

Sequencer64 Developer Reference Manual

0.9.16

Generated by Doxygen 1.8.11

Contents

1	Sequencer64	1
1.1	Introduction	1
2	MIDI File Parsing in Sequencer64	3
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

3 JACK, Live, and Song Modes in Sequencer64	13
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
4 User Testing of Sequencer64 with Yoshimi	19
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

5	Licenses	25
5.1	License Terms for the This Project.	25
5.2	XPC Application License	25
5.3	XPC Library License	26
5.4	XPC Documentation License	26
5.5	XPC Affero License	27
5.6	XPC License Summary	27
6	Todo List	29
7	Deprecated List	31
8	Namespace Index	33
8.1	Namespace List	33
9	Hierarchical Index	35
9.1	Class Hierarchy	35
10	Data Structure Index	37
10.1	Data Structures	37
11	Namespace Documentation	41
11.1	Gtk Namespace Reference	41
11.2	seq64 Namespace Reference	41
11.2.1	Detailed Description	51
11.2.2	Typedef Documentation	51
11.2.2.1	midibyte	51
11.2.2.2	bussbyte	51
11.2.2.3	midishort	51
11.2.2.4	midilong	51
11.2.2.5	midipulse	51
11.2.3	Enumeration Type Documentation	51
11.2.3.1	seq_modifier_t	51
11.2.3.2	seq_event_type_t	52

11.2.3.3	<code>seq_scroll_direction_t</code>	52
11.2.3.4	<code>clock_e</code>	53
11.2.3.5	<code>interaction_method_t</code>	53
11.2.3.6	<code>c_music_scales</code>	53
11.2.3.7	<code>draw_type</code>	54
11.2.3.8	<code>mouse_action_e</code>	54
11.2.4	Function Documentation	54
11.2.4.1	<code>extract_timing_numbers(const std::string &s, std::string &part_1, std::string &part_2, std::string &part_3, std::string &fraction)</code>	54
11.2.4.2	<code>pulses_to_string(midipulse p)</code>	55
11.2.4.3	<code>pulses_to_measurestring(midipulse p, const midi_timing &seqparms)</code>	55
11.2.4.4	<code>pulses_to_midi_measures(midipulse p, const midi_timing &seqparms, midi_← measures &measures)</code>	55
11.2.4.5	<code>pulses_to_timestring(midipulse p, int bpm, int ppqn)</code>	56
11.2.4.6	<code>pulses_to_timestring(midipulse p, const midi_timing &timinginfo)</code>	56
11.2.4.7	<code>measurestring_to_pulses(const std::string &measures, const midi_timing &seq- parms)</code>	56
11.2.4.8	<code>midi_measures_to_pulses(const midi_measures &measures, const midi_timing &seqparms)</code>	57
11.2.4.9	<code>timestring_to_pulses(const std::string &timestring, int bpm, int ppqn)</code>	57
11.2.4.10	<code>string_to_pulses(const std::string &s, const midi_timing &mt)</code>	57
11.2.4.11	<code>string_to_midibyte(const std::string &s)</code>	58
11.2.4.12	<code>shorten_file_spec(const std::string &fpath, int leng)</code>	58
11.2.4.13	<code>string_not_void(const std::string &s)</code>	58
11.2.4.14	<code>string_is_void(const std::string &s)</code>	59
11.2.4.15	<code>strings_match(const std::string &target, const std::string &x)</code>	59
11.2.4.16	<code>log2_time_sig_value(int tsd)</code>	59
11.2.4.17	<code>tempo_to_bytes(midibyte t[3], int tempo_us)</code>	60
11.2.4.18	<code>zoom_power_of_2(int ppqn)</code>	60
11.2.4.19	<code>beats_per_minute_from_tempo(double tempo)</code>	60
11.2.4.20	<code>tempo_from_beats_per_minute(double bpm)</code>	61
11.2.4.21	<code>pulse_length_us(int bpm, int ppqn)</code>	61

11.2.4.22 <code>delta_time_us_to_ticks(unsigned long us, int bpm, int ppqn)</code>	61
11.2.4.23 <code>ticks_to_delta_time_us(midipulse delta_ticks, int bpm, int ppqn)</code>	62
11.2.4.24 <code>clock_tick_duration_bogus(int bpm, int ppqn)</code>	62
11.2.4.25 <code>clock_ticks_from_ppqn(int ppqn)</code>	63
11.2.4.26 <code>double_ticks_from_ppqn(int ppqn)</code>	63
11.2.4.27 <code>measures_to_ticks(int bpm, int ppqn, int bw, int measures=1)</code>	63
11.2.4.28 <code>help_check(int argc, char *argv[])</code>	64
11.2.4.29 <code>parse_options_files(perform &p, int argc, char *argv[])</code>	64
11.2.4.30 <code>parse_command_line_options(int argc, char *argv[])</code>	65
11.2.4.31 <code>write_options_files(const perform &p)</code>	65
11.2.4.32 <code>build_details()</code>	65
11.2.4.33 <code>to_string(const event &ev)</code>	65
11.2.4.34 <code>file_access(const std::string &targetfile, int mode)</code>	66
11.2.4.35 <code>file_exists(const std::string &filename)</code>	66
11.2.4.36 <code>file_readable(const std::string &filename)</code>	66
11.2.4.37 <code>file_writable(const std::string &filename)</code>	66
11.2.4.38 <code>file_accessible(const std::string &filename)</code>	66
11.2.4.39 <code>file_executable(const std::string &filename)</code>	67
11.2.4.40 <code>file_is_directory(const std::string &filename)</code>	67
11.2.4.41 <code>make_directory(const std::string &pathname)</code>	67
11.2.4.42 <code>ppqn_is_valid(int ppqn)</code>	67
11.2.4.43 <code>jack_sync_callback(jack_transport_state_t state, jack_position_t *pos, void *arg)</code>	68
11.2.4.44 <code>jack_shutdown_callback(void *arg)</code>	68
11.2.4.45 <code>jack_timebase_callback(jack_transport_state_t state, jack_nframes_t nframes, jack_position_t *pos, int new_pos, void *arg)</code>	68
11.2.4.46 <code>jack_process_callback(jack_nframes_t nframes, void *arg)</code>	69
11.2.4.47 <code>jack_session_callback(jack_session_event_t *ev, void *arg)</code>	69
11.2.4.48 <code>keyval_name(unsigned int key)</code>	69
11.2.4.49 <code>keyval_normalize(keys_perform_transfer &k)</code>	70
11.2.4.50 <code>create_lash_driver(perform &p, int argc, char **argv)</code>	70
11.2.4.51 <code>lash_driver()</code>	70

11.2.4.52 delete_lash_driver()	70
11.2.4.53 output_thread_func(void *p)	70
11.2.4.54 input_thread_func(void *myperf)	71
11.2.4.55 min(long a, long b)	71
11.2.4.56 rc()	71
11.2.4.57 usr()	71
11.2.4.58 choose_ppqn(int ppqn)	71
11.2.4.59 make_section_name(const std::string &label, int value)	72
11.2.4.60 font_render()	72
11.2.4.61 adjustment_dummy()	72
11.2.4.62 update_mainwid_sequences()	72
11.2.4.63 update_perfedited_sequences()	73
11.2.4.64 clamp(long val, long low, long hi)	73
11.2.4.65 clamp(long val, long low, long hi)	73
11.2.5 Variable Documentation	73
11.2.5.1 c_controller_names	73
11.2.5.2 EVENT_STATUS_BIT	73
11.2.5.3 EVENT_ANY	73
11.2.5.4 EVENT_NOTE_OFF	73
11.2.5.5 EVENT_NOTE_ON	73
11.2.5.6 EVENT_AFTERTOUCH	73
11.2.5.7 EVENT_CONTROL_CHANGE	73
11.2.5.8 EVENT_PROGRAM_CHANGE	73
11.2.5.9 EVENT_CHANNEL_PRESSURE	73
11.2.5.10 EVENT_PITCH_WHEEL	73
11.2.5.11 EVENT_MIDI_SYSEX	73
11.2.5.12 EVENT_MIDI_QUARTER_FRAME	74
11.2.5.13 EVENT_MIDI_SONG_POS	74
11.2.5.14 EVENT_MIDI_SONG_SELECT	74
11.2.5.15 EVENT_MIDI_SONG_F4	74

11.2.5.16 EVENT_MIDI_SONG_F5	74
11.2.5.17 EVENT_MIDI_TUNE_SELECT	74
11.2.5.18 EVENT_MIDI_SYSEX_END	74
11.2.5.19 EVENT_MIDI_CLOCK	74
11.2.5.20 EVENT_MIDI_SONG_F9	74
11.2.5.21 EVENT_MIDI_START	74
11.2.5.22 EVENT_MIDI_CONTINUE	74
11.2.5.23 EVENT_MIDI_STOP	74
11.2.5.24 EVENT_MIDI_SONG_FD	74
11.2.5.25 EVENT_MIDI_ACTIVE_SENS	74
11.2.5.26 EVENT_MIDI_RESET	74
11.2.5.27 EVENT_MIDI_META	74
11.2.5.28 EVENT_SYSEX	74
11.2.5.29 EVENT_SYSEX_END	74
11.2.5.30 EVENT_SYSEX_CONTINUE	74
11.2.5.31 EVENT_NULL_CHANNEL	74
11.2.5.32 EVENT_GET_CHAN_MASK	75
11.2.5.33 EVENT_CLEAR_CHAN_MASK	75
11.2.5.34 c_midibus_output_size	75
11.2.5.35 c_midibus_input_size	75
11.2.5.36 c_midibus_sysex_chunk	75
11.2.5.37 c_midibus	75
11.2.5.38 c_midich	76
11.2.5.39 c_midiclocks	76
11.2.5.40 c_triggers	76
11.2.5.41 c_notes	76
11.2.5.42 c_timesig	76
11.2.5.43 c_bpmtag	76
11.2.5.44 c_triggers_new	76
11.2.5.45 c_mutegroups	76

11.2.5.46 c_midictrl	76
11.2.5.47 c_musickey	76
11.2.5.48 c_musicscale	76
11.2.5.49 c_backsequence	76
11.2.5.50 c_midi_track_ctrl	76
11.2.5.51 c_midi_control_bpm_up	76
11.2.5.52 c_midi_control_bpm_dn	76
11.2.5.53 c_midi_control_ss_up	76
11.2.5.54 c_midi_control_ss_dn	76
11.2.5.55 c_midi_control_mod_replace	76
11.2.5.56 c_midi_control_mod_snapshot	76
11.2.5.57 c_midi_control_mod_queue	76
11.2.5.58 c_midi_control_mod_gmute	77
11.2.5.59 c_midi_control_mod_glearn	77
11.2.5.60 c_midi_control_play_ss	77
11.2.5.61 c_midi_controls	77
11.2.5.62 c_scales_policy	77
11.2.5.63 c_scales_transpose_up	77
11.2.5.64 c_scales_transpose_dn	78
11.2.5.65 c_scales_text	78
11.2.5.66 c_key_text	78
11.2.5.67 c_interval_text	78
11.2.5.68 c_chord_text	78
11.2.5.69 c_max_instruments	78
11.2.5.70 c_max_busses	79
11.2.5.71 versiontext	79
11.2.5.72 long_options	79
11.2.5.73 s_arg_list	79
11.2.5.74 s_help_1a	79
11.2.5.75 s_help_1b	79

11.2.5.76 s_help_2	79
11.2.5.77 s_help_3	79
11.2.5.78 s_help_4	79
11.2.5.79 s_build_highlight_empty	79
11.2.5.80 s_build_lash_support	79
11.2.5.81 s_build_jack_support	79
11.2.5.82 s_build_jack_session	79
11.2.5.83 s_build_pause_support	79
11.2.5.84 s_build_use_event_map	79
11.2.5.85 s_build_chord_generator	80
11.2.5.86 s_build_edit_highlight	80
11.2.5.87 s_build_timesig_tempo	80
11.2.5.88 s_build_midi_vector	80
11.2.5.89 s_build_solid_grid	80
11.2.5.90 s_build_follow_progress	80
11.2.5.91 s_character_mapping	80
11.2.5.92 s_global_lash_driver	80
11.2.5.93 c_status_replace	80
11.2.5.94 c_status_snapshot	80
11.2.5.95 c_status_queue	80
11.2.5.96 g_rc_settings	80
11.2.5.97 g_user_settings	80
11.2.5.98 s_handlesize	80
11.2.5.99 s_jitter_amount	80
11.2.5.100gs_mainwid_pointer	80
11.2.5.101c_mainwid_x	81
11.2.5.102c_mainwid_y	81
11.2.5.103gs_perfedited_pointer_0	81
11.2.5.104gs_perfedited_pointer_1	81
11.2.5.105c_select_all_notes	81
11.2.5.106c_select_all_events	81
11.2.5.107c_select_inverse_notes	81
11.2.5.108c_select_inverse_events	81
11.2.5.109c_quantize_notes	81
11.2.5.110c_quantize_events	81
11.2.5.111c_tighten_events	81
11.2.5.112c_tighten_notes	81
11.2.5.113c_transpose_notes	81
11.2.5.114c_reserved	81
11.2.5.115c_transpose_h	81
11.2.5.116c_swing_notes	81
11.2.5.117s_handlesize	81

12 Data Structure Documentation	83
12.1 seq64::AbstractPerfInput Class Reference	83
12.1.1 Constructor & Destructor Documentation	84
12.1.1.1 AbstractPerfInput()	84
12.1.1.2 ~AbstractPerfInput()	84
12.1.2 Member Function Documentation	84
12.1.2.1 on_button_press_event(GdkEventButton *a_ev, perfroll &roll)=0	84
12.1.2.2 on_button_release_event(GdkEventButton *a_ev, perfroll &roll)=0	84
12.1.2.3 on_motion_notify_event(GdkEventMotion *a_ev, perfroll &roll)=0	84
12.1.3 Field Documentation	84
12.1.3.1 m_adding_pressed	84
12.2 seq64::automutex Class Reference	84
12.2.1 Detailed Description	85
12.2.2 Constructor & Destructor Documentation	85
12.2.2.1 automutex()	85
12.2.2.2 automutex(const automutex &)	85
12.2.2.3 automutex(mutex &my_mutex)	85
12.2.2.4 ~automutex()	85
12.2.3 Member Function Documentation	85
12.2.3.1 operator=(const automutex &)	85
12.2.4 Field Documentation	85
12.2.4.1 m_safety_mutex	85
12.3 seq64::click Class Reference	86
12.3.1 Detailed Description	87
12.3.2 Constructor & Destructor Documentation	87
12.3.2.1 click()	87
12.3.2.2 click(int x, int y, int button=SEQ64_CLICK_BUTTON_LEFT, bool press=true, seq_modifier_t modkey=SEQ64_NO_MASK)	87
12.3.2.3 click(const click &rhs)	87
12.3.3 Member Function Documentation	87
12.3.3.1 operator=(const click &rhs)	87

12.3.3.2	is_press() const	88
12.3.3.3	is_left() const	88
12.3.3.4	is_middle() const	88
12.3.3.5	is_right() const	88
12.3.3.6	x() const	88
12.3.3.7	y() const	88
12.3.3.8	button() const	88
12.3.3.9	modifier() const	88
12.3.3.10	mod_control() const	88
12.3.3.11	mod_control_shift() const	88
12.3.3.12	mod_super() const	88
12.3.4	Field Documentation	88
12.3.4.1	m_is_press	88
12.3.4.2	m_x	88
12.3.4.3	m_y	88
12.3.4.4	m_button	88
12.3.4.5	m_modifier	89
12.4	seq64::condition_var Class Reference	89
12.4.1	Detailed Description	90
12.4.2	Constructor & Destructor Documentation	90
12.4.2.1	condition_var()	90
12.4.3	Member Function Documentation	90
12.4.3.1	wait()	90
12.4.3.2	signal()	90
12.4.4	Field Documentation	90
12.4.4.1	sm_cond	90
12.4.4.2	m_cond	90
12.5	seq64::configfile Class Reference	90
12.5.1	Constructor & Destructor Documentation	92
12.5.1.1	configfile(const std::string &name)	92

12.5.1.2	<code>~configfile()</code>	92
12.5.2	Member Function Documentation	92
12.5.2.1	<code>next_data_line(std::ifstream &file)</code>	92
12.5.2.2	<code>line_after(std::ifstream &file, const std::string &tag)</code>	92
12.5.2.3	<code>parse(perform &perf)=0</code>	93
12.5.2.4	<code>write(const perform &perf)=0</code>	93
12.5.3	Field Documentation	93
12.5.3.1	<code>m_name</code>	93
12.5.3.2	<code>m_d</code>	93
12.5.3.3	<code>m_line</code>	93
12.6	<code>seq64::editable_event</code> Class Reference	93
12.6.1	Detailed Description	97
12.6.2	Member Enumeration Documentation	97
12.6.2.1	<code>category_t</code>	97
12.6.2.2	<code>timestamp_format_t</code>	98
12.6.3	Constructor & Destructor Documentation	98
12.6.3.1	<code>editable_event()</code>	98
12.6.3.2	<code>editable_event(const editable_events &parent)</code>	98
12.6.3.3	<code>editable_event(const editable_events &parent, const event &ev)</code>	98
12.6.3.4	<code>editable_event(const editable_event &rhs)</code>	98
12.6.3.5	<code>~editable_event()</code>	99
12.6.4	Member Function Documentation	99
12.6.4.1	<code>value_to_name(midibyte value, category_t cat)</code>	99
12.6.4.2	<code>name_to_value(const std::string &name, category_t cat)</code>	99
12.6.4.3	<code>operator=(const editable_event &rhs)</code>	99
12.6.4.4	<code>parent() const</code>	99
12.6.4.5	<code>category() const</code>	99
12.6.4.6	<code>category(category_t c)</code>	99
12.6.4.7	<code>category_string() const</code>	100
12.6.4.8	<code>category(const std::string &cs)</code>	100

