - add\_bus, 1094
  - add\_instrument, 1094
  - allow\_two\_perfedits, 1101, 1107
  - baseline\_ppqn, 1107
  - bpm\_page\_increment, 1106, 1109
  - bpm\_precision, 1106, 1108
  - bpm\_step\_increment, 1106, 1109
  - bus, 1095
  - bus\_count, 1095
  - bus\_instrument, 1095
  - bus\_name, 1095
  - BussConstIterator, 1092
  - BussIterator, 1092
  - Busses, 1092
  - control\_height, 1100, 1105
  - controller\_active, 1096
  - controller\_name, 1097
  - dump\_summary, 1105
  - global\_seq\_feature, 1100
  - gmute\_tracks, 1098
  - grid\_brackets, 1097, 1103
  - grid\_is\_black, 1097
  - grid\_is\_normal, 1097
  - grid\_is\_white, 1097
  - grid\_style, 1097, 1103
  - instrument, 1095
  - instrument\_controller\_active, 1096
  - instrument\_controller\_name, 1096
  - instrument\_count, 1095
  - instrument\_name, 1096
  - InstrumentConstIterator, 1093
  - InstrumentIterator, 1093
  - Instruments, 1093
  - inverse\_colors, 1102, 1108
  - m\_allow\_two\_perfedits, 1113
  - m\_bpm\_page\_increment, 1116
  - m\_bpm\_precision, 1116
  - m\_bpm\_step\_increment, 1116
  - m\_control\_height, 1111
  - m\_current\_zoom, 1111

- m\_global\_seq\_feature\_save, 1112
- m\_gmute\_tracks, 1117
- m\_grid\_brackets, 1110
- m\_grid\_style, 1110
- m\_h\_perf\_page\_increment, 1113
- m\_instruments, 1110
- m\_inverse\_colors, 1113
- m\_mainwid\_border, 1111
- m\_mainwid\_spacing, 1111
- m\_mainwid\_x, 1118
- m\_mainwid\_y, 1118
- m\_mainwnd\_cols, 1111
- m\_mainwnd\_rows, 1111
- m\_max\_sequence, 1117
- m\_max\_sets, 1111
- m\_midi\_beat\_width, 1115
- m\_midi\_beats\_per\_measure, 1115
- m\_midi\_beats\_per\_minute, 1115
- m\_midi\_buses, 1110
- m\_midi\_buss\_override, 1115
- m\_midi\_ppqn, 1115
- m\_progress\_bar\_colored, 1113
- m\_progress\_bar\_thick, 1113
- m\_save\_user\_config, 1118
- m\_seqarea\_seq\_x, 1117
- m\_seqarea\_seq\_y, 1117
- m\_seqarea\_x, 1117
- m\_seqarea\_y, 1117
- m\_seqchars\_x, 1114
- m\_seqchars\_y, 1114
- m\_seqedit\_bgsequence, 1112
- m\_seqedit\_key, 1112
- m\_seqedit\_scale, 1112
- m\_seqs\_in\_set, 1116
- m\_text\_x, 1114
- m\_text\_y, 1114
- m\_total\_seqs, 1116
- m\_use\_more\_icons, 1114
- m\_use\_new\_font, 1112
- m\_v\_perf\_page\_increment, 1113
- m\_velocity\_override, 1115
- m\_window\_redraw\_rate\_ms, 1113
- mainwid\_border, 1099, 1105
- mainwid\_grid\_style\_t, 1093
- mainwid\_spacing, 1099, 1105
- mainwid\_x, 1099
- mainwid\_y, 1100
- mainwnd\_cols, 1098, 1103
- mainwnd\_rows, 1097, 1103
- max\_sequence, 1098
- max\_sets, 1098, 1103
- max\_zoom, 1106
- mc\_baseline\_ppqn, 1118
- mc\_max\_zoom, 1118
- mc\_min\_zoom, 1118
- midi\_beat\_width, 1106, 1109
- midi\_beats\_per\_bar, 1105, 1109
- midi\_beats\_per\_minute, 1105, 1109
- midi\_buss\_override, 1106, 1108
- midi\_ppqn, 1105, 1108
- min\_zoom, 1106
- normalize, 1094
- operator=, 1094
- perf\_h\_page\_increment, 1102, 1107
- perf\_v\_page\_increment, 1102, 1107
- private\_bus, 1109
- private\_instrument, 1109
- progress\_bar\_colored, 1102, 1107
- progress\_bar\_thick, 1102, 1107
- save\_user\_config, 1102, 1103
- seqarea\_seq\_x, 1099, 1104
- seqarea\_seq\_y, 1099, 1104
- seqarea\_x, 1099, 1104
- seqarea\_y, 1099, 1104
- seqchars\_x, 1098, 1104
- seqchars\_y, 1099, 1104
- seqedit\_bgsequence, 1101
- seqedit\_key, 1101
- seqedit\_scale, 1100, 1101
- seqs\_in\_set, 1098
- set\_bus\_instrument, 1095
- set\_defaults, 1094
- set\_instrument\_controllers, 1096
- text\_x, 1098, 1103
- text\_y, 1098, 1104
- use\_more\_icons, 1102, 1108
- use\_new\_font, 1101, 1107
- user\_settings, 1093, 1094
- userfile, 1110