12.6.4.9	<code>timestamp_string() const</code>	100
12.6.4.10	<code>timestamp() const</code>	100
12.6.4.11	<code>timestamp(midipulse ts)</code>	100
12.6.4.12	<code>timestamp(const std::string &ts_string)</code>	100
12.6.4.13	<code>time_as_pulses()</code>	100
12.6.4.14	<code>time_as_measures()</code>	100
12.6.4.15	<code>time_as_minutes()</code>	100
12.6.4.16	<code>set_status_from_string(const std::string &ts, const std::string &s, const std::string &sd0, const std::string &sd1)</code>	101
12.6.4.17	<code>format_timestamp()</code>	101
12.6.4.18	<code>stock_event_string()</code>	101
12.6.4.19	<code>status_string() const</code>	101
12.6.4.20	<code>meta_string() const</code>	101
12.6.4.21	<code>seqspec_string() const</code>	101
12.6.4.22	<code>channel_string() const</code>	101
12.6.4.23	<code>data_string() const</code>	101
12.6.4.24	<code>analyze()</code>	101
12.6.5	Field Documentation	102
12.6.5.1	<code>sm_category_names</code>	102
12.6.5.2	<code>sm_channel_event_names</code>	102
12.6.5.3	<code>sm_system_event_names</code>	102
12.6.5.4	<code>sm_meta_event_names</code>	102
12.6.5.5	<code>sm_prop_event_names</code>	102
12.6.5.6	<code>sm_category_arrays</code>	103
12.6.5.7	<code>m_parent</code>	103
12.6.5.8	<code>m_category</code>	103
12.6.5.9	<code>m_name_category</code>	103
12.6.5.10	<code>m_format_timestamp</code>	103
12.6.5.11	<code>m_name_timestamp</code>	103
12.6.5.12	<code>m_name_status</code>	103
12.6.5.13	<code>m_name_meta</code>	103

12.6.5.14	<code>m_name_seqspect</code>	103
12.6.5.15	<code>m_name_channel</code>	103
12.6.5.16	<code>m_name_data</code>	103
12.7	<code>seq64::editable_events</code> Class Reference	103
12.7.1	Member Typedef Documentation	105
12.7.1.1	<code>Key</code>	105
12.7.1.2	<code>EventsPair</code>	105
12.7.1.3	<code>Events</code>	105
12.7.1.4	<code>iterator</code>	105
12.7.1.5	<code>const_iterator</code>	105
12.7.2	Constructor & Destructor Documentation	105
12.7.2.1	<code>editable_events()</code>	105
12.7.2.2	<code>editable_events(sequence &seq, int bpm)</code>	105
12.7.2.3	<code>editable_events(const editable_events &rhs)</code>	106
12.7.2.4	<code>~editable_events()</code>	106
12.7.3	Member Function Documentation	106
12.7.3.1	<code>operator=(const editable_events &rhs)</code>	106
12.7.3.2	<code>timing() const</code>	106
12.7.3.3	<code>string_to_pulses(const std::string &ts_string) const</code>	106
12.7.3.4	<code>load_events()</code>	106
12.7.3.5	<code>save_events()</code>	107
12.7.3.6	<code>events()</code>	107
12.7.3.7	<code>begin()</code>	107
12.7.3.8	<code>begin() const</code>	107
12.7.3.9	<code>end()</code>	107
12.7.3.10	<code>end() const</code>	107
12.7.3.11	<code>count() const</code>	107
12.7.3.12	<code>add(const event &e)</code>	107
12.7.3.13	<code>add(const editable_event &e)</code>	107
12.7.3.14	<code>replace(iterator ie, const editable_event &e)</code>	108

12.7.3.15	<code>remove(iterator ie)</code>	108
12.7.3.16	<code>clear()</code>	108
12.7.3.17	<code>current_event() const</code>	108
12.7.3.18	<code>current_event(iterator cei)</code>	108
12.7.4	Friends And Related Function Documentation	108
12.7.4.1	<code>eventslots</code>	108
12.7.5	Field Documentation	108
12.7.5.1	<code>m_events</code>	108
12.7.5.2	<code>m_current_event</code>	108
12.7.5.3	<code>m_sequence</code>	108
12.7.5.4	<code>m_midi_parameters</code>	109
12.8	<code>seq64::event</code> Class Reference	109
12.8.1	Detailed Description	112
12.8.2	Constructor & Destructor Documentation	113
12.8.2.1	<code>event()</code>	113
12.8.2.2	<code>event(const event &rhs)</code>	113
12.8.2.3	<code>~event()</code>	113
12.8.3	Member Function Documentation	113
12.8.3.1	<code>operator=(const event &rhs)</code>	113
12.8.3.2	<code>operator<(const event &rhsevent) const</code>	114
12.8.3.3	<code>set_timestamp(midipulse time)</code>	114
12.8.3.4	<code>get_timestamp() const</code>	114
12.8.3.5	<code>get_channel() const</code>	114
12.8.3.6	<code>check_channel(int channel) const</code>	114
12.8.3.7	<code>is_channel_msg(midibyte m)</code>	115
12.8.3.8	<code>is_one_byte_msg(midibyte m)</code>	115
12.8.3.9	<code>is_two_byte_msg(midibyte m)</code>	115
12.8.3.10	<code>is_note_msg(midibyte m)</code>	115
12.8.3.11	<code>is_desired_cc_or_not_cc(midibyte m, midibyte cc, midibyte datum)</code>	116
12.8.3.12	<code>mod_timestamp(midipulse a_mod)</code>	116

12.8.3.13 set_status(midibyte status)	116
12.8.3.14 set_status(midibyte eventcode, midibyte channel)	117
12.8.3.15 set_channel(midibyte channel)	117
12.8.3.16 get_status() const	117
12.8.3.17 set_data(midibyte d1)	117
12.8.3.18 set_data(midibyte d1, midibyte d2)	117
12.8.3.19 get_data(midibyte &d0, midibyte &d1) const	118
12.8.3.20 increment_data1()	118
12.8.3.21 decrement_data1()	118
12.8.3.22 increment_data2()	118
12.8.3.23 decrement_data2()	118
12.8.3.24 restart_sysex()	118
12.8.3.25 append_sysex(midibyte *data, int len)	118
12.8.3.26 get_sysex() const	118
12.8.3.27 set_sysex_size(int len)	118
12.8.3.28 get_sysex_size() const	118
12.8.3.29 link(event *ev)	118
12.8.3.30 get_linked() const	119
12.8.3.31 is_linked() const	119
12.8.3.32 clear_link()	119
12.8.3.33 paint()	119
12.8.3.34 unpaint()	119
12.8.3.35 is_painted() const	119
12.8.3.36 mark()	119
12.8.3.37 unmark()	119
12.8.3.38 is_marked() const	119
12.8.3.39 select()	119
12.8.3.40 unselect()	119
12.8.3.41 is_selected() const	119
12.8.3.42 make_clock()	119

12.8.3.43	<code>data(int index) const</code>	119
12.8.3.44	<code>get_note() const</code>	119
12.8.3.45	<code>set_note(midibyte note)</code>	119
12.8.3.46	<code>get_note_velocity() const</code>	119
12.8.3.47	<code>set_note_velocity(int a_vel)</code>	119
12.8.3.48	<code>is_note_on() const</code>	120
12.8.3.49	<code>is_note_off() const</code>	120
12.8.3.50	<code>is_note() const</code>	120
12.8.3.51	<code>print() const</code>	120
12.8.3.52	<code>get_rank() const</code>	120
12.8.4	Field Documentation	120
12.8.4.1	<code>m_timestamp</code>	120
12.8.4.2	<code>m_status</code>	120
12.8.4.3	<code>m_channel</code>	121
12.8.4.4	<code>m_data</code>	121
12.8.4.5	<code>m_sysex</code>	121
12.8.4.6	<code>m_sysex_size</code>	121
12.8.4.7	<code>m_linked</code>	121
12.8.4.8	<code>m_has_link</code>	121
12.8.4.9	<code>m_selected</code>	121
12.8.4.10	<code>m_marked</code>	121
12.8.4.11	<code>m_painted</code>	121
12.9	<code>seq64::event_list::event_key</code> Class Reference	121
12.9.1	Detailed Description	122
12.9.2	Constructor & Destructor Documentation	122
12.9.2.1	<code>event_key(midipulse tstamp, int rank)</code>	122
12.9.2.2	<code>event_key(const event &e)</code>	123
12.9.3	Member Function Documentation	123
12.9.3.1	<code>operator<(const event_key &rhs) const</code>	123
12.9.4	Field Documentation	123

12.9.4.1	<code>m_timestamp</code>	123
12.9.4.2	<code>m_rank</code>	123
12.10	<code>seq64::event_list</code> Class Reference	123
12.10.1	Detailed Description	126
12.10.2	Member Typedef Documentation	126
12.10.2.1	Events	126
12.10.2.2	EventsPair	126
12.10.2.3	iterator	126
12.10.2.4	const_iterator	126
12.10.3	Constructor & Destructor Documentation	126
12.10.3.1	<code>event_list()</code>	126
12.10.3.2	<code>event_list(const event_list &a_rhs)</code>	126
12.10.3.3	<code>~event_list()</code>	126
12.10.4	Member Function Documentation	126
12.10.4.1	<code>operator=(const event_list &a_rhs)</code>	126
12.10.4.2	<code>begin()</code>	126
12.10.4.3	<code>begin() const</code>	126
12.10.4.4	<code>end()</code>	126
12.10.4.5	<code>end() const</code>	127
12.10.4.6	<code>count() const</code>	127
12.10.4.7	<code>empty() const</code>	127
12.10.4.8	<code>add(const event &e, bool postsort=true)</code>	127
12.10.4.9	<code>is_modified() const</code>	127
12.10.4.10	<code>unmodify()</code>	127
12.10.4.11	<code>remove(iterator ie)</code>	127
12.10.4.12	<code>clear()</code>	128
12.10.4.13	<code>merge(event_list &el, bool presort=true)</code>	128
12.10.4.14	<code>sort()</code>	128
12.10.4.15	<code>dref(iterator ie)</code>	128
12.10.4.16	<code>dref(const_iterator ie)</code>	128

12.10.4.17	<code>link_new()</code>	129
12.10.4.18	<code>clear_links()</code>	129
12.10.4.19	<code>verify_and_link(midipulse slength)</code>	129
12.10.4.20	<code>mark_selected()</code>	129
12.10.4.21	<code>mark_out_of_range(midipulse slength)</code>	129
12.10.4.22	<code>mark_all()</code>	129
12.10.4.23	<code>unmark_all()</code>	129
12.10.4.24	<code>remove_marked()</code>	129
12.10.4.25	<code>unpaint_all()</code>	130
12.10.4.26	<code>count_selected_notes() const</code>	130
12.10.4.27	<code>any_selected_notes() const</code>	130
12.10.4.28	<code>count_selected_events(midibyte status, midibyte cc) const</code>	130
12.10.4.29	<code>select_all()</code>	130
12.10.4.30	<code>unselect_all()</code>	130
12.10.4.31	<code>print() const</code>	130
12.10.4.32	<code>events() const</code>	130
12.10.5	Friends And Related Function Documentation	130
12.10.5.1	<code>editable_events</code>	130
12.10.5.2	<code>midi_container</code>	130
12.10.5.3	<code>midi_splitter</code>	130
12.10.5.4	<code>sequence</code>	130
12.10.6	Field Documentation	130
12.10.6.1	<code>m_events</code>	130
12.10.6.2	<code>m_is_modified</code>	130
12.11	<code>seq64::eventedit</code> Class Reference	131
12.11.1	Constructor & Destructor Documentation	134
12.11.1.1	<code>eventedit(perform &p, sequence &seq)</code>	134
12.11.1.2	<code>~eventedit()</code>	135
12.11.2	Member Function Documentation	136
12.11.2.1	<code>enqueue_draw()</code>	136

12.11.2.2 set_seq_title(const std::string &title)	136
12.11.2.3 set_seq_time_sig(const std::string &sig)	136
12.11.2.4 set_seq_ppqn(const std::string &p)	136
12.11.2.5 set_seq_count()	136
12.11.2.6 set_event_category(const std::string &c)	136
12.11.2.7 set_event_timestamp(const std::string &ts)	136
12.11.2.8 set_event_name(const std::string &n)	136
12.11.2.9 set_event_data_0(const std::string &d)	136
12.11.2.10 set_event_data_1(const std::string &d)	137
12.11.2.11 perf_modify()	137
12.11.2.12 set_dirty(bool flag=true)	137
12.11.2.13 v_adjustment(int value)	137
12.11.2.14 w_adjustment(int value, int lower, int upper)	137
12.11.2.15 change_focus(bool set_it=true)	138
12.11.2.16 close_out()	138
12.11.2.17 handle_close()	138
12.11.2.18 handle_delete()	138
12.11.2.19 handle_insert()	138
12.11.2.20 handle_modify()	138
12.11.2.21 handle_save()	138
12.11.2.22 handle_cancel()	138
12.11.2.23 on_realize()	138
12.11.2.24 on_set_focus(Widget *focus)	138
12.11.2.25 on_focus_in_event(GdkEventFocus *)	139
12.11.2.26 on_focus_out_event(GdkEventFocus *)	139
12.11.2.27 on_key_press_event(GdkEventKey *ev)	139
12.11.2.28 on_delete_event(GdkEventAny *event)	139
12.11.3 Friends And Related Function Documentation	140
12.11.3.1 eventslots	140
12.11.4 Field Documentation	140

12.11.4.1 m_table	140
12.11.4.2 m_vadjust	140
12.11.4.3 m_vscroll	140
12.11.4.4 m_eventslots	140
12.11.4.5 m_htopbox	140
12.11.4.6 m_showbox	140
12.11.4.7 m_editbox	140
12.11.4.8 m_optsbox	140
12.11.4.9 m_bottbox	140
12.11.4.10m_rightbox	140
12.11.4.11m_button_del	140
12.11.4.12m_button_ins	140
12.11.4.13m_button_modify	140
12.11.4.14m_button_save	140
12.11.4.15m_button_cancel	140
12.11.4.16m_label_seq_name	140
12.11.4.17m_label_time_sig	141
12.11.4.18m_label_ppqn	141
12.11.4.19m_label_channel	141
12.11.4.20m_label_ev_count	141
12.11.4.21m_label_spacer	141
12.11.4.22m_label_modified	141
12.11.4.23m_label_category	141
12.11.4.24m_entry_ev_timestamp	141
12.11.4.25m_entry_ev_name	141
12.11.4.26m_entry_ev_data_0	141
12.11.4.27m_entry_ev_data_1	141
12.11.4.28m_label_time_fmt	141
12.11.4.29m_label_right	141
12.11.4.30m_seq	141

12.11.4.31m_have_focus	141
12.12seq64::eventslots Class Reference	142
12.12.1 Constructor & Destructor Documentation	146
12.12.1.1 eventslots(perform &p, eventedit &parent, sequence &seq, Gtk::Adjustment &vad-just)	146
12.12.1.2 ~eventslots()	146
12.12.2 Member Function Documentation	146
12.12.2.1 event_count() const	146
12.12.2.2 line_count() const	146
12.12.2.3 line_maximum() const	146
12.12.2.4 line_increment() const	146
12.12.2.5 top_index() const	146
12.12.2.6 current_index() const	146
12.12.2.7 pager_index() const	146
12.12.2.8 load_events()	146
12.12.2.9 set_current_event(const editable_events::iterator ei, int index, bool full↵redraw=true)	146
12.12.2.10insert_event(const editable_event &edev)	147
12.12.2.11insert_event(const std::string &evtimestamp, const std::string &evname, const std::string &evdata0, const std::string &evdata1)	147
12.12.2.12delete_current_event()	148
12.12.2.13modify_current_event(const std::string &evtimestamp, const std::string &evname, const std::string &evdata0, const std::string &evdata1)	148
12.12.2.14save_events()	149
12.12.2.15select_event(int event_index=SEQ64_NULL_EVENT_INDEX, bool full↵redraw=true)	149
12.12.2.16set_text(const std::string &evcategory, const std::string &evtimestamp, const std::string &evname, const std::string &evdata0, const std::string &evdata1)	149
12.12.2.17enqueue_draw()	150
12.12.2.18convert_y(int y)	150
12.12.2.19draw_event(editable_events::iterator ei, int index)	150
12.12.2.20draw_events()	151
12.12.2.21change_vert()	151

12.12.2.22	<code>page_movement(int new_value)</code>	151
12.12.2.23	<code>page_topper(editable_events::iterator newcurrent)</code>	151
12.12.2.24	<code>decrement_top()</code>	151
12.12.2.25	<code>increment_top()</code>	152
12.12.2.26	<code>decrement_current()</code>	152
12.12.2.27	<code>increment_current()</code>	152
12.12.2.28	<code>decrement_bottom()</code>	152
12.12.2.29	<code>increment_bottom()</code>	152
12.12.2.30	<code>on_realize()</code>	152
12.12.2.31	<code>on_expose_event(GdkEventExpose *ev)</code>	153
12.12.2.32	<code>on_button_press_event(GdkEventButton *ev)</code>	153
12.12.2.33	<code>on_button_release_event(GdkEventButton *ev)</code>	153
12.12.2.34	<code>on_focus_in_event(GdkEventFocus *ev)</code>	153
12.12.2.35	<code>on_focus_out_event(GdkEventFocus *ev)</code>	153
12.12.2.36	<code>on_scroll_event(GdkEventScroll *ev)</code>	153
12.12.2.37	<code>on_size_allocate(Gtk::Allocation &)</code>	153
12.12.2.38	<code>on_move_up()</code>	153
12.12.2.39	<code>on_move_down()</code>	153
12.12.2.40	<code>on_frame_up()</code>	153
12.12.2.41	<code>on_frame_down()</code>	153
12.12.2.42	<code>on_frame_home()</code>	153
12.12.2.43	<code>on_frame_end()</code>	153
12.12.3	Friends And Related Function Documentation	153
12.12.3.1	<code>eventedit</code>	153
12.12.4	Field Documentation	153
12.12.4.1	<code>m_parent</code>	153
12.12.4.2	<code>m_seq</code>	153
12.12.4.3	<code>m_event_container</code>	153
12.12.4.4	<code>m_slots_chars</code>	153
12.12.4.5	<code>m_char_w</code>	154