- velocity\_override, 1106, 1108
- window\_redraw\_rate, 1102, 1108
- zoom, 1100
- seq64::userfile, 1119
  - ~userfile, 1120
  - dump\_setting\_summary, 1121
  - parse, 1120
  - userfile, 1120
  - write, 1121
- seq\_app\_name
  - seq64, 101
- seq\_clear\_perf
  - seq64::seqmenu, 940
- seq\_client\_name
  - seq64, 101
- seq\_copy
  - seq64::seqmenu, 939
- seq\_cut
  - seq64::seqmenu, 939
- seq\_edit
  - seq64::seqmenu, 937
- seq\_event\_edit
  - seq64::seqmenu, 937
- seq\_event\_type\_t
  - seq64, 67
- seq\_from\_xy
  - seq64::mainwid, 408

- seq\_in\_playing\_screen
  - seq64::perform, [740](#)
- seq\_modifier\_t
  - seq64, [66](#)
- seq\_new
  - seq64::seqmenu, [939](#)
- seq\_number\_size
  - seq64::midifile, [634](#)
- seq\_paste
  - seq64::seqmenu, [940](#)
- seq\_scroll\_direction\_t
  - seq64, [67](#)
- seq\_set\_and\_edit
  - seq64::mainwid, [406](#)
  - seq64::seqmenu, [938](#)
- seq\_set\_and\_eventedit
  - seq64::mainwid, [406](#)
  - seq64::seqmenu, [939](#)
- seq\_version
  - seq64, [101](#)
- seqarea\_seq\_x
  - seq64::user\_settings, [1099](#), [1104](#)
- seqarea\_seq\_y
  - seq64::user\_settings, [1099](#), [1104](#)
- seqarea\_x
  - seq64::user\_settings, [1099](#), [1104](#)
- seqarea\_y
  - seq64::user\_settings, [1099](#), [1104](#)
- seqchars\_x
  - seq64::user\_settings, [1098](#), [1104](#)
- seqchars\_y
  - seq64::user\_settings, [1099](#), [1104](#)
- seqdata
  - seq64::seqdata, [861](#)
- seqedit
  - seq64::seqedit, [879](#)
  - seq64::seqmenu, [941](#)
- seqedit\_bgsequence
  - seq64::user\_settings, [1101](#)
- seqedit\_key
  - seq64::user\_settings, [1101](#)
- seqedit\_scale
  - seq64::user\_settings, [1100](#), [1101](#)
- SeqeditMap
  - seq64::seqmenu, [934](#)
- SeqeditPair
  - seq64::seqmenu, [934](#)
- seqevent
  - seq64::seqdata, [869](#)
  - seq64::seqevent, [907](#)
- seqkeys
  - seq64::seqkeys, [921](#)
- seqmenu
  - seq64::seqedit, [892](#)
  - seq64::seqmenu, [934](#)
- seqroll
  - seq64::seqdata, [869](#)
  - seq64::seqkeys, [928](#)
  - seq64::seqroll, [949](#)
- seqs\_in\_set
  - seq64::user\_settings, [1098](#)
- seqspec\_string
  - seq64::editable\_event, [176](#)
- seqtime
  - seq64::seqtime, [978](#)
- sequence
  - seq64::event\_list, [222](#)
  - seq64::sequence, [994](#)
  - seq64::triggers, [1072](#)
- sequence\_count
  - seq64::perform, [692](#)
- sequence\_key
  - seq64::mainwnd, [425](#)
  - seq64::perform, [720](#)
- sequence\_label
  - seq64::perform, [720](#)
- sequence\_max
  - seq64::perform, [692](#)
- sequence\_playing\_change
  - seq64::perform, [708](#)
- sequence\_playing\_off
  - seq64::perform, [709](#)
- sequence\_playing\_on
  - seq64::perform, [708](#)
- sequence\_playing\_toggle
  - seq64::perform, [708](#)
- session\_event
  - seq64::jack\_assistant, [322](#)
- set
  - seq64::keybindentry, [343](#)
  - seq64::midi\_control, [504](#)
- set\_32nds\_per\_quarter
  - seq64::perform, [694](#)
  - seq64::sequence, [1000](#)
- set\_active
  - seq64::perform, [737](#)
- set\_adding
  - seq64::AbstractPerfInput, [136](#)
  - seq64::Seq24SeqEventInput, [856](#)
  - seq64::seqroll, [961](#)
- set\_adding\_pressed
  - seq64::AbstractPerfInput, [136](#)
- set\_all\_clocks
  - seq64::busarray, [143](#)
- set\_all\_inputs
  - seq64::busarray, [146](#)
- set\_all\_key\_events
  - seq64::keys\_perform, [364](#)
  - seq64::keys\_perform\_gtk2, [376](#)
- set\_all\_key\_groups
  - seq64::keys\_perform, [364](#)
  - seq64::keys\_perform\_gtk2, [377](#)
- set\_alsa\_client\_id
  - seq64::lash, [387](#)