12.12.4.6 m_setbox_w	154
12.12.4.7 m_slots_x	154
12.12.4.8 m_slots_y	154
12.12.4.9 m_event_count	154
12.12.4.10 m_line_count	154
12.12.4.11 m_line_maximum	154
12.12.4.12 m_line_overlap	154
12.12.4.13 m_top_index	154
12.12.4.14 m_current_index	154
12.12.4.15 m_top_iterator	154
12.12.4.16 m_bottom_iterator	154
12.12.4.17 m_current_iterator	154
12.12.4.18 m_pager_index	154
12.13 seq64::font Class Reference	154
12.13.1 Member Enumeration Documentation	156
12.13.1.1 Color	156
12.13.2 Constructor & Destructor Documentation	156
12.13.2.1 font()	156
12.13.3 Member Function Documentation	156
12.13.3.1 init(Glib::RefPtr< Gdk::Window > windo)	156
12.13.3.2 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) const	156
12.13.3.3 char_width() const	157
12.13.3.4 char_height() const	157
12.13.3.5 padded_height() const	157
12.13.4 Field Documentation	157
12.13.4.1 m_use_new_font	157
12.13.4.2 m_cell_w	157
12.13.4.3 m_cell_h	157
12.13.4.4 m_font_w	157
12.13.4.5 m_font_h	157

12.13.4.6 m_offset	157
12.13.4.7 m_padded_h	157
12.13.4.8 m_pixmap	157
12.13.4.9 m_black_pixmap	157
12.13.4.10m_white_pixmap	158
12.13.4.11m_b_on_y_pixmap	158
12.13.4.12m_y_on_b_pixmap	158
12.13.4.13m_b_on_c_pixmap	158
12.13.4.14m_c_on_b_pixmap	158
12.13.4.15m_clip_mask	158
12.14seq64::FruityPerfInput Class Reference	158
12.14.1 Constructor & Destructor Documentation	160
12.14.1.1 FruityPerfInput()	160
12.14.2 Member Function Documentation	160
12.14.2.1 on_button_press_event(GdkEventButton *ev, perfroll &roll)	160
12.14.2.2 on_button_release_event(GdkEventButton *ev, perfroll &roll)	160
12.14.2.3 on_motion_notify_event(GdkEventMotion *ev, perfroll &roll)	161
12.14.2.4 update_mouse_pointer(perfroll &roll)	161
12.14.2.5 on_left_button_pressed(GdkEventButton *ev, perfroll &roll)	161
12.14.2.6 on_right_button_pressed(GdkEventButton *ev, perfroll &roll)	161
12.14.3 Friends And Related Function Documentation	162
12.14.3.1 perfroll	162
12.14.4 Field Documentation	162
12.14.4.1 m_current_x	162
12.14.4.2 m_current_y	162
12.15seq64::FruitySeqEventInput Struct Reference	162
12.15.1 Constructor & Destructor Documentation	163
12.15.1.1 FruitySeqEventInput()	163
12.15.2 Member Function Documentation	163
12.15.2.1 update_mouse_pointer(seqevent &ths)	163

12.15.2.2 on_button_press_event(GdkEventButton *ev, sequevent &ths)	164
12.15.2.3 on_button_release_event(GdkEventButton *ev, sequevent &ths)	164
12.15.2.4 on_motion_notify_event(GdkEventMotion *ev, sequevent &ths)	165
12.15.3 Field Documentation	165
12.15.3.1 m_justselected_one	165
12.15.3.2 m_is_drag_pasting_start	165
12.15.3.3 m_is_drag_pasting	165
12.16seq64::FruitySeqRollInput Class Reference	165
12.16.1 Constructor & Destructor Documentation	165
12.16.1.1 FruitySeqRollInput()	165
12.16.2 Member Function Documentation	165
12.16.2.1 update_mouse_pointer(seqroll &ths)	165
12.16.2.2 on_button_press_event(GdkEventButton *ev, seqroll &ths)	166
12.16.2.3 on_button_release_event(GdkEventButton *ev, seqroll &ths)	166
12.16.2.4 on_motion_notify_event(GdkEventMotion *ev, seqroll &ths)	166
12.16.3 Field Documentation	167
12.16.3.1 m_erase_painting	167
12.16.3.2 m_drag_paste_start_pos	167
12.17seq64::gui_assistant Class Reference	167
12.17.1 Detailed Description	168
12.17.2 Constructor & Destructor Documentation	168
12.17.2.1 gui_assistant(keys_perform &kp)	168
12.17.2.2 ~gui_assistant()	168
12.17.3 Member Function Documentation	168
12.17.3.1 quit()=0	168
12.17.3.2 jack_idle_connect(jack_assistant &jack)=0	168
12.17.3.3 lash_timeout_connect(lash *lashobject)=0	168
12.17.3.4 keys() const	168
12.17.3.5 keys()	168
12.17.4 Field Documentation	168

12.17.4.1 m_keys_perform	168
12.18seq64::gui_assistant_gtk2 Class Reference	169
12.18.1 Constructor & Destructor Documentation	170
12.18.1.1 gui_assistant_gtk2()	170
12.18.1.2 ~gui_assistant_gtk2()	170
12.18.2 Member Function Documentation	170
12.18.2.1 quit()	170
12.18.2.2 lash_timeout_connect(lash *lashobject)	170
12.18.2.3 jack_idle_connect(jack_assistant &jack)	170
12.18.3 Field Documentation	170
12.18.3.1 sm_internal_keys	170
12.19seq64::gui_drawingarea_gtk2 Class Reference	170
12.19.1 Detailed Description	174
12.19.2 Constructor & Destructor Documentation	174
12.19.2.1 gui_drawingarea_gtk2(const gui_drawingarea_gtk2 &)	174
12.19.2.2 gui_drawingarea_gtk2(perform &p, int window_x=0, int window_y=0)	174
12.19.2.3 gui_drawingarea_gtk2(perform &a_perf, Gtk::Adjustment &a_hadjust, Gtk::Adjustment &a_vadjust, int window_x=0, int window_y=0)	174
12.19.2.4 ~gui_drawingarea_gtk2()	174
12.19.3 Member Function Documentation	174
12.19.3.1 operator=(const gui_drawingarea_gtk2 &)	174
12.19.3.2 window_x() const	174
12.19.3.3 window_y() const	174
12.19.3.4 current_x() const	174
12.19.3.5 current_y() const	174
12.19.3.6 drop_x() const	174
12.19.3.7 drop_y() const	174
12.19.3.8 force_draw()	174
12.19.3.9 perf()	175
12.19.3.10clear_window()	175
12.19.3.11set_line(Gtk::LineStyle ls, int width=1)	175

12.19.3.12	<code>draw_line(int x1, int y1, int x2, int y2)</code>	175
12.19.3.13	<code>draw_line(const Color &c, int x1, int y1, int x2, int y2)</code>	175
12.19.3.14	<code>draw_line_on_pixmap(int x1, int y1, int x2, int y2)</code>	175
12.19.3.15	<code>draw_line_on_pixmap(const Color &c, int x1, int y1, int x2, int y2)</code>	176
12.19.3.16	<code>draw_line(Glib::RefPtr< Gdk::Pixmap > &pixmap, int x1, int y1, int x2, int y2)</code>	176
12.19.3.17	<code>draw_line(Glib::RefPtr< Gdk::Pixmap > &pixmap, const Color &c, int x1, int y1, int x2, int y2)</code>	176
12.19.3.18	<code>draw_line(Glib::RefPtr< Gdk::Drawable > &drawable, int x1, int y1, int x2, int y2)</code>	176
12.19.3.19	<code>draw_line(Glib::RefPtr< Gdk::Drawable > &drawable, const Color &c, int x1, int y1, int x2, int y2)</code>	176
12.19.3.20	<code>render_string(int x, int y, const std::string &s, font::Color color)</code>	177
12.19.3.21	<code>render_string_on_pixmap(int x, int y, const std::string &s, font::Color color)</code>	177
12.19.3.22	<code>draw_rectangle(int x, int y, int lx, int ly, bool fill=true)</code>	177
12.19.3.23	<code>draw_rectangle(const Color &c, int x, int y, int lx, int ly, bool fill=true)</code>	177
12.19.3.24	<code>draw_rectangle(Glib::RefPtr< Gdk::Drawable > &drawable, int x, int y, int lx, int ly, bool fill=true)</code>	178
12.19.3.25	<code>draw_rectangle(Glib::RefPtr< Gdk::Drawable > &drawable, const Color &c, int x, int y, int lx, int ly, bool fill=true)</code>	178
12.19.3.26	<code>draw_rectangle(Glib::RefPtr< Gdk::Pixmap > &pixmap, int x, int y, int lx, int ly, bool fill=true)</code>	178
12.19.3.27	<code>draw_rectangle(Glib::RefPtr< Gdk::Pixmap > &pixmap, const Color &c, int x, int y, int lx, int ly, bool fill=true)</code>	179
12.19.3.28	<code>draw_rectangle_on_pixmap(int x, int y, int lx, int ly, bool fill=true)</code>	179
12.19.3.29	<code>draw_rectangle_on_pixmap(const Color &c, int x, int y, int lx, int ly, bool fill=true)</code>	179
12.19.3.30	<code>draw_normal_rectangle_on_pixmap(int x, int y, int lx, int ly, bool fill=true)</code>	180
12.19.3.31	<code>draw_drawable(int xsrc, int ysrc, int xdest, int ydest, int width, int height)</code>	180
12.19.3.32	<code>scroll_hadjust(Gtk::Adjustment &hadjust, double step)</code>	180
12.19.3.33	<code>scroll_vadjust(Gtk::Adjustment &vadjust, double step)</code>	180
12.19.3.34	<code>scroll_hset(Gtk::Adjustment &hadjust, double value)</code>	181
12.19.3.35	<code>scroll_vset(Gtk::Adjustment &vadjust, double value)</code>	181
12.19.3.36	<code>set_current_drop_x(int x)</code>	181
12.19.3.37	<code>set_current_drop_y(int y)</code>	181
12.19.3.38	<code>gtk_drawarea_init()</code>	181

12.19.3.39on_realize()	181
12.19.4 Field Documentation	181
12.19.4.1 m_gc	181
12.19.4.2 m_window	181
12.19.4.3 m_vadjust	181
12.19.4.4 m_hadjust	181
12.19.4.5 m_pixmap	181
12.19.4.6 m_background	181
12.19.4.7 m_foreground	182
12.19.4.8 m_mainperf	182
12.19.4.9 m_window_x	182
12.19.4.10m_window_y	182
12.19.4.11m_current_x	182
12.19.4.12m_current_y	182
12.19.4.13m_drop_x	182
12.19.4.14m_drop_y	182
12.20seq64::gui_palette_gtk2 Class Reference	182
12.20.1 Detailed Description	184
12.20.2 Member Typedef Documentation	184
12.20.2.1 Color	184
12.20.3 Constructor & Destructor Documentation	185
12.20.3.1 gui_palette_gtk2()	185
12.20.3.2 ~gui_palette_gtk2()	185
12.20.4 Member Function Documentation	185
12.20.4.1 line_color() const	185
12.20.4.2 progress_color() const	185
12.20.4.3 black() const	186
12.20.4.4 white() const	186
12.20.4.5 grey() const	186
12.20.4.6 dark_grey() const	186

12.20.4.7 light_grey() const	186
12.20.4.8 red() const	186
12.20.4.9 orange() const	186
12.20.4.10dark_orange() const	186
12.20.4.11yellow() const	186
12.20.4.12green() const	186
12.20.4.13blue() const	186
12.20.4.14dark_cyan() const	186
12.20.4.15bg_color() const	186
12.20.4.16bg_color(const Color &c)	186
12.20.4.17fg_color() const	186
12.20.4.18fg_color(const Color &c)	186
12.20.5 Field Documentation	186
12.20.5.1 m_black	186
12.20.5.2 m_white	186
12.20.5.3 m_grey	186
12.20.5.4 m_dk_grey	186
12.20.5.5 m_lt_grey	186
12.20.5.6 m_red	186
12.20.5.7 m_orange	187
12.20.5.8 m_dk_orange	187
12.20.5.9 m_yellow	187
12.20.5.10m_green	187
12.20.5.11m_blue	187
12.20.5.12m_dk_cyan	187
12.20.5.13m_line_color	187
12.20.5.14m_progress_color	187
12.20.5.15m_bg_color	187
12.20.5.16m_fg_color	187
12.21 seq64::gui_window_gtk2 Class Reference	187

12.21.1 Constructor & Destructor Documentation	189
12.21.1.1 gui_window_gtk2(perform &p, int window_x=0, int window_y=0)	189
12.21.1.2 ~gui_window_gtk2()	189
12.21.2 Member Function Documentation	189
12.21.2.1 perf()	190
12.21.2.2 quit()	190
12.21.2.3 redraw_period_ms() const	190
12.21.2.4 is_realized() const	190
12.21.2.5 scroll_hadjust(Gtk::Adjustment &hadjust, double step)	190
12.21.2.6 scroll_vadjust(Gtk::Adjustment &vadjust, double step)	190
12.21.2.7 scroll_hset(Gtk::Adjustment &hadjust, double value)	190
12.21.2.8 scroll_vset(Gtk::Adjustment &vadjust, double value)	190
12.21.2.9 on_realize()	190
12.21.3 Field Documentation	190
12.21.3.1 m_mainperf	191
12.21.3.2 m_window_x	191
12.21.3.3 m_window_y	191
12.21.3.4 m_redraw_period_ms	191
12.21.3.5 m_is_realized	191
12.22seq64::jack_assistant Class Reference	191
12.22.1 Constructor & Destructor Documentation	194
12.22.1.1 jack_assistant(perform &parent, int bpminute=SEQ64_DEFAULT_BPM, int ppqn=SEQ64_USE_DEFAULT_PPQN, int bpm=SEQ64_DEFAULT_BEAT↔ S_PER_MEASURE, int beatwidth=SEQ64_DEFAULT_BEAT_WIDTH)	194
12.22.1.2 ~jack_assistant()	194
12.22.2 Member Function Documentation	194
12.22.2.1 parent()	194
12.22.2.2 is_running() const	194
12.22.2.3 is_master() const	194
12.22.2.4 get_ppqn() const	194
12.22.2.5 get_beat_width() const	194

12.22.2.6 set_beat_width(int bw)	194
12.22.2.7 get_beats_per_measure() const	195
12.22.2.8 set_beats_per_measure(int bpm)	195
12.22.2.9 get_beats_per_minute() const	195
12.22.2.10 set_beats_per_minute(int bpmminute)	195
12.22.2.11 init()	195
12.22.2.12 deinit()	196
12.22.2.13 session_event()	196
12.22.2.14 start()	196
12.22.2.15 stop()	196
12.22.2.16 position(bool to_left_tick, bool relocate=false)	197
12.22.2.17 output(jack_scratchpad &pad)	197
12.22.2.18 set_ppqn(int ppqn)	198
12.22.2.19 get_jack_tick() const	198
12.22.2.20 get_jack_pos() const	198
12.22.2.21 set_jack_running(bool flag)	198
12.22.2.22 client() const	198
12.22.2.23 client_name() const	198
12.22.2.24 client_uuid() const	198
12.22.2.25 info_message(const std::string &msg)	198
12.22.2.26 error_message(const std::string &msg)	199
12.22.2.27 client_open(const std::string &clientname)	199
12.22.2.28 get_jack_client_info()	200
12.22.2.29 show_statuses(unsigned bits)	200
12.22.2.30 show_position(const jack_position_t &pos) const	200
12.22.2.31 sync(jack_transport_state_t state=(jack_transport_state_t)(-1))	201
12.22.2.32 set_position(midipulse currenttick)	202
12.22.3 Friends And Related Function Documentation	202
12.22.3.1 jack_process_callback	202
12.22.3.2 jack_shutdown_callback	202

12.22.3.3 jack_sync_callback	202
12.22.3.4 jack_timebase_callback	203
12.22.3.5 jack_session_callback	203
12.22.4 Field Documentation	204
12.22.4.1 sm_status_pairs	204
12.22.4.2 m_jack_parent	204
12.22.4.3 m_jack_client	204
12.22.4.4 m_jack_client_name	204
12.22.4.5 m_jack_client_uuid	204
12.22.4.6 m_jack_frame_current	204
12.22.4.7 m_jack_frame_last	204
12.22.4.8 m_jack_pos	204
12.22.4.9 m_jack_transport_state	204
12.22.4.10 m_jack_transport_state_last	205
12.22.4.11 m_jack_tick	205
12.22.4.12 m_jsession_ev	205
12.22.4.13 m_jack_running	205
12.22.4.14 m_jack_master	205
12.22.4.15 m_ppqn	205
12.22.4.16 m_beats_per_measure	205
12.22.4.17 m_beat_width	205
12.22.4.18 m_beats_per_minute	205
12.23seq64::jack_scratchpad Class Reference	205
12.23.1 Detailed Description	206
12.23.2 Field Documentation	206
12.23.2.1 js_current_tick	206
12.23.2.2 js_total_tick	206
12.23.2.3 js_clock_tick	206
12.23.2.4 js_jack_stopped	206
12.23.2.5 js_dumping	206

12.23.2.6 js_init_clock	206
12.23.2.7 js_looping	206
12.23.2.8 js_playback_mode	206
12.23.2.9 js_ticks_converted_last	206
12.24seq64::jack_status_pair_t Struct Reference	206
12.24.1 Field Documentation	206
12.24.1.1 jf_bit	206
12.24.1.2 jf_meaning	206
12.25seq64::keybindentry Class Reference	206
12.25.1 Member Enumeration Documentation	207
12.25.1.1 type	207
12.25.2 Constructor & Destructor Documentation	207
12.25.2.1 keybindentry(type t, unsigned int *location_to_write=NULLptr, perform *p=NULLptr, long s=0)	207
12.25.3 Member Function Documentation	208
12.25.3.1 set(unsigned int val)	208
12.25.3.2 on_key_press_event(GdkEventKey *event)	208
12.25.4 Friends And Related Function Documentation	208
12.25.4.1 options	208
12.25.5 Field Documentation	208
12.25.5.1 m_key	208
12.25.5.2 m_type	208
12.25.5.3 m_perf	208
12.25.5.4 m_slot	208
12.26seq64::keys_perform Class Reference	209
12.26.1 Detailed Description	214
12.26.2 Member Typedef Documentation	214
12.26.2.1 SlotMap	214
12.26.2.2 RevSlotMap	214
12.26.3 Constructor & Destructor Documentation	214
12.26.3.1 keys_perform()	214

12.26.3.2 ~keys_perform()	214
12.26.4 Member Function Documentation	214
12.26.4.1 set_keys(const keys_perform_transfer &kpt)	214
12.26.4.2 get_keys(keys_perform_transfer &kpt)	215
12.26.4.3 bpm_up() const	215
12.26.4.4 bpm_up(unsigned int x)	215
12.26.4.5 bpm_dn() const	215
12.26.4.6 bpm_dn(unsigned int x)	215
12.26.4.7 replace() const	215
12.26.4.8 replace(unsigned int x)	215
12.26.4.9 queue() const	215
12.26.4.10queue(unsigned int x)	215
12.26.4.11keep_queue() const	216
12.26.4.12keep_queue(unsigned int x)	216
12.26.4.13snapshot_1() const	216
12.26.4.14snapshot_1(unsigned int x)	216
12.26.4.15snapshot_2() const	216
12.26.4.16snapshot_2(unsigned int x)	216
12.26.4.17screenshot_up() const	216
12.26.4.18screenshot_up(unsigned int x)	216
12.26.4.19screenshot_dn() const	216
12.26.4.20screenshot_dn(unsigned int x)	216
12.26.4.21set_playing_screenshot() const	216
12.26.4.22set_playing_screenshot(unsigned int x)	216
12.26.4.23group_on() const	217
12.26.4.24group_on(unsigned int x)	217
12.26.4.25group_off() const	217
12.26.4.26group_off(unsigned int x)	217
12.26.4.27group_learn() const	217
12.26.4.28group_learn(unsigned int x)	217

12.26.4.29	<code>start()</code> const	217
12.26.4.30	<code>start(unsigned int x)</code>	217
12.26.4.31	<code>pause()</code> const	217
12.26.4.32	<code>pause(unsigned int x)</code>	217
12.26.4.33	<code>pattern_edit()</code> const	218
12.26.4.34	<code>pattern_edit(unsigned int x)</code>	218
12.26.4.35	<code>event_edit()</code> const	218
12.26.4.36	<code>event_edit(unsigned int x)</code>	218
12.26.4.37	<code>stop()</code> const	218
12.26.4.38	<code>stop(unsigned int x)</code>	218
12.26.4.39	<code>show_ui_sequence_key()</code> const	218
12.26.4.40	<code>show_ui_sequence_key(bool flag)</code>	218
12.26.4.41	<code>show_ui_sequence_number()</code> const	218
12.26.4.42	<code>show_ui_sequence_number(bool flag)</code>	218
12.26.4.43	<code>get_key_events()</code>	219
12.26.4.44	<code>get_key_groups()</code>	219
12.26.4.45	<code>get_key_events_rev()</code>	219
12.26.4.46	<code>get_key_groups_rev()</code>	219
12.26.4.47	<code>lookup_keyevent_key(long seqnum)</code>	219
12.26.4.48	<code>lookup_keyevent_seq(unsigned int keycode)</code>	219
12.26.4.49	<code>lookup_keygroup_key(long groupnum)</code>	219
12.26.4.50	<code>lookup_keygroup_group(unsigned int keycode)</code>	219
12.26.4.51	<code>key_name(unsigned int key)</code> const	219
12.26.4.52	<code>set_all_key_events()</code>	220
12.26.4.53	<code>set_all_key_groups()</code>	220
12.26.4.54	<code>set_key_event(unsigned int keycode, long sequence_slot)</code>	220
12.26.4.55	<code>set_key_group(unsigned int keycode, long group_slot)</code>	220
12.26.4.56	<code>at_bpm_up()</code>	220
12.26.4.57	<code>at_bpm_dn()</code>	220
12.26.4.58	<code>at_replace()</code>	221

12.26.4.59at_queue()	221
12.26.4.60at_keep_queue()	221
12.26.4.61at_snapshot_1()	221
12.26.4.62at_snapshot_2()	221
12.26.4.63at_screenset_up()	221
12.26.4.64at_screenset_dn()	221
12.26.4.65at_set_playing_screenset()	221
12.26.4.66at_group_on()	221
12.26.4.67at_group_off()	221
12.26.4.68at_group_learn()	222
12.26.4.69at_start()	222
12.26.4.70at_pause()	222
12.26.4.71at_pattern_edit()	222
12.26.4.72at_event_edit()	222
12.26.4.73at_stop()	222
12.26.4.74at_show_ui_sequence_key()	222
12.26.4.75at_show_ui_sequence_number()	222
12.26.5 Friends And Related Function Documentation	222
12.26.5.1 options	222
12.26.5.2 perform	222
12.26.5.3 optionsfile	222
12.26.6 Field Documentation	222
12.26.6.1 m_key_show_ui_sequence_key	222
12.26.6.2 m_key_show_ui_sequence_number	222
12.26.6.3 m_key_events	223
12.26.6.4 m_key_groups	223
12.26.6.5 m_key_events_rev	223
12.26.6.6 m_key_groups_rev	223
12.26.6.7 m_key_bpm_up	223
12.26.6.8 m_key_bpm_dn	223

12.26.6.9 m_key_replace	223
12.26.6.10m_key_queue	223
12.26.6.11m_key_keep_queue	223
12.26.6.12m_key_snapshot_1	223
12.26.6.13m_key_snapshot_2	223
12.26.6.14m_key_screensset_up	223
12.26.6.15m_key_screensset_dn	223
12.26.6.16m_key_set_playing_screensset	223
12.26.6.17m_key_group_on	223
12.26.6.18m_key_group_off	224
12.26.6.19m_key_group_learn	224
12.26.6.20m_key_start	224
12.26.6.21m_key_pause	224
12.26.6.22m_key_pattern_edit	224
12.26.6.23m_key_event_edit	224
12.26.6.24m_key_stop	224
12.27seq64::keys_perform_gtk2 Class Reference	224
12.27.1 Detailed Description	226
12.27.2 Constructor & Destructor Documentation	226
12.27.2.1 keys_perform_gtk2()	226
12.27.2.2 ~keys_perform_gtk2()	226
12.27.3 Member Function Documentation	226
12.27.3.1 key_name(unsigned int key) const	226
12.27.3.2 set_all_key_events()	226
12.27.3.3 set_all_key_groups()	226
12.28seq64::keys_perform_transfer Struct Reference	226
12.28.1 Field Documentation	227
12.28.1.1 kpt_bpm_up	227
12.28.1.2 kpt_bpm_dn	227
12.28.1.3 kpt_screensset_up	227

12.28.1.4 kpt_screenset_dn	227
12.28.1.5 kpt_set_playing_screenset	227
12.28.1.6 kpt_group_on	227
12.28.1.7 kpt_group_off	227
12.28.1.8 kpt_group_learn	227
12.28.1.9 kpt_replace	227
12.28.1.10 kpt_queue	227
12.28.1.11 kpt_keep_queue	227
12.28.1.12 kpt_snapshot_1	227
12.28.1.13 kpt_snapshot_2	228
12.28.1.14 kpt_start	228
12.28.1.15 kpt_stop	228
12.28.1.16 kpt_show_ui_sequence_key	228
12.28.1.17 kpt_show_ui_sequence_number	228
12.28.1.18 kpt_pattern_edit	228
12.28.1.19 kpt_event_edit	228
12.28.1.20 kpt_pause	228
12.29 seq64::keystroke Class Reference	228
12.29.1 Detailed Description	229
12.29.2 Constructor & Destructor Documentation	229
12.29.2.1 keystroke()	229
12.29.2.2 keystroke(unsigned int key, bool press=SEQ64_KEYSTROKE_PRESS, int modkey=int(SEQ64_NO_MASK))	229
12.29.2.3 keystroke(const keystroke &rhs)	229
12.29.3 Member Function Documentation	229
12.29.3.1 operator=(const keystroke &rhs)	229
12.29.3.2 is_press() const	230
12.29.3.3 is_letter(unsigned int ch=SEQ64_KEYSTROKE_BAD_VALUE) const	230
12.29.3.4 is(unsigned int ch)	230
12.29.3.5 is_delete() const	230
12.29.3.6 key() const	230

12.29.3.7	shift_lock()	230
12.29.3.8	modifier() const	231
12.29.3.9	mod_control() const	231
12.29.3.10	mod_control_shift() const	231
12.29.3.11	mod_super() const	231
12.29.4	Field Documentation	231
12.29.4.1	m_is_press	231
12.29.4.2	m_key	231
12.29.4.3	m_modifier	231
12.30	seq64::lash Class Reference	231
12.30.1	Detailed Description	232
12.30.2	Constructor & Destructor Documentation	232
12.30.2.1	lash(perform &p, int argc, char **argv)	232
12.30.3	Member Function Documentation	232
12.30.3.1	set_alsa_client_id(int id)	232
12.30.3.2	start()	232
12.30.3.3	process_events()	232
12.30.3.4	init()	233
12.30.3.5	handle_event(lash_event_t *conf)	233
12.30.3.6	handle_config(lash_config_t *conf)	233
12.30.4	Field Documentation	233
12.30.4.1	m_perform	233
12.30.4.2	m_client	233
12.30.4.3	m_lash_args	233
12.30.4.4	m_is_lash_supported	233
12.31	seq64::maintime Class Reference	233
12.31.1	Detailed Description	235
12.31.2	Constructor & Destructor Documentation	236
12.31.2.1	maintime(const maintime &)	236
12.31.2.2	maintime(perform &p, int ppqn=SEQ64_USE_DEFAULT_PPQN)	236

12.31.2.3 ~maintime()	236
12.31.3 Member Function Documentation	236
12.31.3.1 operator=(const maintime &)	236
12.31.3.2 idle_progress(midipulse ticks)	236
12.31.3.3 on_realize()	236
12.31.3.4 on_expose_event(GdkEventExpose *ev)	236
12.31.4 Friends And Related Function Documentation	236
12.31.4.1 mainwnd	236
12.31.5 Field Documentation	236
12.31.5.1 m_beat_width	236
12.31.5.2 m_bar_width	237
12.31.5.3 m_pill_width	237
12.31.5.4 m_box_width	237
12.31.5.5 m_box_height	237
12.31.5.6 m_flash_width	237
12.31.5.7 m_flash_height	237
12.31.5.8 m_flash_x	237
12.31.5.9 m_box_less_pill	237
12.31.5.10 m_tick	237
12.31.5.11 m_ppqn	237
12.32 seq64::mainwid Class Reference	238
12.32.1 Detailed Description	241
12.32.2 Constructor & Destructor Documentation	241
12.32.2.1 mainwid(perform &p)	241
12.32.2.2 ~mainwid()	241
12.32.3 Member Function Documentation	241
12.32.3.1 set_screenset(int ss, bool setperf=false)	241
12.32.3.2 reset()	242
12.32.3.3 update_sequences_on_window()	242
12.32.3.4 draw_pixmap_on_window()	242

12.32.3.5 fill_background_window()	242
12.32.3.6 redraw(int seq)	242
12.32.3.7 seq_set_and_edit(int seqnum)	242
12.32.3.8 seq_set_and_eventedit(int seqnum)	242
12.32.3.9 draw_marker_on_sequence(int seq, int tick)	242
12.32.3.10 update_markers(int ticks)	243
12.32.3.11 valid_sequence(int seq)	243
12.32.3.12 draw_sequence_on_pixmap(int seq)	243
12.32.3.13 draw_sequences_on_pixmap()	243
12.32.3.14 draw_sequence_pixmap_on_window(int seq)	244
12.32.3.15 seq_from_xy(int x, int y)	244
12.32.3.16 timeout()	244
12.32.3.17 calculate_base_sizes(int seq, int &basex, int &basey)	244
12.32.3.18 select_fg_bg_colors(int seqnum)	244
12.32.3.19 on_realize()	244
12.32.3.20 on_expose_event(GdkEventExpose *ev)	245
12.32.3.21 on_button_press_event(GdkEventButton *ev)	245
12.32.3.22 on_button_release_event(GdkEventButton *ev)	245
12.32.3.23 on_motion_notify_event(GdkEventMotion *p0)	246
12.32.3.24 on_focus_in_event(GdkEventFocus *)	246
12.32.3.25 on_focus_out_event(GdkEventFocus *)	246
12.32.4 Friends And Related Function Documentation	246
12.32.4.1 mainwnd	246
12.32.4.2 update_mainwid_sequences	246
12.32.5 Field Documentation	247
12.32.5.1 m_moving_seq	247
12.32.5.2 m_button_down	247
12.32.5.3 m_moving	247
12.32.5.4 m_old_seq	247
12.32.5.5 m_screenset	247

12.32.5.6 m_last_tick_x	247
12.32.5.7 m_mainwnd_rows	247
12.32.5.8 m_mainwnd_cols	247
12.32.5.9 m_seqarea_x	247
12.32.5.10m_seqarea_y	247
12.32.5.11m_seqarea_seq_x	247
12.32.5.12m_seqarea_seq_y	247
12.32.5.13m_mainwid_x	247
12.32.5.14m_mainwid_y	247
12.32.5.15m_mainwid_border	247
12.32.5.16m_mainwid_spacing	247
12.32.5.17m_text_size_x	247
12.32.5.18m_text_size_y	247
12.32.5.19m_max_sets	247
12.32.5.20m_screenset_slots	247
12.32.5.21m_screenset_offset	248
12.32.5.22m_progress_height	248
12.33seq64::mainwnd Class Reference	248
12.33.1 Constructor & Destructor Documentation	253
12.33.1.1 mainwnd(perform &a_p, bool allowperf2=true, int ppqn=SEQ64_USE_DEFAU↵ LT_PPQN)	253
12.33.1.2 ~mainwnd()	253
12.33.2 Member Function Documentation	253
12.33.2.1 open_file(const std::string &filename)	253
12.33.2.2 ppqn() const	254
12.33.2.3 ppqn(int ppqn)	254
12.33.2.4 handle_signal(int sig)	254
12.33.2.5 file_import_dialog()	254
12.33.2.6 options_dialog()	254
12.33.2.7 about_dialog()	254
12.33.2.8 adj_callback_ss()	254

12.33.2.9 <code>adj_callback_bpm()</code>	254
12.33.2.10 <code>edit_callback_notepad()</code>	255
12.33.2.11 <code>timer_callback()</code>	255
12.33.2.12 <code>set_image(bool isrunning)</code>	255
12.33.2.13 <code>start_playing()</code>	255
12.33.2.14 <code>pause_playing()</code>	255
12.33.2.15 <code>stop_playing()</code>	255
12.33.2.16 <code>toggle_playing()</code>	256
12.33.2.17 <code>learn_toggle()</code>	256
12.33.2.18 <code>open_performance_edit()</code>	256
12.33.2.19 <code>open_performance_edit_2()</code>	256
12.33.2.20 <code>enregister_perfedits()</code>	256
12.33.2.21 <code>sequence_key(int seq)</code>	256
12.33.2.22 <code>update_window_title()</code>	256
12.33.2.23 <code>toLower(std::string &)</code>	256
12.33.2.24 <code>file_new()</code>	256
12.33.2.25 <code>file_open()</code>	256
12.33.2.26 <code>file_save()</code>	256
12.33.2.27 <code>file_save_as()</code>	256
12.33.2.28 <code>file_exit()</code>	256
12.33.2.29 <code>new_file()</code>	256
12.33.2.30 <code>save_file()</code>	257
12.33.2.31 <code>choose_file()</code>	257
12.33.2.32 <code>query_save_changes()</code>	257
12.33.2.33 <code>s_save()</code>	257
12.33.2.34 <code>install_signal_handlers()</code>	257
12.33.2.35 <code>signal_action(Glib::IOCondition condition)</code>	257
12.33.2.36 <code>on_delete_event(GdkEventAny *a_e)</code>	257
12.33.2.37 <code>on_key_press_event(GdkEventKey *a_ev)</code>	257
12.33.2.38 <code>on_key_release_event(GdkEventKey *a_ev)</code>	257

12.33.2.39on_grouplearnchange(bool state)	258
12.33.3 Field Documentation	258
12.33.3.1 m_sigpipe	258
12.33.3.2 m_tooltips	258
12.33.3.3 m_menubar	258
12.33.3.4 m_menu_file	258
12.33.3.5 m_menu_view	258
12.33.3.6 m_menu_help	258
12.33.3.7 m_ppqn	258
12.33.3.8 m_main_wid	258
12.33.3.9 m_main_time	258
12.33.3.10m_perf_edit	258
12.33.3.11m_perf_edit_2	258
12.33.3.12m_options	259
12.33.3.13m_main_cursor	259
12.33.3.14m_image_play	259
12.33.3.15m_button_learn	259
12.33.3.16m_button_stop	259
12.33.3.17m_button_play	259
12.33.3.18m_button_perfedit	259
12.33.3.19m_adjust_bpm	259
12.33.3.20m_spinbutton_bpm	259
12.33.3.21m_adjust_ss	259
12.33.3.22m_spinbutton_ss	259
12.33.3.23m_adjust_load_offset	259
12.33.3.24m_spinbutton_load_offset	259
12.33.3.25m_entry_notes	259
12.33.3.26m_is_running	260
12.33.3.27m_timeout_connect	260
12.33.3.28m_call_seq_edit	260

12.33.3.29m_call_seq_eventedit	260
12.34seq64::mastermidibus Class Reference	260
12.34.1 Constructor & Destructor Documentation	262
12.34.1.1 mastermidibus(int ppqn=SEQ64_USE_DEFAULT_PPQN, int bpm=c_beats_per_minute)	262
12.34.1.2 ~mastermidibus()	262
12.34.2 Member Function Documentation	263
12.34.2.1 init(int ppqn)	263
12.34.2.2 get_alsa_seq() const	263
12.34.2.3 get_num_out_buses() const	263
12.34.2.4 get_num_in_buses() const	263
12.34.2.5 set_beats_per_minute(int bpm)	263
12.34.2.6 set_ppqn(int ppqn)	263
12.34.2.7 get_beats_per_minute() const	264
12.34.2.8 get_ppqn() const	264
12.34.2.9 get_midi_out_bus_name(int bus)	264
12.34.2.10get_midi_in_bus_name(int bus)	264
12.34.2.11print()	264
12.34.2.12flush()	264
12.34.2.13start()	264
12.34.2.14stop()	264
12.34.2.15clock(midipulse tick)	264
12.34.2.16continue_from(midipulse tick)	265
12.34.2.17nit_clock(midipulse tick)	265
12.34.2.18poll_for_midi()	265
12.34.2.19s_more_input()	265
12.34.2.20get_midi_event(event *in)	265
12.34.2.21set_sequence_input(bool state, sequence *seq)	266
12.34.2.22s_dumping() const	266
12.34.2.23get_sequence() const	266
12.34.2.24sysex(event *event)	266