- set\_alt\_name
  - seq64::midibase, [593](#)



- set\_and\_copy\_mute\_group
  - seq64::perform, [734](#)
- set\_api
  - seq64::rtmidi, [829](#)
- set\_api\_info
  - seq64::rtmidi\_info, [847](#)
- set\_background\_sequence
  - seq64::seqedit, [884](#)
  - seq64::seqroll, [952](#)
- set\_beat\_width
  - seq64::jack\_assistant, [319](#)
  - seq64::perfedit, [658](#)
  - seq64::perform, [694](#)
  - seq64::seqedit, [881](#)
  - seq64::sequence, [999](#)
- set\_beats\_per\_bar
  - seq64::perfedit, [658](#)
  - seq64::perform, [693](#)
  - seq64::seqedit, [880](#)
  - seq64::sequence, [999](#)
- set\_beats\_per\_measure
  - seq64::jack\_assistant, [320](#)
- set\_beats\_per\_minute
  - seq64::jack\_assistant, [320](#)
  - seq64::mastermidibase, [450](#)
  - seq64::perform, [738](#)
- set\_bus\_and\_midi\_channel
  - seq64::seqmenu, [940](#)
- set\_bus\_id
  - seq64::midibase, [592](#)
- set\_bus\_instrument
  - seq64::user\_settings, [1095](#)
- set\_channel
  - seq64::event, [199](#)
- set\_chord
  - seq64::seqedit, [884](#)
  - seq64::seqroll, [952](#)
- set\_clock
  - seq64::busarray, [143](#)
  - seq64::mastermidibase, [443](#), [446](#)
  - seq64::midibase, [591](#)
  - seq64::perform, [746](#)
- set\_clock\_bus
  - seq64::perform, [721](#)
- set\_clock\_mod
  - seq64::midibase, [594](#)
- set\_clock\_status
  - seq64::midibase, [591](#)
- set\_config\_files
  - seq64::rc\_settings, [810](#)
- set\_controller
  - seq64::user\_instrument, [1077](#)
- set\_current\_drop\_x
  - seq64::gui\_drawingarea\_gtk2, [295](#)
- set\_current\_drop\_y
  - seq64::gui\_drawingarea\_gtk2, [295](#)
- set\_current\_event
  - seq64::eventslots, [245](#)
- set\_current\_offset\_x\_y
  - seq64::seqroll, [965](#)
- set\_data
  - seq64::event, [200](#)
- set\_data\_type
  - seq64::seqdata, [863](#)
  - seq64::seqedit, [886](#)
  - seq64::seqevent, [908](#)
  - seq64::seqroll, [952](#)
- set\_defaults
  - seq64::rc\_settings, [801](#)
  - seq64::user\_instrument, [1076](#)
  - seq64::user\_midi\_bus, [1081](#)
  - seq64::user\_settings, [1094](#)
- set\_dirty
  - seq64::eventedit, [231](#)
  - seq64::sequence, [1008](#)
- set\_dirty\_mp
  - seq64::sequence, [1008](#)
- set\_edit\_sequence
  - seq64::perform, [692](#)
  - seq64::seqmenu, [935](#)
- set\_editing
  - seq64::sequence, [1002](#)
- set\_error\_message
  - seq64::configfile, [165](#)
- set\_event\_category
  - seq64::eventedit, [230](#)
- set\_event\_data\_0
  - seq64::eventedit, [230](#)
- set\_event\_data\_1
  - seq64::eventedit, [230](#)
- set\_event\_name
  - seq64::eventedit, [230](#)
- set\_event\_timestamp
  - seq64::eventedit, [230](#)
- set\_follow\_transport
  - seq64::jack\_assistant, [326](#)
  - seq64::perfedit, [657](#)
  - seq64::perform, [699](#)
- set\_group\_mute\_state
  - seq64::perform, [711](#)
- set\_guides
  - seq64::perfedit, [659](#)
  - seq64::perffroll, [769](#)
  - seq64::perftime, [787](#)
- set\_have\_redo
  - seq64::perform, [727](#)
  - seq64::sequence, [997](#)
- set\_have\_undo
  - seq64::perform, [727](#)
  - seq64::sequence, [997](#)
- set\_hint\_key
  - seq64::seqkeys, [922](#)
- set\_hint\_state
  - seq64::seqkeys, [923](#)
- set\_hold\_undo
  - seq64::sequence, [996](#)

- set\_image
  - seq64::perfedit, 660
- set\_input
  - seq64::busarray, 145
  - seq64::mastermidibase, 447
  - seq64::midibase, 598
  - seq64::perform, 746
- set\_input\_bus
  - seq64::perform, 721
- set\_input\_status
  - seq64::midibase, 592
- set\_instrument
  - seq64::user\_midi\_bus, 1082
- set\_instrument\_controllers
  - seq64::user\_settings, 1096
- set\_jack\_mode
  - seq64::jack\_assistant, 325
  - seq64::perfedit, 657
  - seq64::perform, 697
- set\_jack\_running
  - seq64::jack\_assistant, 327
- set\_jack\_stop\_tick
  - seq64::jack\_assistant, 325
  - seq64::perform, 697
- set\_jack\_tick
  - seq64::perform, 703
- set\_key
  - seq64::seqedit, 884
  - seq64::seqkeys, 922
  - seq64::seqroll, 952
- set\_key\_event
  - seq64::keys\_perform, 365
  - seq64::perform, 745
- set\_key\_group
  - seq64::keys\_perform, 365
  - seq64::perform, 745
- set\_keys
  - seq64::keys\_perform, 352
- set\_last\_tick
  - seq64::sequence, 1004
- set\_left\_tick
  - seq64::perform, 704
- set\_length
  - seq64::sequence, 1003
  - seq64::triggers, 1062
- set\_line
  - seq64::gui\_drawingarea\_gtk2, 285
- set\_listen\_button\_press
  - seq64::seqkeys, 923
- set\_listen\_button\_release
  - seq64::seqkeys, 923
- set\_listen\_motion\_notify
  - seq64::seqkeys, 923
- set\_looped
  - seq64::perfedit, 659
- set\_looping
  - seq64::perform, 739
- set\_master\_midi\_bus
  - seq64::sequence, 1020
- set\_measures
  - seq64::seqedit, 883
  - seq64::sequence, 999
- set\_midi\_bus
  - seq64::seqedit, 884
  - seq64::sequence, 1020
- set\_midi\_channel
  - seq64::seqedit, 883
  - seq64::sequence, 1008
- set\_mode\_group\_learn
  - seq64::perform, 734
- set\_mode\_group\_mute
  - seq64::perform, 733
- set\_multi\_name
  - seq64::midibase, 593
- set\_name
  - seq64::midibase, 592
  - seq64::sequence, 998
  - seq64::user\_instrument, 1077
  - seq64::user\_midi\_bus, 1083
- set\_note
  - seq64::event, 205
- set\_note\_length
  - seq64::seqedit, 880
  - seq64::seqroll, 951
- set\_note\_velocity
  - seq64::event, 206
- set\_offset
  - seq64::perform, 712
- set\_orig\_ticks
  - seq64::perform, 738
- set\_parent
  - seq64::sequence, 1043
- set\_play\_image
  - seq64::mainwnd, 423
- set\_playback\_mode
  - seq64::perform, 742
- set\_playing
  - seq64::sequence, 1004
- set\_playing\_screensets
  - seq64::keys\_perform, 356
  - seq64::perform, 732
- set\_port\_id
  - seq64::midibase, 598
- set\_port\_open
  - seq64::midi\_api, 490
- set\_position
  - seq64::jack\_assistant, 330
- set\_ppqn
  - seq64::jack\_assistant, 324
  - seq64::mastermidibase, 448
  - seq64::perfroll, 771
  - seq64::perftime, 788
  - seq64::triggers, 1062
- set\_quantized\_rec
  - seq64::sequence, 1006
- set\_raise

- seq64::sequence, [1003](#)
- set\_rec\_vol
  - seq64::seqedit, [881](#)
  - seq64::sequence, [1001](#)
- set\_recording
  - seq64::sequence, [1006](#)
- set\_reposition
  - seq64::perform, [699](#)
- set\_right\_tick
  - seq64::perform, [704](#)
- set\_running
  - seq64::perform, [741](#)
- set\_scale
  - seq64::perftime, [787](#)
  - seq64::seqedit, [884](#)
  - seq64::seqkeys, [922](#)
  - seq64::seqroll, [952](#)
- set\_screen\_set\_notepad
  - seq64::perform, [731](#), [732](#)
- set\_screenshot
  - seq64::mainwid, [405](#)
  - seq64::perform, [726](#)
- set\_seq\_count
  - seq64::eventedit, [229](#)
- set\_seq\_ppqn
  - seq64::eventedit, [229](#)
- set\_seq\_time\_sig
  - seq64::eventedit, [229](#)
- set\_seq\_title
  - seq64::eventedit, [229](#)
- set\_sequence\_control\_status
  - seq64::perform, [706](#)
- set\_sequence\_input
  - seq64::mastermidibase, [444](#)
- set\_snap
  - seq64::perfedit, [658](#)
  - seq64::seqedit, [880](#)
  - seq64::seqevent, [908](#)
  - seq64::seqroll, [950](#)
- set\_snap\_tick
  - seq64::sequence, [1006](#)
- set\_song\_mute
  - seq64::mainwnd, [426](#)
  - seq64::perform, [733](#)
  - seq64::sequence, [1001](#)