12.34.2.25	<code>port_start(int client, int port)</code>	266
12.34.2.26	<code>port_exit(int client, int port)</code>	266
12.34.2.27	<code>play(bussbyte bus, event *e24, midibyte channel)</code>	267
12.34.2.28	<code>set_clock(bussbyte bus, clock_e clock_type)</code>	267
12.34.2.29	<code>get_clock(bussbyte bus)</code>	267
12.34.2.30	<code>set_input(bussbyte bus, bool inputing)</code>	267
12.34.2.31	<code>get_input(bussbyte bus)</code>	267
12.34.3	Field Documentation	268
12.34.3.1	<code>m_alsa_seq</code>	268
12.34.3.2	<code>m_num_out_buses</code>	268
12.34.3.3	<code>m_num_in_buses</code>	268
12.34.3.4	<code>m_buses_out</code>	268
12.34.3.5	<code>m_buses_in</code>	268
12.34.3.6	<code>m_bus_announce</code>	268
12.34.3.7	<code>m_buses_out_active</code>	268
12.34.3.8	<code>m_buses_in_active</code>	268
12.34.3.9	<code>m_buses_out_init</code>	268
12.34.3.10	<code>m_buses_in_init</code>	268
12.34.3.11	<code>m_init_clock</code>	268
12.34.3.12	<code>m_init_input</code>	268
12.34.3.13	<code>m_queue</code>	268
12.34.3.14	<code>m_ppqn</code>	268
12.34.3.15	<code>m_beats_per_minute</code>	268
12.34.3.16	<code>m_num_poll_descriptors</code>	269
12.34.3.17	<code>m_poll_descriptors</code>	269
12.34.3.18	<code>m_dumping_input</code>	269
12.34.3.19	<code>m_seq</code>	269
12.34.3.20	<code>m_mutex</code>	269
12.35	<code>seq64::midi_container</code> Class Reference	269
12.35.1	Detailed Description	270

12.35.2 Constructor & Destructor Documentation	270
12.35.2.1 midi_container(sequence &seq)	270
12.35.2.2 ~midi_container()	271
12.35.3 Member Function Documentation	271
12.35.3.1 fill(int tracknumber)	271
12.35.3.2 size() const	271
12.35.3.3 done() const	271
12.35.3.4 put(midibyte b)=0	272
12.35.3.5 get()=0	272
12.35.3.6 position_reset() const	272
12.35.3.7 position() const	272
12.35.3.8 position_increment() const	272
12.35.3.9 add_variable(midipulse v)	272
12.35.3.10 add_long(midipulse x)	272
12.35.4 Field Documentation	272
12.35.4.1 m_sequence	272
12.35.4.2 m_position_for_get	272
12.36seq64::midi_control Class Reference	272
12.36.1 Detailed Description	273
12.36.2 Constructor & Destructor Documentation	274
12.36.2.1 midi_control()	274
12.36.3 Member Function Documentation	274
12.36.3.1 active() const	274
12.36.3.2 inverse_active() const	274
12.36.3.3 status() const	274
12.36.3.4 data() const	274
12.36.3.5 min_value() const	274
12.36.3.6 max_value() const	274
12.36.3.7 set(int values[6])	274
12.36.3.8 set(midibyte values[6])	274

12.36.3.9 match(midibyte status, midibyte data) const	275
12.36.3.10 n_range(midibyte data) const	275
12.36.4 Field Documentation	275
12.36.4.1 m_active	275
12.36.4.2 m_inverse_active	275
12.36.4.3 m_status	275
12.36.4.4 m_data	275
12.36.4.5 m_min_value	275
12.36.4.6 m_max_value	275
12.37 seq64::midi_list Class Reference	275
12.37.1 Member Typedef Documentation	277
12.37.1.1 CharList	277
12.37.2 Constructor & Destructor Documentation	277
12.37.2.1 midi_list(sequence &seq)	277
12.37.2.2 ~midi_list()	277
12.37.3 Member Function Documentation	277
12.37.3.1 size() const	277
12.37.3.2 done() const	277
12.37.3.3 put(midibyte b)	277
12.37.3.4 get()	278
12.37.4 Field Documentation	278
12.37.4.1 m_char_list	278
12.38 seq64::midi_measures Class Reference	278
12.38.1 Detailed Description	278
12.38.2 Constructor & Destructor Documentation	278
12.38.2.1 midi_measures()	278
12.38.2.2 midi_measures(int measures, int beats, int divisions)	278
12.38.3 Member Function Documentation	279
12.38.3.1 measures() const	279
12.38.3.2 measures(int m)	279

12.38.3.3 beats() const	279
12.38.3.4 beats(int b)	279
12.38.3.5 divisions() const	279
12.38.3.6 divisions(int d)	279
12.38.4 Field Documentation	279
12.38.4.1 m_measures	279
12.38.4.2 m_beats	279
12.38.4.3 m_divisions	279
12.39seq64::midi_splitter Class Reference	280
12.39.1 Detailed Description	281
12.39.2 Constructor & Destructor Documentation	281
12.39.2.1 midi_splitter(int ppqn=SEQ64_USE_DEFAULT_PPQN)	281
12.39.2.2 ~midi_splitter()	281
12.39.3 Member Function Documentation	281
12.39.3.1 log_main_sequence(sequence &seq, int seqnum)	281
12.39.3.2 initialize()	281
12.39.3.3 increment(int channel)	281
12.39.3.4 split(perform &p, int screenset)	282
12.39.3.5 ppqn() const	282
12.39.3.6 count() const	282
12.39.3.7 split_channel(const sequence &main_seq, sequence *seq, int channel)	282
12.39.4 Field Documentation	283
12.39.4.1 m_ppqn	283
12.39.4.2 m_use_default_ppqn	283
12.39.4.3 m_smf0_channels_count	283
12.39.4.4 m_smf0_channels	283
12.39.4.5 m_smf0_main_sequence	283
12.39.4.6 m_smf0_seq_number	283
12.40seq64::midi_timing Class Reference	283
12.40.1 Detailed Description	284

12.40.2 Constructor & Destructor Documentation	284
12.40.2.1 midi_timing()	284
12.40.2.2 midi_timing(int bpmminute, int bpmmeasure, int beatwidth, int ppqn)	284
12.40.3 Member Function Documentation	284
12.40.3.1 beats_per_minute() const	284
12.40.3.2 beats_per_minute(int b)	284
12.40.3.3 beats_per_measure() const	284
12.40.3.4 beats_per_measure(int b)	284
12.40.3.5 beat_width() const	285
12.40.3.6 beat_width(int bw)	285
12.40.3.7 ppqn() const	285
12.40.3.8 ppqn(int p)	285
12.40.4 Field Documentation	285
12.40.4.1 m_beats_per_minute	285
12.40.4.2 m_beats_per_measure	285
12.40.4.3 m_beat_width	285
12.40.4.4 m_ppqn	285
12.41 seq64::midi_vector Class Reference	286
12.41.1 Member Typedef Documentation	287
12.41.1.1 CharVector	287
12.41.2 Constructor & Destructor Documentation	287
12.41.2.1 midi_vector(sequence &seq)	287
12.41.2.2 ~midi_vector()	287
12.41.3 Member Function Documentation	287
12.41.3.1 size() const	287
12.41.3.2 done() const	287
12.41.3.3 put(midibyte b)	287
12.41.3.4 get()	288
12.41.4 Field Documentation	288
12.41.4.1 m_char_vector	288

12.42seq64::midibus Class Reference	288
12.42.1 Constructor & Destructor Documentation	290
12.42.1.1 midibus(int localclient, int destclient, int destport, snd_seq_t *seq, const char *client_name, const char *port_name, int id, int queue, int ppqn=SEQ64_US↵SE_DEFAULT_PPQN)	290
12.42.1.2 midibus(int localclient, snd_seq_t *seq, int id, int queue, int ppqn=SEQ64_US↵E_DEFAULT_PPQN)	291
12.42.1.3 ~midibus()	291
12.42.2 Member Function Documentation	291
12.42.2.1 init_out()	291
12.42.2.2 init_in()	291
12.42.2.3 deinit_in()	291
12.42.2.4 init_out_sub()	291
12.42.2.5 init_in_sub()	291
12.42.2.6 print()	292
12.42.2.7 get_name() const	292
12.42.2.8 get_id() const	292
12.42.2.9 play(event *e24, midibyte channel)	292
12.42.2.10sysex(event *e24)	292
12.42.2.11start()	292
12.42.2.12stop()	292
12.42.2.13clock(midipulse tick)	292
12.42.2.14continue_from(midipulse tick)	292
12.42.2.15nit_clock(midipulse tick)	292
12.42.2.16set_clock(clock_e clocktype)	292
12.42.2.17get_clock() const	293
12.42.2.18set_input(bool inputing)	293
12.42.2.19get_input() const	293
12.42.2.20flush()	293
12.42.2.21get_client() const	293
12.42.2.22get_port() const	293
12.42.2.23set_clock_mod(int clockmod)	293

12.42.2.24	<code>get_clock_mod()</code>	293
12.42.3	Friends And Related Function Documentation	293
12.42.3.1	<code>mastermidibus</code>	293
12.42.4	Field Documentation	293
12.42.4.1	<code>m_clock_mod</code>	293
12.42.4.2	<code>m_id</code>	294
12.42.4.3	<code>m_clock_type</code>	294
12.42.4.4	<code>m_inputing</code>	294
12.42.4.5	<code>m_ppqn</code>	294
12.42.4.6	<code>m_seq</code>	294
12.42.4.7	<code>m_dest_addr_client</code>	294
12.42.4.8	<code>m_dest_addr_port</code>	294
12.42.4.9	<code>m_local_addr_client</code>	294
12.42.4.10	<code>m_local_addr_port</code>	294
12.42.4.11	<code>m_queue</code>	294
12.42.4.12	<code>m_name</code>	294
12.42.4.13	<code>m_lasttick</code>	294
12.42.4.14	<code>m_mutex</code>	294
12.43	<code>seq64::midifile</code> Class Reference	294
12.43.1	Detailed Description	297
12.43.2	Constructor & Destructor Documentation	297
12.43.2.1	<code>midifile(const std::string &name, int ppqn=SEQ64_USE_DEFAULT_PPQN, bool oldformat=false, bool globalbgs=true)</code>	297
12.43.2.2	<code>~midifile()</code>	297
12.43.3	Member Function Documentation	297
12.43.3.1	<code>parse(perform &a_perf, int a_screen_set=0)</code>	297
12.43.3.2	<code>write(perform &a_perf)</code>	298
12.43.3.3	<code>error_message() const</code>	299
12.43.3.4	<code>error_is_fatal() const</code>	299
12.43.3.5	<code>ppqn() const</code>	299
12.43.3.6	<code>parse_smf_0(perform &p, int screenset)</code>	299

12.43.3.7 <code>parse_smf_1(perform &p, int screenset, bool is_smf0=false)</code>	299
12.43.3.8 <code>parse_prop_header(int file_size)</code>	299
12.43.3.9 <code>parse_proprietary_track(perform &a_perf, int file_size)</code>	300
12.43.3.10 <code>pow2(int logbase2)</code>	301
12.43.3.11 <code>checklen(midilong len, midibyte type)</code>	301
12.43.3.12 <code>add_trigger(sequence &seq, midishort ppqn)</code>	301
12.43.3.13 <code>read_long()</code>	302
12.43.3.14 <code>read_short()</code>	302
12.43.3.15 <code>read_byte()</code>	302
12.43.3.16 <code>read_varinum()</code>	302
12.43.3.17 <code>write_long(midilong value)</code>	302
12.43.3.18 <code>write_short(midishort value)</code>	303
12.43.3.19 <code>read_byte_array(midibyte *b, int len)</code>	303
12.43.3.20 <code>write_byte(midibyte c)</code>	303
12.43.3.21 <code>write_varinum(midilong)</code>	303
12.43.3.22 <code>write_track_name(const std::string &trackname)</code>	304
12.43.3.23 <code>read_track_name()</code>	304
12.43.3.24 <code>write_seq_number(midishort seqnum)</code>	304
12.43.3.25 <code>read_seq_number()</code>	304
12.43.3.26 <code>write_track_end()</code>	305
12.43.3.27 <code>write_prop_header(midilong tag, long len)</code>	305
12.43.3.28 <code>write_proprietary_track(perform &a_perf)</code>	306
12.43.3.29 <code>varinum_size(long len) const</code>	306
12.43.3.30 <code>prop_item_size(long datalen) const</code>	307
12.43.3.31 <code>track_name_size(const std::string &trackname) const</code>	307
12.43.3.32 <code>errdump(const std::string &msg)</code>	307
12.43.3.33 <code>errdump(const std::string &msg, unsigned long p)</code>	307
12.43.3.34 <code>seq_number_size() const</code>	308
12.43.3.35 <code>track_end_size() const</code>	308
12.43.3.36 <code>syssex_special_id(midibyte ch)</code>	308

12.43.4 Field Documentation	308
12.43.4.1 m_file_size	308
12.43.4.2 m_error_message	308
12.43.4.3 m_error_is_fatal	308
12.43.4.4 m_disable_reported	308
12.43.4.5 m_pos	309
12.43.4.6 m_name	309
12.43.4.7 m_data	309
12.43.4.8 m_char_list	309
12.43.4.9 m_new_format	309
12.43.4.10m_global_bgsequence	309
12.43.4.11m_ppqn	309
12.43.4.12m_use_default_ppqn	309
12.43.4.13m_smf0_splitter	309
12.44seq64::mutex Class Reference	310
12.44.1 Constructor & Destructor Documentation	311
12.44.1.1 mutex()	311
12.44.2 Member Function Documentation	311
12.44.2.1 lock() const	311
12.44.2.2 unlock() const	311
12.44.3 Field Documentation	311
12.44.3.1 sm_recursive_mutex	311
12.44.3.2 m_mutex_lock	311
12.45seq64::editable_event::name_value_t Struct Reference	311
12.45.1 Field Documentation	311
12.45.1.1 event_value	311
12.45.1.2 event_name	311
12.46seq64::options Class Reference	311
12.46.1 Member Enumeration Documentation	313
12.46.1.1 button	313

12.46.2 Constructor & Destructor Documentation	313
12.46.2.1 options(Gtk::Window &parent, perform &p)	313
12.46.3 Member Function Documentation	313
12.46.3.1 perf()	313
12.46.3.2 clock_callback_off(int bus, Gtk::RadioButton *button)	313
12.46.3.3 clock_callback_on(int bus, Gtk::RadioButton *button)	313
12.46.3.4 clock_callback_mod(int bus, Gtk::RadioButton *button)	313
12.46.3.5 clock_mod_callback(Gtk::Adjustment *adj)	313
12.46.3.6 input_callback(int bus, Gtk::Button *button)	313
12.46.3.7 transport_callback(button type, Gtk::Button *button)	313
12.46.3.8 mouse_seq24_callback(Gtk::RadioButton *)	313
12.46.3.9 mouse_fruity_callback(Gtk::RadioButton *)	314
12.46.3.10 mouse_mod4_callback(Gtk::CheckButton *)	314
12.46.3.11 lash_support_callback(Gtk::CheckButton *)	314
12.46.3.12 add_midi_clock_page()	314
12.46.3.13 add_midi_input_page()	314
12.46.3.14 add_keyboard_page()	314
12.46.3.15 add_mouse_page()	314
12.46.3.16 add_jack_sync_page()	314
12.46.4 Field Documentation	314
12.46.4.1 m_tooltips	314
12.46.4.2 m_mainperf	314
12.46.4.3 m_button_ok	314
12.46.4.4 m_button_jack_transport	314
12.46.4.5 m_button_jack_master	314
12.46.4.6 m_button_jack_master_cond	314
12.46.4.7 m_button_jack_connect	314
12.46.4.8 m_button_jack_disconnect	314
12.46.4.9 m_notebook	314
12.47 seq64::optionsfile Class Reference	315

12.47.1 Detailed Description	316
12.47.2 Constructor & Destructor Documentation	316
12.47.2.1 optionsfile(const std::string &name)	316
12.47.2.2 ~optionsfile()	316
12.47.3 Member Function Documentation	316
12.47.3.1 parse(perform &perf)	316
12.47.3.2 write(const perform &perf)	317
12.47.3.3 error_message(const std::string §ionname)	318
12.48seq64::perfedit Class Reference	318
12.48.1 Detailed Description	322
12.48.2 Constructor & Destructor Documentation	323
12.48.2.1 perfedit(perform &p, bool second_perfedit=false, int ppqn=SEQ64_USE_DEFA↵ ULT_PPQN)	323
12.48.2.2 ~perfedit()	323
12.48.3 Member Function Documentation	323
12.48.3.1 init_before_show()	323
12.48.3.2 enqueue_draw(bool forward=true)	323
12.48.3.3 zoom_check(int z)	323
12.48.3.4 enregister_peer(perfedit *peer)	324
12.48.3.5 set_zoom(int z)	324
12.48.3.6 set_beats_per_bar(int bpm)	324
12.48.3.7 set_beat_width(int bw)	324
12.48.3.8 set_snap(int snap)	324
12.48.3.9 set_guides()	324
12.48.3.10grow()	325
12.48.3.11set_looped()	325
12.48.3.12expand()	325
12.48.3.13collapse()	325
12.48.3.14copy()	325
12.48.3.15undo()	325
12.48.3.16popup_menu(Gtk::Menu *menu)	325