- set\_sONGLIVE\_image
  - seq64::mainwnd, [423](#)
- set\_start\_from\_perfedit
  - seq64::jack\_assistant, [326](#)
- set\_start\_tick
  - seq64::perform, [704](#)
- set\_status
  - seq64::event, [198](#), [199](#)
- set\_status\_from\_string
  - seq64::editable\_event, [175](#)
- set\_status\_keep\_channel
  - seq64::event, [199](#)
- set\_sysex\_size
  - seq64::event, [203](#)
- set\_system\_port\_flag
  - seq64::midibase, [591](#)
- set\_text
  - seq64::eventslots, [249](#)
- set\_thru
  - seq64::sequence, [1006](#)
- set\_tick
  - seq64::perform, [703](#)
- set\_timestamp
  - seq64::event, [194](#)
- set\_transposable
  - seq64::seqmenu, [940](#)
  - seq64::sequence, [1002](#)
- set\_transpose
  - seq64::perfedit, [657](#)
  - seq64::perform, [706](#)
- set\_transpose\_image
  - seq64::seqedit, [881](#)
- set\_trigger\_offset
  - seq64::sequence, [1044](#)
- set\_trigger\_paste\_tick
  - seq64::sequence, [995](#)
  - seq64::triggers, [1070](#)
- set\_virtual\_name
  - seq64::midi\_alsa, [474](#)
  - seq64::midi\_jack, [535](#)
  - seq64::midibus, [612](#)
- set\_was\_active
  - seq64::perform, [737](#)
- set\_zoom
  - seq64::perfedit, [656](#)
  - seq64::perfroll, [774](#)
  - seq64::perftime, [788](#)
  - seq64::seqdata, [862](#)
  - seq64::seqedit, [879](#)
  - seq64::seqevent, [908](#)
  - seq64::seqroll, [950](#)
  - seq64::seqtime, [979](#)
- shift\_lock
  - seq64::keystroke, [385](#)
- shift\_notes
  - seq64::sequence, [1041](#)
- shorten\_file\_spec
  - seq64, [76](#)
- show\_bus\_values
  - seq64::midibase, [589](#)
- show\_events
  - seq64::sequence, [1042](#)
- show\_jack\_statuses
  - seq64, [96](#)
- show\_midi
  - seq64::rc\_settings, [802](#), [806](#)
- show\_position
  - seq64::jack\_assistant, [328](#)
- show\_ui\_sequence\_key
  - seq64::keys\_perform, [362](#)
  - seq64::perform, [713](#)

- show\_ui\_sequence\_number
  - seq64::keys\_perform, [362](#)
  - seq64::perform, [713](#), [714](#)
- signal
  - seq64::condition\_var, [160](#)
- signal\_action
  - seq64::mainwnd, [428](#)
- silence\_jack\_errors
  - seq64, [106](#)
- silence\_jack\_info
  - seq64, [106](#)
- size
  - seq64::midi\_container, [495](#)
  - seq64::midi\_list, [549](#)
  - seq64::midi\_vector, [581](#)
- SlotMap
  - seq64::keys\_perform, [351](#)
- sm\_category\_arrays
  - seq64::editable\_event, [177](#)
- sm\_category\_names
  - seq64::editable\_event, [177](#)
- sm\_channel\_event\_names
  - seq64::editable\_event, [177](#)
- sm\_cond
  - seq64::condition\_var, [161](#)
- sm\_input\_caps
  - seq64::midi\_alsa\_info, [481](#)
- sm\_internal\_keys
  - seq64::gui\_assistant\_gtk2, [279](#)
- sm\_mc\_dummy
  - seq64::perform, [751](#)
- sm\_meta\_event\_names
  - seq64::editable\_event, [177](#)
- sm\_output\_caps
  - seq64::midi\_alsa\_info, [481](#)
- sm\_prop\_event\_names
  - seq64::editable\_event, [177](#)
- sm\_recursive\_mutex
  - seq64::mutex, [638](#)
- sm\_selected\_api
  - seq64::rtmidi\_info, [849](#)
- sm\_seqedit\_list
  - seq64::seqmenu, [942](#)
- sm\_status\_pairs
  - seq64::jack\_assistant, [335](#)
- sm\_system\_event\_names
  - seq64::editable\_event, [177](#)
- snap\_x
  - seq64::perfroll, [773](#)
  - seq64::seqevent, [912](#)
  - seq64::seqroll, [956](#)
- snap\_y
  - seq64::seqevent, [912](#)
  - seq64::seqroll, [956](#)
- snapshot\_1
  - seq64::keys\_perform, [354](#)
- snapshot\_2
  - seq64::keys\_perform, [355](#)
- song\_fill\_seq\_event
  - seq64::midi\_container, [500](#)
- song\_fill\_seq\_trigger
  - seq64::midi\_container, [500](#)