12.48.3.17	<code>draw_sequences()</code>	325
12.48.3.18	<code>timeout()</code>	325
12.48.3.19	<code>set_image(bool isrunning)</code>	325
12.48.3.20	<code>start_playing()</code>	326
12.48.3.21	<code>pause_playing()</code>	326
12.48.3.22	<code>stop_playing()</code>	326
12.48.3.23	<code>toggle_playing()</code>	326
12.48.3.24	<code>on_realize()</code>	326
12.48.3.25	<code>on_key_press_event(GdkEventKey *ev)</code>	326
12.48.3.26	<code>on_delete_event(GdkEventAny *)</code>	326
12.48.4	Friends And Related Function Documentation	326
12.48.4.1	<code>update_perfedit_sequences</code>	326
12.48.5	Field Documentation	327
12.48.5.1	<code>m_peer_perfedit</code>	327
12.48.5.2	<code>m_table</code>	327
12.48.5.3	<code>m_vadjust</code>	327
12.48.5.4	<code>m_hadjust</code>	327
12.48.5.5	<code>m_vscroll</code>	327
12.48.5.6	<code>m_hscroll</code>	327
12.48.5.7	<code>m_perfnames</code>	327
12.48.5.8	<code>m_perfroll</code>	327
12.48.5.9	<code>m_perftime</code>	327
12.48.5.10	<code>m_menu_snap</code>	327
12.48.5.11	<code>m_image_play</code>	327
12.48.5.12	<code>m_button_snap</code>	327
12.48.5.13	<code>m_entry_snap</code>	327
12.48.5.14	<code>m_button_stop</code>	327
12.48.5.15	<code>m_button_play</code>	327
12.48.5.16	<code>m_button_loop</code>	328
12.48.5.17	<code>m_button_expand</code>	328

12.48.5.18m_button_collapse	328
12.48.5.19m_button_copy	328
12.48.5.20m_button_grow	328
12.48.5.21m_button_undo	328
12.48.5.22m_button_bpm	328
12.48.5.23m_entry_bpm	328
12.48.5.24m_button_bw	328
12.48.5.25m_entry_bw	328
12.48.5.26m_hbox	328
12.48.5.27m_hlbox	328
12.48.5.28m_tooltips	328
12.48.5.29m_menu_bpm	328
12.48.5.30m_menu_bw	328
12.48.5.31m_snap	328
12.48.5.32m_bpm	328
12.48.5.33m_bw	328
12.48.5.34m_ppqn	329
12.48.5.35m_is_running	329
12.48.5.36m_standard_bpm	329
12.49seq64::perfnames Class Reference	329
12.49.1 Detailed Description	331
12.49.2 Constructor & Destructor Documentation	331
12.49.2.1 perfnames(perform &p, perfedit &parent, Gtk::Adjustment &vadjust)	331
12.49.2.2 ~perfnames()	331
12.49.3 Member Function Documentation	332
12.49.3.1 redraw_dirty_sequences()	332
12.49.3.2 enqueue_draw()	332
12.49.3.3 convert_y(int y)	332
12.49.3.4 draw_sequences()	332
12.49.3.5 draw_sequence(int sequence)	332

12.49.3.6	change_vert()	332
12.49.3.7	redraw(int sequence)	333
12.49.3.8	on_realize()	333
12.49.3.9	on_expose_event(GdkEventExpose *ev)	333
12.49.3.10	on_button_press_event(GdkEventButton *ev)	333
12.49.3.11	on_button_release_event(GdkEventButton *ev)	334
12.49.3.12	on_size_allocate(Gtk::Allocation &)	335
12.49.3.13	on_scroll_event(GdkEventScroll *ev)	335
12.49.4	Friends And Related Function Documentation	335
12.49.4.1	perfedir	335
12.49.5	Field Documentation	335
12.49.5.1	m_parent	335
12.49.5.2	m_names_chars	335
12.49.5.3	m_char_w	336
12.49.5.4	m_setbox_w	336
12.49.5.5	m_namebox_w	336
12.49.5.6	m_names_x	336
12.49.5.7	m_names_y	336
12.49.5.8	m_xy_offset	336
12.49.5.9	m_seqs_in_set	336
12.49.5.10	m_sequence_max	336
12.49.5.11	m_sequence_offset	336
12.49.5.12	m_sequence_active	336
12.50	seq64::perform Class Reference	336
12.50.1	Detailed Description	346
12.50.2	Constructor & Destructor Documentation	346
12.50.2.1	perform(gui_assistant &mygui, int ppqn=SEQ64_USE_DEFAULT_PPQN)	346
12.50.2.2	~perform()	347
12.50.3	Member Function Documentation	347
12.50.3.1	is_modified() const	347

12.50.3.2 modify()	347
12.50.3.3 sequence_count() const	347
12.50.3.4 sequence_max() const	347
12.50.3.5 is_control_status() const	347
12.50.3.6 set_edit_sequence(int seqnum)	347
12.50.3.7 unset_edit_sequence(int seqnum)	347
12.50.3.8 is_edit_sequence(int seqnum) const	347
12.50.3.9 get_beats_per_bar() const	348
12.50.3.10 set_beats_per_bar(int bpm)	348
12.50.3.11 get_beat_width() const	348
12.50.3.12 set_beat_width(int bw)	348
12.50.3.13 gui() const	348
12.50.3.14 gui()	348
12.50.3.15 keys() const	348
12.50.3.16 keys()	348
12.50.3.17 master_bus()	348
12.50.3.18 s_running() const	348
12.50.3.19 s_jack_running() const	348
12.50.3.20 s_paused() const	348
12.50.3.21 is_pausable() const	348
12.50.3.22 enregister(performcallback *pfc)	348
12.50.3.23 clear_all()	349
12.50.3.24 launch(int ppqn)	349
12.50.3.25 new_sequence(int seq)	349
12.50.3.26 add_sequence(sequence *seq, int perf)	349
12.50.3.27 delete_sequence(int seq)	350
12.50.3.28 s_sequence_in_edit(int seq)	350
12.50.3.29 clear_sequence_triggers(int seq)	350
12.50.3.30 print_triggers() const	350
12.50.3.31 finish()	350

12.50.3.32	<code>get_tick()</code> const	351
12.50.3.33	<code>get_jack_tick()</code> const	351
12.50.3.34	<code>set_jack_tick</code> (midipulse tick)	351
12.50.3.35	<code>set_left_tick</code> (midipulse tick, bool setstart=true)	351
12.50.3.36	<code>get_left_tick()</code> const	351
12.50.3.37	<code>set_start_tick</code> (midipulse tick)	351
12.50.3.38	<code>set_right_tick</code> (midipulse tick, bool setstart=true)	351
12.50.3.39	<code>get_right_tick()</code> const	351
12.50.3.40	<code>move_triggers</code> (bool direction)	352
12.50.3.41	<code>copy_triggers()</code>	352
12.50.3.42	<code>push_trigger_undo()</code>	352
12.50.3.43	<code>pop_trigger_undo()</code>	352
12.50.3.44	<code>split_trigger</code> (int seqnum, midipulse tick)	352
12.50.3.45	<code>get_max_trigger()</code>	352
12.50.3.46	<code>collapse()</code>	352
12.50.3.47	<code>copy()</code>	352
12.50.3.48	<code>expand()</code>	352
12.50.3.49	<code>midi_control_toggle</code> (int seq)	352
12.50.3.50	<code>midi_control_on</code> (int seq)	353
12.50.3.51	<code>midi_control_off</code> (int seq)	353
12.50.3.52	<code>handle_midi_control</code> (int control, bool state)	353
12.50.3.53	<code>get_screen_set_notepad</code> (int screen_set) const	353
12.50.3.54	<code>current_screen_set_notepad()</code> const	354
12.50.3.55	<code>set_screen_set_notepad</code> (int screenset, const std::string ¬e)	354
12.50.3.56	<code>set_screen_set_notepad</code> (const std::string ¬e)	354
12.50.3.57	<code>get_screenset()</code> const	354
12.50.3.58	<code>set_playing_screenset()</code>	354
12.50.3.59	<code>set_screenset</code> (int ss)	354
12.50.3.60	<code>get_playing_screenset()</code> const	355
12.50.3.61	<code>any_group_unmutes()</code> const	355

12.50.3.62	<code>mute_group_tracks()</code>	355
12.50.3.63	<code>select_and_mute_group(int g_group)</code>	355
12.50.3.64	<code>set_mode_group_mute()</code>	355
12.50.3.65	<code>unset_mode_group_mute()</code>	355
12.50.3.66	<code>select_group_mute(int g_mute)</code>	355
12.50.3.67	<code>set_mode_group_learn()</code>	355
12.50.3.68	<code>unset_mode_group_learn()</code>	356
12.50.3.69	<code>s_group_learning()</code>	356
12.50.3.70	<code>set_and_copy_mute_group(int group)</code>	356
12.50.3.71	<code>start(bool state)</code>	356
12.50.3.72	<code>stop()</code>	356
12.50.3.73	<code>start_jack()</code>	356
12.50.3.74	<code>stop_jack()</code>	356
12.50.3.75	<code>position_jack(bool state)</code>	356
12.50.3.76	<code>off_sequences()</code>	356
12.50.3.77	<code>all_notes_off()</code>	356
12.50.3.78	<code>set_active(int seq, bool active)</code>	357
12.50.3.79	<code>set_was_active(int seq)</code>	357
12.50.3.80	<code>s_dirty_main(int seq)</code>	357
12.50.3.81	<code>is_dirty_edit(int seq)</code>	357
12.50.3.82	<code>s_dirty_perf(int seq)</code>	357
12.50.3.83	<code>s_dirty_names(int seq)</code>	358
12.50.3.84	<code>s_active(int seq) const</code>	358
12.50.3.85	<code>get_sequence(int seq)</code>	358
12.50.3.86	<code>reset_sequences(bool pause=false)</code>	359
12.50.3.87	<code>play(midipulse tick)</code>	359
12.50.3.88	<code>set_orig_ticks(midipulse tick)</code>	359
12.50.3.89	<code>set_beats_per_minute(int bpm)</code>	359
12.50.3.90	<code>get_beats_per_minute()</code>	359
12.50.3.91	<code>set_looping(bool looping)</code>	360

12.50.3.92	set_sequence_control_status(int status)	361
12.50.3.93	unset_sequence_control_status(int status)	361
12.50.3.94	sequence_playing_toggle(int seq)	361
12.50.3.95	sequence_playing_change(int seq, bool on)	361
12.50.3.96	sequence_playing_on(int seq)	361
12.50.3.97	sequence_playing_off(int seq)	362
12.50.3.98	mute_all_tracks(bool flag=true)	362
12.50.3.99	toggle_all_tracks()	362
12.50.3.100	mute_screenset(int ss, bool flag=true)	362
12.50.3.101	output_func()	362
12.50.3.102	input_func()	363
12.50.3.103	set_group_mute_state(int gtrack, bool muted)	363
12.50.3.104	get_group_mute_state(int gtrack)	363
12.50.3.105	set_offset(int offset)	363
12.50.3.106	get_offset() const	364
12.50.3.107	save_playing_state()	364
12.50.3.108	restore_playing_state()	364
12.50.3.109	key_name(unsigned int k) const	364
12.50.3.110	get_key_events()	364
12.50.3.111	get_key_groups()	364
12.50.3.112	get_key_events_rev()	364
12.50.3.113	get_key_groups_rev()	364
12.50.3.114	show_ui_sequence_key() const	364
12.50.3.115	show_ui_sequence_key(bool flag)	364
12.50.3.116	show_ui_sequence_number() const	364
12.50.3.117	show_ui_sequence_number(bool flag)	365
12.50.3.118	lookup_keyevent_key(int seqnum)	365
12.50.3.119	lookup_keyevent_seq(unsigned int keycode)	365
12.50.3.120	lookup_keygroup_key(long groupnum)	365
12.50.3.121	lookup_keygroup_group(unsigned int keycode)	366

12.50.3.122	start_playing(bool songmode=false)	367
12.50.3.123	pause_playing()	367
12.50.3.124	stop_playing()	368
12.50.3.125	start_key(bool songmode=false)	368
12.50.3.126	pause_key(bool songmode=false)	368
12.50.3.127	stop_key()	368
12.50.3.128	learn_toggle()	368
12.50.3.129	decrement_beats_per_minute()	368
12.50.3.130	increment_beats_per_minute()	368
12.50.3.131	decrement_screenset()	368
12.50.3.132	increment_screenset()	368
12.50.3.133	highlight(const sequence &seq) const	368
12.50.3.134	smf_0(const sequence &seq) const	369
12.50.3.135	sequence_key(int seq)	369
12.50.3.136	sequence_label(const sequence &seq)	369
12.50.3.137	set_input_bus(int bus, bool input_active)	370
12.50.3.138	mainwnd_key_event(const keystroke &k)	370
12.50.3.139	perfroll_key_event(const keystroke &k, int drop_sequence)	370
12.50.3.140	playback_key_event(const keystroke &k, bool songmode=false)	371
12.50.3.141	max_active_set() const	371
12.50.3.142	launch_input_thread()	371
12.50.3.143	launch_output_thread()	371
12.50.3.144	init_jack()	371
12.50.3.145	reinit_jack()	372
12.50.3.146	seq_in_playing_screen(int seq)	372
12.50.3.147	is_modified(bool flag)	372
12.50.3.148	is_midi_control_valid(int seq) const	372
12.50.3.149	is_screenset_valid(int screenset) const	372
12.50.3.150	set_running(bool running)	373
12.50.3.151	set_playback_mode(bool playbackmode)	373

12.50.3.152	mute_group_offset(int track)	373
12.50.3.153	seq_valid(int seq) const	373
12.50.3.154	mseq_valid(int seq) const	373
12.50.3.155	stall_sequence(sequence *seq, int seqnum)	374
12.50.3.156	ner_start(bool state)	374
12.50.3.157	ner_stop()	375
12.50.3.158	amp_track(int track) const	375
12.50.3.159	set_all_key_events()	375
12.50.3.160	set_all_key_groups()	375
12.50.3.161	set_key_event(unsigned int keycode, long sequence_slot)	375
12.50.3.162	set_key_group(unsigned int keycode, long group_slot)	375
12.50.4	Friends And Related Function Documentation	376
12.50.4.1	jack_assistant	376
12.50.4.2	keybindentry	376
12.50.4.3	midifile	376
12.50.4.4	optionsfile	376
12.50.4.5	options	376
12.50.4.6	jack_sync_callback	376
12.50.5	Field Documentation	376
12.50.5.1	sm_mc_dummy	376
12.50.5.2	m_gui_support	376
12.50.5.3	m_mute_group	376
12.50.5.4	m_tracks_mute_state	377
12.50.5.5	m_mode_group	377
12.50.5.6	m_mode_group_learn	377
12.50.5.7	m_mute_group_selected	377
12.50.5.8	m_playing_screen	377
12.50.5.9	m_playscreen_offset	377
12.50.5.10	m_seqs	377
12.50.5.11	m_seqs_active	377

12.50.5.12m_was_active_main	377
12.50.5.13m_was_active_edit	378
12.50.5.14m_was_active_perf	378
12.50.5.15m_was_active_names	378
12.50.5.16m_sequence_state	378
12.50.5.17m_master_bus	378
12.50.5.18m_out_thread	378
12.50.5.19m_in_thread	378
12.50.5.20m_out_thread_launched	378
12.50.5.21m_in_thread_launched	378
12.50.5.22m_running	378
12.50.5.23m_inputting	378
12.50.5.24m_outputting	378
12.50.5.25m_looping	378
12.50.5.26m_playback_mode	378
12.50.5.27m_ppqn	379
12.50.5.28m_beats_per_bar	379
12.50.5.29m_beat_width	379
12.50.5.30m_one_measure	379
12.50.5.31m_left_tick	379
12.50.5.32m_right_tick	379
12.50.5.33m_starting_tick	379
12.50.5.34m_tick	379
12.50.5.35m_jack_tick	380
12.50.5.36m_usemidiclock	380
12.50.5.37m_midiclockrunning	380
12.50.5.38m_midiclocktick	380
12.50.5.39m_midiclockpos	380
12.50.5.40m_is_paused	380
12.50.5.41m_screen_set_notepad	380

12.50.5.42	m_midi_cc_toggle	380
12.50.5.43	m_midi_cc_on	380
12.50.5.44	m_midi_cc_off	380
12.50.5.45	m_offset	380
12.50.5.46	m_control_status	380
12.50.5.47	m_screenset	380
12.50.5.48	m_seqs_in_set	380
12.50.5.49	m_max_sets	380
12.50.5.50	m_sequence_count	381
12.50.5.51	m_sequence_max	381
12.50.5.52	m_edit_sequence	381
12.50.5.53	m_is_modified	381
12.50.5.54	m_condition_var	381
12.50.5.55	m_jack_asst	381
12.50.5.56	m_notify	381
12.51	seq64::performcallback Struct Reference	381
12.51.1	Detailed Description	382
12.51.2	Member Function Documentation	383
12.51.2.1	on_grouplearnchange(bool)	383
12.52	seq64::perfroll Class Reference	383
12.52.1	Constructor & Destructor Documentation	388
12.52.1.1	perfroll(perform &perf, perfedit &parent, Gtk::Adjustment &hadjust, Gtk::Adjustment &vadjust, int ppqn=SEQ64_USE_DEFAULT_PPQN)	388
12.52.1.2	~perfroll()	388
12.52.2	Member Function Documentation	388
12.52.2.1	set_guides(int snap, int measure, int beat)	388
12.52.2.2	update_sizes()	388
12.52.2.3	init_before_show()	388
12.52.2.4	fill_background_pixmap()	389
12.52.2.5	increment_size()	389
12.52.2.6	draw_all()	389

12.52.2.7 follow_progress()	389
12.52.2.8 redraw_progress()	389
12.52.2.9 draw_progress()	389
12.52.2.10 redraw_dirty_sequences()	389
12.52.2.11 set_ppqn(int ppqn)	389
12.52.2.12 convert_xy(int x, int y, midipulse &ticks, int &seq)	389
12.52.2.13 convert_x(int x, midipulse &ticks)	390
12.52.2.14 snap_x(int &x)	390
12.52.2.15 draw_sequence_on(int seqnum)	390
12.52.2.16 draw_background_on(int seqnum)	390
12.52.2.17 draw_drawable_row(long y)	390
12.52.2.18 change_horz()	390
12.52.2.19 change_vert()	390
12.52.2.20 split_trigger(int sequence, midipulse tick)	390
12.52.2.21 enqueue_draw()	390
12.52.2.22 set_zoom(int z)	391
12.52.2.23 horizontal_adjust(double step)	391
12.52.2.24 vertical_adjust(double step)	392
12.52.2.25 horizontal_set(double value)	392
12.52.2.26 vertical_set(double value)	392
12.52.2.27 on_realize()	392
12.52.2.28 on_expose_event(GdkEventExpose *ev)	392
12.52.2.29 on_button_press_event(GdkEventButton *ev)	393
12.52.2.30 on_button_release_event(GdkEventButton *ev)	393
12.52.2.31 on_motion_notify_event(GdkEventMotion *ev)	393
12.52.2.32 on_scroll_event(GdkEventScroll *ev)	393
12.52.2.33 on_focus_in_event(GdkEventFocus *ev)	393
12.52.2.34 on_focus_out_event(GdkEventFocus *ev)	393
12.52.2.35 on_size_allocate(Gtk::Allocation &al)	393
12.52.2.36 on_key_press_event(GdkEventKey *ev)	394