- song\_mode
  - seq64::keys\_perform, [359](#)
- song\_start\_mode
  - seq64::jack\_assistant, [326](#)
  - seq64::perform, [696](#)
- sort
  - seq64::event\_list, [218](#)
- sort\_events
  - seq64::sequence, [1013](#)
- split
  - seq64::midi\_splitter, [572](#)
  - seq64::triggers, [1064](#), [1071](#)
- split\_channel
  - seq64::midi\_splitter, [572](#)
- split\_trigger
  - seq64::perform, [728](#)
  - seq64::perfroll, [774](#)
  - seq64::sequence, [1013](#)
- Stack
  - seq64::triggers, [1060](#)
- start
  - seq64::busarray, [142](#)
  - seq64::businfo, [152](#)
  - seq64::jack\_assistant, [322](#)
  - seq64::keys\_perform, [357](#)
  - seq64::lash, [387](#)
  - seq64::mastermidibase, [441](#)
  - seq64::midibase, [596](#)
  - seq64::perform, [735](#)
- start\_from\_perfeddit
  - seq64::perform, [698](#)
- start\_jack
  - seq64::perform, [736](#)
- start\_key
  - seq64::perform, [717](#)
- start\_paste
  - seq64::seqevent, [911](#)
  - seq64::seqroll, [955](#)
- start\_playing
  - seq64::mainwnd, [423](#)
  - seq64::perfedit, [661](#)
  - seq64::perform, [715](#)
  - seq64::seqedit, [889](#)
- stats
  - seq64::rc\_settings, [803](#), [807](#)
- status
  - seq64::midi\_control, [503](#)
- status\_string
  - seq64::editable\_event, [175](#)
- stock\_event\_string
  - seq64::editable\_event, [175](#)
- stop
  - seq64::busarray, [142](#)
  - seq64::businfo, [152](#)

- seq64::jack\_assistant, [322](#)
- seq64::keys\_perform, [359](#)
- seq64::mastermidibase, [441](#)
- seq64::midibase, [596](#)
- seq64::perform, [735](#)
- seq64::sequence, [1036](#)
- stop\_jack
  - seq64::perform, [736](#)
- stop\_key
  - seq64::perform, [717](#)
- stop\_playing
  - seq64::mainwnd, [424](#)
  - seq64::perfedit, [661](#)
  - seq64::perform, [717](#)
  - seq64::seqedit, [889](#)
- stream\_event
  - seq64::sequence, [1029](#)
- stretch\_selected
  - seq64::sequence, [1033](#)
- string\_is\_void
  - seq64, [77](#)
- string\_not\_void
  - seq64, [76](#)
- string\_to\_midibyte
  - seq64, [76](#)
- string\_to\_pulses
  - seq64, [75](#)
  - seq64::editable\_events, [183](#)
- strings\_match
  - seq64, [77](#)
- swap
  - seq64, [70](#)
- sync
  - seq64::jack\_assistant, [329](#)
- sysex
  - seq64::busarray, [143](#)
  - seq64::businfo, [153](#)
  - seq64::mastermidibase, [443](#)
  - seq64::midibase, [596](#)
- SysexContainer
  - seq64::event, [192](#)
- tap\_bpm
  - seq64::keys\_perform, [361](#)
- tempo\_us\_from\_bpm
  - seq64, [79](#)
- tempo\_us\_to\_bytes
  - seq64, [78](#)
- test\_ignore\_flags
  - seq64::rtmidi\_in\_data, [835](#)
- test\_widget\_click
  - seq64, [105](#)
- text\_x
  - seq64::user\_settings, [1098](#), [1103](#)
- text\_y
  - seq64::user\_settings, [1098](#), [1104](#)
- thru\_change\_callback
  - seq64::seqedit, [885](#)
- tick\_end
  - seq64::trigger, [1055](#)
- tick\_multiplier
  - seq64::jack\_assistant, [327](#)
- tick\_offset
  - seq64::perftime, [789](#)
- tick\_start
  - seq64::trigger, [1055](#)
- tick\_to\_pixel
  - seq64::perftime, [789](#)
- ticks\_to\_delta\_time\_us
  - seq64, [81](#)
- time\_as\_measures
  - seq64::editable\_event, [174](#)
- time\_as\_minutes
  - seq64::editable\_event, [174](#)
- time\_as\_pulses
  - seq64::editable\_event, [174](#)
- timeout
  - seq64::mainwid, [408](#)
  - seq64::perfedit, [660](#)
  - seq64::seqedit, [888](#)
- timer\_callback
  - seq64::mainwnd, [422](#)
- timestamp
  - seq64::editable\_event, [174](#)
  - seq64::midi\_message, [555](#)
- timestamp\_format\_t
  - seq64::editable\_event, [170](#)
- timestamp\_string