12.52.2.37	<code>n_size_request(GtkRequisition *)</code>	394
12.52.3	Friends And Related Function Documentation	394
12.52.3.1	<code>FruityPerfInput</code>	394
12.52.3.2	<code>Seq24PerfInput</code>	394
12.52.3.3	<code>perfedit</code>	394
12.52.4	Field Documentation	394
12.52.4.1	<code>m_parent</code>	394
12.52.4.2	<code>m_h_page_increment</code>	394
12.52.4.3	<code>m_v_page_increment</code>	395
12.52.4.4	<code>m_snap</code>	395
12.52.4.5	<code>m_ppqn</code>	395
12.52.4.6	<code>m_page_factor</code>	395
12.52.4.7	<code>m_divs_per_beat</code>	395
12.52.4.8	<code>m_ticks_per_bar</code>	395
12.52.4.9	<code>m_perf_scale_x</code>	395
12.52.4.10	<code>m_zoom</code>	395
12.52.4.11	<code>m_names_y</code>	395
12.52.4.12	<code>m_background_x</code>	395
12.52.4.13	<code>m_size_box_w</code>	395
12.52.4.14	<code>m_measure_length</code>	395
12.52.4.15	<code>m_beat_length</code>	395
12.52.4.16	<code>m_old_progress_ticks</code>	395
12.52.4.17	<code>m_4bar_offset</code>	396
12.52.4.18	<code>m_sequence_offset</code>	396
12.52.4.19	<code>m_roll_length_ticks</code>	396
12.52.4.20	<code>m_drop_tick</code>	396
12.52.4.21	<code>m_drop_tick_trigger_offset</code>	396
12.52.4.22	<code>m_drop_sequence</code>	396
12.52.4.23	<code>m_sequence_max</code>	396
12.52.4.24	<code>m_sequence_active</code>	396

12.52.4.25m_fruity_interaction	396
12.52.4.26m_seq24_interaction	397
12.52.4.27m_moving	397
12.52.4.28m_growing	397
12.52.4.29m_grow_direction	397
12.53seq64::perftime Class Reference	397
12.53.1 Constructor & Destructor Documentation	400
12.53.1.1 perftime(perform &perf, perfedit &parent, Gtk::Adjustment &hadjust, int ppqn=SEQ64_USE_DEFAULT_PPQN)	400
12.53.1.2 ~perftime()	401
12.53.2 Member Function Documentation	401
12.53.2.1 reset()	401
12.53.2.2 set_scale(int scale)	401
12.53.2.3 set_guides(int snap, int measure)	401
12.53.2.4 increment_size()	401
12.53.2.5 enqueue_draw()	401
12.53.2.6 set_zoom(int z)	401
12.53.2.7 draw_background()	401
12.53.2.8 draw_progress_on_window()	401
12.53.2.9 change_horz()	402
12.53.2.10set_ppqn(int ppqn)	402
12.53.2.11tick_to_pixel(midipulse tick)	402
12.53.2.12pixel_to_tick(long pixel)	402
12.53.2.13tick_offset()	402
12.53.2.14update_sizes()	403
12.53.2.15dle_progress()	403
12.53.2.16update_pixmap()	403
12.53.2.17draw_pixmap_on_window()	403
12.53.2.18on_realize()	403
12.53.2.19on_expose_event(GtkEventExpose *ev)	403
12.53.2.20on_button_press_event(GtkEventButton *ev)	403

12.53.2.21	<code>on_size_allocate(Gtk::Allocation &r)</code>	404
12.53.2.22	<code>on_button_release_event(GdkEventButton *)</code>	404
12.53.2.23	<code>key_press_event(GdkEventKey *ev)</code>	404
12.53.3	Friends And Related Function Documentation	404
12.53.3.1	<code>perfedit</code>	404
12.53.4	Field Documentation	404
12.53.4.1	<code>m_parent</code>	404
12.53.4.2	<code>m_4bar_offset</code>	404
12.53.4.3	<code>m_tick_offset</code>	405
12.53.4.4	<code>m_ppqn</code>	405
12.53.4.5	<code>m_snap</code>	405
12.53.4.6	<code>m_measure_length</code>	405
12.53.4.7	<code>m_left_marker_tick</code>	405
12.53.4.8	<code>m_right_marker_tick</code>	405
12.53.4.9	<code>m_perf_scale_x</code>	405
12.53.4.10	<code>m_timearea_y</code>	405
12.54	<code>seq64::rc_settings</code> Class Reference	405
12.54.1	Constructor & Destructor Documentation	408
12.54.1.1	<code>rc_settings()</code>	408
12.54.1.2	<code>rc_settings(const rc_settings &rhs)</code>	408
12.54.2	Member Function Documentation	408
12.54.2.1	<code>operator=(const rc_settings &rhs)</code>	408
12.54.2.2	<code>config_filespec() const</code>	409
12.54.2.3	<code>user_filespec() const</code>	409
12.54.2.4	<code>set_defaults()</code>	409
12.54.2.5	<code>auto_option_save() const</code>	409
12.54.2.6	<code>auto_option_save(bool flag)</code>	409
12.54.2.7	<code>legacy_format() const</code>	409
12.54.2.8	<code>legacy_format(bool flag)</code>	409
12.54.2.9	<code>lash_support() const</code>	409

12.54.2.10	dash_support(bool flag)	409
12.54.2.11	allow_mod4_mode() const	409
12.54.2.12	allow_mod4_mode(bool flag)	409
12.54.2.13	show_midi() const	409
12.54.2.14	show_midi(bool flag)	409
12.54.2.15	priority() const	409
12.54.2.16	priority(bool flag)	409
12.54.2.17	stats() const	409
12.54.2.18	stats(bool flag)	410
12.54.2.19	pass_sysex() const	410
12.54.2.20	pass_sysex(bool flag)	410
12.54.2.21	with_jack_transport() const	410
12.54.2.22	with_jack_transport(bool flag)	410
12.54.2.23	with_jack_master() const	410
12.54.2.24	with_jack_master(bool flag)	410
12.54.2.25	with_jack_master_cond() const	410
12.54.2.26	with_jack_master_cond(bool flag)	410
12.54.2.27	with_jack() const	410
12.54.2.28	jack_start_mode() const	410
12.54.2.29	jack_start_mode(bool flag)	410
12.54.2.30	manual_alsa_ports() const	410
12.54.2.31	manual_alsa_ports(bool flag)	410
12.54.2.32	reveal_alsa_ports() const	410
12.54.2.33	reveal_alsa_ports(bool flag)	410
12.54.2.34	s_pattern_playing() const	410
12.54.2.35	s_pattern_playing(bool flag)	410
12.54.2.36	print_keys() const	410
12.54.2.37	print_keys(bool flag)	410
12.54.2.38	device_ignore() const	410
12.54.2.39	device_ignore(bool flag)	410

12.54.2.40	<code>device_ignore_num()</code> const	410
12.54.2.41	<code>interaction_method()</code> const	411
12.54.2.42	<code>filename()</code> const	411
12.54.2.43	<code>jack_session_uuid()</code> const	411
12.54.2.44	<code>last_used_dir()</code> const	411
12.54.2.45	<code>config_directory()</code> const	411
12.54.2.46	<code>config_filename()</code> const	411
12.54.2.47	<code>user_filename()</code> const	411
12.54.2.48	<code>config_filename_alt()</code> const	411
12.54.2.49	<code>user_filename_alt()</code> const	411
12.54.2.50	<code>device_ignore_num(int value)</code>	411
12.54.2.51	<code>interaction_method(interaction_method_t value)</code>	411
12.54.2.52	<code>filename(const std::string &value)</code>	411
12.54.2.53	<code>jack_session_uuid(const std::string &value)</code>	411
12.54.2.54	<code>last_used_dir(const std::string &value)</code>	411
12.54.2.55	<code>config_directory(const std::string &value)</code>	412
12.54.2.56	<code>set_config_files(const std::string &value)</code>	412
12.54.2.57	<code>config_filename(const std::string &value)</code>	412
12.54.2.58	<code>user_filename(const std::string &value)</code>	412
12.54.2.59	<code>config_filename_alt(const std::string &value)</code>	412
12.54.2.60	<code>user_filename_alt(const std::string &value)</code>	412
12.54.2.61	<code>home_config_directory()</code> const	413
12.54.3	Field Documentation	413
12.54.3.1	<code>m_auto_option_save</code>	413
12.54.3.2	<code>m_legacy_format</code>	413
12.54.3.3	<code>m_lash_support</code>	413
12.54.3.4	<code>m_allow_mod4_mode</code>	413
12.54.3.5	<code>m_show_midi</code>	413
12.54.3.6	<code>m_priority</code>	413
12.54.3.7	<code>m_stats</code>	413

12.54.3.8 m_pass_sysex	413
12.54.3.9 m_with_jack_transport	413
12.54.3.10m_with_jack_master	413
12.54.3.11m_with_jack_master_cond	413
12.54.3.12m_jack_start_mode	413
12.54.3.13m_manual_alsa_ports	413
12.54.3.14m_reveal_alsa_ports	413
12.54.3.15m_is_pattern_playing	413
12.54.3.16m_print_keys	413
12.54.3.17m_device_ignore	414
12.54.3.18m_device_ignore_num	414
12.54.3.19m_interaction_method	414
12.54.3.20m_filename	414
12.54.3.21m_jack_session_uuid	414
12.54.3.22m_last_used_dir	414
12.54.3.23m_config_directory	414
12.54.3.24m_config_filename	414
12.54.3.25m_user_filename	414
12.54.3.26m_config_filename_alt	414
12.54.3.27m_user_filename_alt	414
12.55seq64::rect Class Reference	414
12.55.1 Field Documentation	414
12.55.1.1 x	414
12.55.1.2 y	414
12.55.1.3 height	414
12.55.1.4 width	414
12.56seq64::gui_drawingarea_gtk2::rect Struct Reference	414
12.56.1 Field Documentation	415
12.56.1.1 x	415
12.56.1.2 y	415

12.56.1.3 height	415
12.56.1.4 width	415
12.57seq64::Seq24PerfInput Class Reference	415
12.57.1 Constructor & Destructor Documentation	416
12.57.1.1 Seq24PerfInput()	416
12.57.2 Member Function Documentation	416
12.57.2.1 on_button_press_event(GdkEventButton *a_ev, perfroll &roll)	416
12.57.2.2 on_button_release_event(GdkEventButton *a_ev, perfroll &roll)	417
12.57.2.3 on_motion_notify_event(GdkEventMotion *a_ev, perfroll &roll)	417
12.57.2.4 set_adding(bool a_adding, perfroll &roll)	417
12.57.2.5 handle_motion_key(bool is_left, perfroll &roll)	417
12.57.2.6 is_adding() const	418
12.57.3 Friends And Related Function Documentation	418
12.57.3.1 perfroll	418
12.57.4 Field Documentation	418
12.57.4.1 m_adding	418
12.57.4.2 m_effective_tick	418
12.58seq64::Seq24SeqEventInput Struct Reference	418
12.58.1 Constructor & Destructor Documentation	418
12.58.1.1 Seq24SeqEventInput()	418
12.58.2 Member Function Documentation	418
12.58.2.1 set_adding(bool adding, sequevent &ths)	418
12.58.2.2 on_button_press_event(GdkEventButton *ev, sequevent &ths)	419
12.58.2.3 on_button_release_event(GdkEventButton *ev, sequevent &ths)	419
12.58.2.4 on_motion_notify_event(GdkEventMotion *ev, sequevent &ths)	419
12.58.3 Field Documentation	419
12.58.3.1 m_adding	420
12.59seq64::seqdata Class Reference	420
12.59.1 Constructor & Destructor Documentation	422
12.59.1.1 seqdata(sequence &seq, perform &p, int zoom, Gtk::Adjustment &hadjust)	422

12.59.1.2 ~seqdata()	423
12.59.2 Member Function Documentation	423
12.59.2.1 reset()	423
12.59.2.2 redraw()	423
12.59.2.3 set_zoom(int a_zoom)	423
12.59.2.4 set_data_type(midibyte status, midibyte control)	423
12.59.2.5 idle_redraw()	423
12.59.2.6 update_sizes()	424
12.59.2.7 update_pixmap()	424
12.59.2.8 draw_line_on_window()	424
12.59.2.9 xy_to_rect(int x1, int y1, int x2, int y2, int &rx, int &ry, int &rw, int &rh)	424
12.59.2.10 draw_events_on(Glib::RefPtr< Gdk::Drawable > drawable)	424
12.59.2.11 change_horz()	424
12.59.2.12 convert_x(int x, midipulse &tick)	424
12.59.2.13 render_number(Glib::RefPtr< Gdk::Pixmap > &pixmap, int x, int y, const char *const num)	424
12.59.2.14 draw_events_on_pixmap()	425
12.59.2.15 draw_pixmap_on_window()	425
12.59.2.16 on_realize()	425
12.59.2.17 on_expose_event(GdkEventExpose *ev)	425
12.59.2.18 on_button_press_event(GdkEventButton *ev)	425
12.59.2.19 on_button_release_event(GdkEventButton *ev)	425
12.59.2.20 on_motion_notify_event(GdkEventMotion *ev)	426
12.59.2.21 on_leave_notify_event(GdkEventCrossing *ev)	426
12.59.2.22 on_scroll_event(GdkEventScroll *ev)	426
12.59.2.23 on_size_allocate(Gtk::Allocation &)	426
12.59.3 Friends And Related Function Documentation	427
12.59.3.1 seqroll	427
12.59.3.2 sequevent	427
12.59.4 Field Documentation	427
12.59.4.1 m_seq	427

12.59.4.2 m_zoom	427
12.59.4.3 m_scroll_offset_ticks	427
12.59.4.4 m_scroll_offset_x	427
12.59.4.5 m_number_w	427
12.59.4.6 m_number_h	427
12.59.4.7 m_number_offset_y	427
12.59.4.8 m_status	427
12.59.4.9 m_cc	427
12.59.4.10m_numbers	427
12.59.4.11m_old	428
12.59.4.12m_dragging	428
12.60seq64::seqedit Class Reference	428
12.60.1 Detailed Description	434
12.60.2 Constructor & Destructor Documentation	434
12.60.2.1 seqedit(perform &perf, sequence &seq, int pos, int ppqn=SEQ64_USE_DEFA← ULT_PPQN)	434
12.60.2.2 ~seqedit()	435
12.60.3 Member Function Documentation	435
12.60.3.1 set_zoom(int zoom)	435
12.60.3.2 set_snap(int snap)	435
12.60.3.3 set_note_length(int note_length)	435
12.60.3.4 set_beats_per_bar(int bpm)	435
12.60.3.5 set_beat_width(int bw)	436
12.60.3.6 set_rec_vol(int recvol)	436
12.60.3.7 horizontal_adjust(double step)	436
12.60.3.8 vertical_adjust(double step)	436
12.60.3.9 horizontal_set(double value)	436
12.60.3.10vertical_set(double value)	436
12.60.3.11set_measures(int lim)	436
12.60.3.12apply_length(int bpm, int bw, int measures)	437
12.60.3.13get_measures()	437

12.60.3.14	set_midi_channel(int midichannel)	437
12.60.3.15	set_midi_bus(int midibus)	437
12.60.3.16	set_scale(int scale)	437
12.60.3.17	set_key(int note)	437
12.60.3.18	set_background_sequence(int seq)	437
12.60.3.19	name_change_callback()	438
12.60.3.20	play_change_callback()	438
12.60.3.21	record_change_callback()	438
12.60.3.22	q_rec_change_callback()	438
12.60.3.23	thru_change_callback()	438
12.60.3.24	undo_callback()	438
12.60.3.25	redo_callback()	438
12.60.3.26	set_data_type(midibyte status, midibyte control=0)	438
12.60.3.27	update_all_windows()	438
12.60.3.28	fill_top_bar()	438
12.60.3.29	create_menus()	438
12.60.3.30	popup_menu(Gtk::Menu *menu)	439
12.60.3.31	popup_event_menu()	439
12.60.3.32	popup_midibus_menu()	439
12.60.3.33	popup_sequence_menu()	439
12.60.3.34	popup_tool_menu()	439
12.60.3.35	popup_midich_menu()	439
12.60.3.36	create_menu_image(bool state=false)	439
12.60.3.37	timeout()	439
12.60.3.38	do_action(int action, int var)	440
12.60.3.39	mouse_action(mouse_action_e action)	440
12.60.3.40	change_focus(bool set_it=true)	440
12.60.3.41	handle_close()	440
12.60.3.42	on_realize()	440
12.60.3.43	on_set_focus(Widget *focus)	440