  - seq64::editable\_event, [173](#)
- timestring\_to\_pulses
  - seq64, [75](#), [100](#)
- timing
  - seq64::editable\_events, [183](#)
- to\_string
  - seq64, [89](#)
- toLower
  - seq64::mainwnd, [425](#)
- toggle\_all\_tracks
  - seq64::perform, [709](#)
  - seq64::seqmenu, [941](#)
- toggle\_current\_sequence
  - seq64::seqmenu, [937](#)
- toggle\_follow\_transport
  - seq64::jack\_assistant, [326](#)
  - seq64::perfedit, [657](#)
  - seq64::perform, [699](#)
- toggle\_jack
  - seq64::keys\_perform, [361](#)
  - seq64::perfedit, [656](#)
- toggle\_jack\_mode
  - seq64::jack\_assistant, [325](#)
  - seq64::perform, [697](#)
- toggle\_mutes
  - seq64::keys\_perform, [361](#)
- toggle\_other\_names
  - seq64::perform, [727](#)
- toggle\_other\_seqs

- seq64::perform, 726
- toggle\_playing
  - seq64::mainwnd, 424
  - seq64::perfedit, 661
  - seq64::sequence, 1005
- toggle\_playing\_tracks
  - seq64::perform, 709
- toggle\_queued
  - seq64::sequence, 1005
- toggle\_song\_mute
  - seq64::sequence, 1001
- toggle\_song\_start\_mode
  - seq64::jack\_assistant, 326
  - seq64::perform, 696
- toggle\_visible
  - seq64::lfownd, 392
- top\_index
  - seq64::eventslots, 244
- track\_end\_size
  - seq64::midifile, 634
- track\_name\_size
  - seq64::midifile, 633
- transport\_callback
  - seq64::options, 642
- transport\_not\_starting
  - seq64::jack\_assistant, 320
- transport\_state
  - seq64::jack\_assistant, 320
- transpose\_button\_callback
  - seq64::perfedit, 658
- transpose\_change\_callback
  - seq64::seqedit, 885
- transpose\_note
  - seq64::event, 205
- transpose\_notes
  - seq64::sequence, 1040
- trigger
  - seq64::trigger, 1054
- triggerlist
  - seq64::sequence, 995
  - seq64::triggers, 1062
- triggers
  - seq64::sequence, 1046
  - seq64::triggers, 1061
- trim\_timestamp
  - seq64::sequence, 1028
- Type
  - seq64::rterror, 820
- type
  - seq64::keybindentry, 342
- undo
  - seq64::perfedit, 660
- undo\_callback
  - seq64::seqedit, 886
- unlock
  - seq64::mutex, 638
- unmark
  - seq64::event, 204
- unmark\_all
  - seq64::event\_list, 220
- unmodify
  - seq64::event\_list, 217
- unmute\_all\_tracks
  - seq64::seqmenu, 941
- unpaint
  - seq64::event, 204
- unpaint\_all
  - seq64::event\_list, 220
  - seq64::sequence, 1034
- unselect
  - seq64::event, 204
  - seq64::sequence, 1034
  - seq64::triggers, 1066
- unselect\_all
  - seq64::event\_list, 221
- unselect\_triggers
  - seq64::sequence, 1015
- unset\_edit\_sequence
  - seq64::perform, 693
  - seq64::seqmenu, 935
- unset\_mode\_group\_learn
  - seq64::perform, 734
- unset\_mode\_group\_mute
  - seq64::perform, 733
- unset\_sequence\_control\_status
  - seq64::perform, 706
- update\_all\_windows
  - seq64::seqedit, 886
- update\_and\_draw
  - seq64::seqroll, 954
- update\_background
  - seq64::seqroll, 953
- update\_mainwid\_sequences
  - seq64, 105
  - seq64::mainwid, 412
- update\_markers
  - seq64::mainwid, 407
- update\_mouse\_pointer
  - seq64::FruityPerfInput, 266
  - seq64::FruitySeqEventInput, 269
  - seq64::FruitySeqRollInput, 272
  - seq64::seqroll, 961
- update\_perfedit\_sequences
  - seq64, 105
  - seq64::perfedit, 662
- update\_pixmap
  - seq64::perftime, 790
  - seq64::seqdata, 863
  - seq64::seqevent, 909
  - seq64::seqkeys, 924
  - seq64::seqroll, 953
  - seq64::seqtime, 980
- update\_sequences\_on\_window
  - seq64::mainwid, 405
- update\_sizes
  - seq64::perfroll, 770

- seq64::perftime, 790
- seq64::seqdata, 863
- seq64::seqevent, 909
- seq64::seqkeys, 925
- seq64::seqroll, 953
- seq64::seqtime, 980
- update\_window\_title
  - seq64::mainwnd, 425
- us\_per\_quarter\_note
  - seq64::perform, 694, 695
  - seq64::sequence, 1001
- use\_more\_icons
  - seq64::user\_settings, 1102, 1108
- use\_new\_font
  - seq64::user\_settings, 1101, 1107
- user\_callback
  - seq64::midi\_api, 490
  - seq64::rtmidi\_in, 833
  - seq64::rtmidi\_in\_data, 838
- user\_data
  - seq64::rtmidi\_in\_data, 837, 838
- user\_filename
  - seq64::rc\_settings, 805, 810
- user\_filename\_alt
  - seq64::rc\_settings, 805, 811
- user\_filespec
  - seq64::rc\_settings, 801
- user\_instrument
  - seq64::user\_instrument, 1075
- user\_midi\_bus
  - seq64::user\_midi\_bus, 1080, 1081
- user\_settings
  - seq64::user\_settings, 1093, 1094
- userfile
  - seq64::user\_settings, 1110
  - seq64::userfile, 1120
- using\_callback
  - seq64::rtmidi\_in\_data, 837
- usr
  - seq64, 99
- v\_adjustment
  - seq64::eventedit, 231
- valid\_buffer
  - seq64::midi\_jack\_data, 537
- valid\_midi\_control\_seq
  - seq64::perform, 741
- valid\_sequence
  - seq64::mainwid, 407
- value\_to\_name
  - seq64::editable\_event, 172
- varinum\_size
  - seq64::midifile, 632
- velocity\_override
  - seq64::user\_settings, 1106, 1108
- verify\_and\_link
  - seq64::event\_list, 219
  - seq64::sequence, 1034
- versiontext
  - seq64, 124
- vertical\_adjust
  - seq64::perfrroll, 775
  - seq64::seqedit, 882
  - seq64::seqroll, 956
- vertical\_set
  - seq64::perfrroll, 775
  - seq64::seqedit, 882
- wait
  - seq64::condition\_var, 160
- wave\_func
  - seq64, 83
- wave\_type\_name
  - seq64, 71
- wave\_type\_t
  - seq64, 66
- what
  - seq64::rterror, 821
- white
  - seq64::gui\_palette\_gtk2, 304
- white\_key
  - seq64::gui\_palette\_gtk2, 305
- white\_paint
  - seq64::gui\_palette\_gtk2, 305
- width
  - seq64::gui\_drawingarea\_gtk2::rect, 819
  - seq64::rect, 818
- window\_redraw\_rate
  - seq64::user\_settings, 1102, 1108
- window\_x
  - seq64::gui\_drawingarea\_gtk2, 284
- window\_y
  - seq64::gui\_drawingarea\_gtk2, 284
- with\_jack
  - seq64::rc\_settings, 803
- with\_jack\_master
  - seq64::rc\_settings, 803, 807
- with\_jack\_master\_cond
  - seq64::rc\_settings, 803, 807
- with\_jack\_midi
  - seq64::rc\_settings, 803, 807
- with\_jack\_transport
  - seq64::rc\_settings, 803, 807
- write
  - seq64::configfile, 164
  - seq64::midifile, 621
  - seq64::optionsfile, 649
  - seq64::userfile, 1121
- write\_byte
  - seq64::midifile, 628
- write\_header
  - seq64::midifile, 630
- write\_long
  - seq64::midifile, 627
- write\_options\_files
  - seq64, 87
- write\_prop\_header
  - seq64::midifile, 630

- write\_proprietary\_track
  - seq64::midifile, [631](#)
- write\_seq\_number
  - seq64::midifile, [629](#)
- write\_short
  - seq64::midifile, [627](#)
- write\_song
  - seq64::midifile, [622](#)
- write\_track
  - seq64::midifile, [634](#)
- write\_track\_end
  - seq64::midifile, [630](#)
- write\_track\_name
  - seq64::midifile, [629](#)
- write\_varinum
  - seq64::midifile, [628](#)
- x
  - seq64::click, [157](#)
  - seq64::gui\_drawingarea\_gtk2::rect, [818](#)
  - seq64::rect, [817](#)
- x\_to\_w
  - seq64::seqevent, [910](#)
- xy\_to\_rect
  - seq64::seqdata, [863](#)
  - seq64::seqroll, [957](#)
- y
  - seq64::click, [157](#)
  - seq64::gui\_drawingarea\_gtk2::rect, [818](#)
  - seq64::rect, [817](#)
- yellow
  - seq64::gui\_palette\_gtk2, [305](#)
- zero\_markers
  - seq64::sequence, [1035](#)
- zoom
  - seq64::user\_settings, [1100](#)
- zoom\_check
  - seq64::perfedit, [655](#)
- zoom\_power\_of\_2
  - seq64, [78](#)