12.60.3.44	<code>on_focus_in_event(GdkEventFocus *)</code>	440
12.60.3.45	<code>on_focus_out_event(GdkEventFocus *)</code>	440
12.60.3.46	<code>on_delete_event(GdkEventAny *event)</code>	440
12.60.3.47	<code>on_scroll_event(GdkEventScroll *ev)</code>	441
12.60.3.48	<code>on_key_press_event(GdkEventKey *ev)</code>	441
12.60.4	Field Documentation	441
12.60.4.1	<code>seqmenu</code>	441
12.60.4.2	<code>m_initial_snap</code>	441
12.60.4.3	<code>m_initial_note_length</code>	442
12.60.4.4	<code>m_initial_zoom</code>	442
12.60.4.5	<code>m_zoom</code>	442
12.60.4.6	<code>m_snap</code>	442
12.60.4.7	<code>m_note_length</code>	442
12.60.4.8	<code>m_scale</code>	442
12.60.4.9	<code>m_key</code>	442
12.60.4.10	<code>m_bgsequence</code>	442
12.60.4.11	<code>m_measures</code>	442
12.60.4.12	<code>m_ppqn</code>	442
12.60.4.13	<code>m_seq</code>	442
12.60.4.14	<code>m_menuubar</code>	442
12.60.4.15	<code>m_menu_tools</code>	443
12.60.4.16	<code>m_menu_zoom</code>	443
12.60.4.17	<code>m_menu_snap</code>	443
12.60.4.18	<code>m_menu_note_length</code>	443
12.60.4.19	<code>m_menu_length</code>	443
12.60.4.20	<code>m_menu_midich</code>	443
12.60.4.21	<code>m_menu_midibus</code>	443
12.60.4.22	<code>m_menu_data</code>	443
12.60.4.23	<code>m_menu_key</code>	443
12.60.4.24	<code>m_menu_scale</code>	443

12.60.4.25m_menu_sequences	443
12.60.4.26m_menu_bpm	443
12.60.4.27m_menu_bw	443
12.60.4.28m_menu_rec_vol	443
12.60.4.29m_vadjust	443
12.60.4.30m_hadjust	443
12.60.4.31m_vscroll_new	443
12.60.4.32m_hscroll_new	443
12.60.4.33m_seqkeys_wid	443
12.60.4.34m_seqtime_wid	443
12.60.4.35m_seqdata_wid	444
12.60.4.36m_sequevent_wid	444
12.60.4.37m_seqroll_wid	444
12.60.4.38m_table	444
12.60.4.39m_vbox	444
12.60.4.40m_hbox	444
12.60.4.41m_hbox2	444
12.60.4.42m_button_undo	444
12.60.4.43m_button_redo	444
12.60.4.44m_button_quantize	444
12.60.4.45m_button_tools	444
12.60.4.46m_button_sequence	444
12.60.4.47m_entry_sequence	444
12.60.4.48m_button_bus	444
12.60.4.49m_entry_bus	444
12.60.4.50m_button_channel	444
12.60.4.51m_entry_channel	444
12.60.4.52m_button_snap	444
12.60.4.53m_entry_snap	444
12.60.4.54m_button_note_length	444

12.60.4.55m_entry_note_length	445
12.60.4.56m_button_zoom	445
12.60.4.57m_entry_zoom	445
12.60.4.58m_button_length	445
12.60.4.59m_entry_length	445
12.60.4.60m_button_key	445
12.60.4.61m_entry_key	445
12.60.4.62m_button_scale	445
12.60.4.63m_entry_scale	445
12.60.4.64m_tooltips	445
12.60.4.65m_button_data	445
12.60.4.66m_entry_data	445
12.60.4.67m_button_bpm	445
12.60.4.68m_entry_bpm	445
12.60.4.69m_button_bw	445
12.60.4.70m_entry_bw	445
12.60.4.71m_button_rec_vol	445
12.60.4.72m_toggle_play	445
12.60.4.73m_toggle_record	445
12.60.4.74m_toggle_q_rec	445
12.60.4.75m_toggle_thru	445
12.60.4.76m_entry_name	445
12.60.4.77m_editing_status	445
12.60.4.78m_editing_cc	446
12.60.4.79m_have_focus	446
12.61 seq64::sequevent Class Reference	446
12.61.1 Constructor & Destructor Documentation	449
12.61.1.1 sequevent(perform &p, sequence &seq, int zoom, int snap, seqdata &seqdata_wid, Gtk::Adjustment &hadjust, int ppqn=SEQ64_USE_DEFAULT_PPQN)	449
12.61.1.2 ~sequevent()	449
12.61.2 Member Function Documentation	449

12.61.2.1	<code>reset()</code>	449
12.61.2.2	<code>redraw()</code>	450
12.61.2.3	<code>set_zoom(int zoom)</code>	450
12.61.2.4	<code>set_snap(int snap)</code>	450
12.61.2.5	<code>set_data_type(midibyte status, midibyte control)</code>	450
12.61.2.6	<code>update_sizes()</code>	450
12.61.2.7	<code>draw_background()</code>	450
12.61.2.8	<code>draw_events_on_pixmap()</code>	450
12.61.2.9	<code>draw_pixmap_on_window()</code>	450
12.61.2.10	<code>draw_selection_on_window()</code>	451
12.61.2.11	<code>update_pixmap()</code>	451
12.61.2.12	<code>force_draw()</code>	451
12.61.2.13	<code>dle_redraw()</code>	451
12.61.2.14	<code>k_to_w(int x1, int x2, int &x, int &w)</code>	451
12.61.2.15	<code>drop_event(midipulse tick)</code>	451
12.61.2.16	<code>draw_events_on(Glib::RefPtr< Gdk::Drawable > draw)</code>	451
12.61.2.17	<code>start_paste()</code>	452
12.61.2.18	<code>change_horz()</code>	452
12.61.2.19	<code>convert_x(int x, midipulse &tick)</code>	452
12.61.2.20	<code>convert_t(midipulse tick, int &x)</code>	452
12.61.2.21	<code>snap_y(int &y)</code>	452
12.61.2.22	<code>snap_x(int &x)</code>	452
12.61.2.23	<code>on_realize()</code>	453
12.61.2.24	<code>on_expose_event(GdkEventExpose *ev)</code>	453
12.61.2.25	<code>on_button_press_event(GdkEventButton *ev)</code>	453
12.61.2.26	<code>on_button_release_event(GdkEventButton *ev)</code>	453
12.61.2.27	<code>on_motion_notify_event(GdkEventMotion *ev)</code>	454
12.61.2.28	<code>on_focus_in_event(GdkEventFocus *)</code>	454
12.61.2.29	<code>on_focus_out_event(GdkEventFocus *)</code>	454
12.61.2.30	<code>on_key_press_event(GdkEventKey *p0)</code>	454

12.61.2.31on_size_allocate(Gtk::Allocation &)	455
12.61.3 Friends And Related Function Documentation	455
12.61.3.1 FruitySeqEventInput	455
12.61.3.2 Seq24SeqEventInput	455
12.61.4 Field Documentation	455
12.61.4.1 m_fruity_interaction	455
12.61.4.2 m_seq24_interaction	455
12.61.4.3 m_seq	455
12.61.4.4 m_zoom	455
12.61.4.5 m_snap	455
12.61.4.6 m_ppqn	455
12.61.4.7 m_old	455
12.61.4.8 m_selected	455
12.61.4.9 m_scroll_offset_ticks	455
12.61.4.10m_scroll_offset_x	455
12.61.4.11m_seqdata_wid	456
12.61.4.12m_selecting	456
12.61.4.13m_moving_init	456
12.61.4.14m_moving	456
12.61.4.15m_growing	456
12.61.4.16m_painting	456
12.61.4.17m_paste	456
12.61.4.18m_move_snap_offset_x	456
12.61.4.19m_status	456
12.61.4.20m_cc	456
12.62seq64::seqkeys Class Reference	456
12.62.1 Constructor & Destructor Documentation	459
12.62.1.1 seqkeys(sequence &seq, perform &p, Gtk::Adjustment &vadjust)	459
12.62.1.2 ~seqkeys()	459
12.62.2 Member Function Documentation	459

12.62.2.1	<code>set_scale(int scale)</code>	459
12.62.2.2	<code>set_key(int key)</code>	460
12.62.2.3	<code>set_hint_key(int key)</code>	460
12.62.2.4	<code>set_hint_state(bool state)</code>	460
12.62.2.5	<code>force_draw()</code>	460
12.62.2.6	<code>draw_area()</code>	460
12.62.2.7	<code>update_pixmap()</code>	460
12.62.2.8	<code>convert_y(int y, int &note)</code>	460
12.62.2.9	<code>draw_key(int key, bool state)</code>	461
12.62.2.10	<code>change_vert()</code>	461
12.62.2.11	<code>update_sizes()</code>	461
12.62.2.12	<code>reset()</code>	461
12.62.2.13	<code>s_black_key(int key) const</code>	461
12.62.2.14	<code>on_realize()</code>	461
12.62.2.15	<code>on_expose_event(GdkEventExpose *ev)</code>	461
12.62.2.16	<code>on_button_press_event(GdkEventButton *ev)</code>	462
12.62.2.17	<code>on_button_release_event(GdkEventButton *ev)</code>	462
12.62.2.18	<code>on_motion_notify_event(GdkEventMotion *p0)</code>	462
12.62.2.19	<code>on_enter_notify_event(GdkEventCrossing *p0)</code>	462
12.62.2.20	<code>on_leave_notify_event(GdkEventCrossing *p0)</code>	463
12.62.2.21	<code>on_scroll_event(GdkEventScroll *ev)</code>	463
12.62.2.22	<code>on_size_allocate(Gtk::Allocation &)</code>	463
12.62.3	Field Documentation	463
12.62.3.1	<code>m_seq</code>	463
12.62.3.2	<code>m_scroll_offset_key</code>	463
12.62.3.3	<code>m_scroll_offset_y</code>	463
12.62.3.4	<code>m_hint_state</code>	463
12.62.3.5	<code>m_hint_key</code>	463
12.62.3.6	<code>m_keying</code>	463
12.62.3.7	<code>m_keying_note</code>	464

12.62.3.8 m_scale	464
12.62.3.9 m_key	464
12.62.3.10m_show_octave_letters	464
12.63seq64::seqmenu Class Reference	464
12.63.1 Detailed Description	467
12.63.2 Constructor & Destructor Documentation	467
12.63.2.1 seqmenu(perform &a_p)	467
12.63.2.2 ~seqmenu()	468
12.63.3 Member Function Documentation	468
12.63.3.1 current_seq() const	468
12.63.3.2 is_modified() const	468
12.63.3.3 current_seq(int seq)	468
12.63.3.4 set_edit_sequence(int seqnum)	468
12.63.3.5 unset_edit_sequence(int seqnum)	468
12.63.3.6 is_edit_sequence(int seqnum) const	468
12.63.3.7 is_modified(bool flag)	468
12.63.3.8 get_current_sequence() const	468
12.63.3.9 get_sequence(int seqnum) const	468
12.63.3.10s_current_seq_active() const	468
12.63.3.11is_current_seq_in_edit() const	468
12.63.3.12new_current_sequence()	468
12.63.3.13new_sequence(int seqnum)	468
12.63.3.14delete_current_sequence()	468
12.63.3.15toggle_current_sequence()	468
12.63.3.16popup_menu()	468
12.63.3.17seq_edit()	469
12.63.3.18seq_event_edit()	469
12.63.3.19seq_set_and_edit(int seqnum)	469
12.63.3.20seq_set_and_eventedit(int seqnum)	469
12.63.3.21seq_new()	469

12.63.3.22	seq_copy()	470
12.63.3.23	seq_cut()	470
12.63.3.24	seq_paste()	470
12.63.3.25	seq_clear_perf()	470
12.63.3.26	set_bus_and_midi_channel(int a_bus, int a_ch)	470
12.63.3.27	set_transposable(bool flag)	470
12.63.3.28	mute_all_tracks()	470
12.63.3.29	unmute_all_tracks()	470
12.63.3.30	toggle_all_tracks()	471
12.63.3.31	redraw(int a_sequence)=0	471
12.63.3.32	on_realize()	471
12.63.4	Field Documentation	471
12.63.4.1	m_menu	471
12.63.4.2	m_mainperf	471
12.63.4.3	m_clipboard	471
12.63.4.4	m_seqedit	471
12.63.4.5	m_eventedit	471
12.63.4.6	m_current_seq	471
12.63.4.7	m_modified	471
12.64	seq64::seqroll Class Reference	471
12.64.1	Constructor & Destructor Documentation	477
12.64.1.1	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)	477
12.64.1.2	~seqroll()	478
12.64.2	Member Function Documentation	478
12.64.2.1	set_snap(int snap)	478
12.64.2.2	set_zoom(int zoom)	478
12.64.2.3	set_note_length(int note_length)	478
12.64.2.4	note_off_length() const	478
12.64.2.5	add_note(midipulse tick, int note, bool paint=true)	478

12.64.2.6 set_key(int key)	478
12.64.2.7 set_scale(int scale)	478
12.64.2.8 set_data_type(midibyte status, midibyte control)	479
12.64.2.9 set_background_sequence(bool state, int seq)	479
12.64.2.10 update_pixmap()	479
12.64.2.11 update_sizes()	479
12.64.2.12 update_background()	479
12.64.2.13 draw_background_on_pixmap()	479
12.64.2.14 draw_events_on_pixmap()	479
12.64.2.15 draw_selection_on_window()	480
12.64.2.16 draw_progress_on_window()	480
12.64.2.17 reset()	480
12.64.2.18 update_and_draw(int force=false)	480
12.64.2.19 redraw()	480
12.64.2.20 redraw_events()	480
12.64.2.21 start_paste()	481
12.64.2.22 complete_paste()	481
12.64.2.23 complete_paste(int x, int y)	481
12.64.2.24 follow_progress()	481
12.64.2.25 force_draw()	481
12.64.2.26 horizontal_adjust(double step)	481
12.64.2.27 vertical_adjust(double step)	481
12.64.2.28 snap_y(int &y)	481
12.64.2.29 snap_x(int &x)	481
12.64.2.30 convert_xy(int x, int y, midipulse &ticks, int ¬e)	482
12.64.2.31 convert_tn(midipulse ticks, int note, int &x, int &y)	482
12.64.2.32 xy_to_rect(int x1, int y1, int x2, int y2, int &x, int &y, int &w, int &h)	482
12.64.2.33 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)	482
12.64.2.34 convert_sel_box_to_rect(midipulse tick_s, midipulse tick_f, int note_h, int note_l)	483
12.64.2.35 get_selected_box(midipulse &tick_s, int ¬e_h, midipulse &tick_f, int ¬e_l)	483

12.64.2.36	<code>draw_events_on(Glib::RefPtr< Gdk::Drawable > draw)</code>	483
12.64.2.37	<code>dle_redraw()</code>	483
12.64.2.38	<code>dle_progress()</code>	483
12.64.2.39	<code>change_horz()</code>	483
12.64.2.40	<code>change_vert()</code>	484
12.64.2.41	<code>move_selection_box(int dx, int dy)</code>	484
12.64.2.42	<code>move_selected_notes(int dx, int dy)</code>	484
12.64.2.43	<code>grow_selected_notes(int dx)</code>	484
12.64.2.44	<code>set_adding(bool adding)</code>	485
12.64.2.45	<code>update_mouse_pointer(bool adding=false)</code>	485
12.64.2.46	<code>button_press_initial(GdkEventButton *ev, int &norm_x, int &snapped_x, int &snapped_y)</code>	485
12.64.2.47	<code>align_selection(midipulse &tick_s, int &note_h, midipulse &tick_f, int &note_l, int snapped_x)</code>	485
12.64.2.48	<code>button_press(GdkEventButton *ev)</code>	485
12.64.2.49	<code>button_release(GdkEventButton *ev)</code>	485
12.64.2.50	<code>motion_notify(GdkEventMotion *ev)</code>	486
12.64.2.51	<code>clear_selected()</code>	486
12.64.2.52	<code>clear_old()</code>	486
12.64.2.53	<code>clear_flags()</code>	486
12.64.2.54	<code>scroll_offset_x(int x) const</code>	486
12.64.2.55	<code>scroll_offset_y(int y) const</code>	486
12.64.2.56	<code>set_current_offset_x_y(int x, int y)</code>	487
12.64.2.57	<code>adding() const</code>	487
12.64.2.58	<code>selecting() const</code>	487
12.64.2.59	<code>growing() const</code>	487
12.64.2.60	<code>normal_action() const</code>	487
12.64.2.61	<code>select_action() const</code>	487
12.64.2.62	<code>drop_action() const</code>	488
12.64.2.63	<code>on_realize()</code>	488
12.64.2.64	<code>on_expose_event(GdkEventExpose *ev)</code>	488

12.64.2.65	<code>on_button_press_event(GdkEventButton *ev)</code>	488
12.64.2.66	<code>on_button_release_event(GdkEventButton *ev)</code>	488
12.64.2.67	<code>on_motion_notify_event(GdkEventMotion *ev)</code>	488
12.64.2.68	<code>on_focus_in_event(GdkEventFocus *)</code>	489
12.64.2.69	<code>on_focus_out_event(GdkEventFocus *)</code>	489
12.64.2.70	<code>on_key_press_event(GdkEventKey *ev)</code>	489
12.64.2.71	<code>on_scroll_event(GdkEventScroll *a_ev)</code>	490
12.64.2.72	<code>on_size_allocate(Gtk::Allocation &)</code>	490
12.64.2.73	<code>on_leave_notify_event(GdkEventCrossing *p0)</code>	490
12.64.2.74	<code>on_enter_notify_event(GdkEventCrossing *p0)</code>	490
12.64.3	Friends And Related Function Documentation	490
12.64.3.1	<code>FruitySeqRollInput</code>	490
12.64.4	Field Documentation	491
12.64.4.1	<code>m_horizontal_adjust</code>	491
12.64.4.2	<code>m_vertical_adjust</code>	491
12.64.4.3	<code>m_old</code>	491
12.64.4.4	<code>m_selected</code>	491
12.64.4.5	<code>m_seq</code>	491
12.64.4.6	<code>m_seqkeys_wid</code>	491
12.64.4.7	<code>m_fruity_interaction</code>	491
12.64.4.8	<code>m_pos</code>	491
12.64.4.9	<code>m_zoom</code>	491
12.64.4.10	<code>m_snap</code>	491
12.64.4.11	<code>m_ppqn</code>	491
12.64.4.12	<code>m_note_length</code>	491
12.64.4.13	<code>m_scale</code>	491
12.64.4.14	<code>m_key</code>	491
12.64.4.15	<code>m_adding</code>	491
12.64.4.16	<code>m_selecting</code>	492
12.64.4.17	<code>m_moving</code>	492

12.64.4.18m_moving_init	492
12.64.4.19m_growing	492
12.64.4.20m_painting	492
12.64.4.21m_paste	492
12.64.4.22m_is_drag_pasting	492
12.64.4.23m_is_drag_pasting_start	492
12.64.4.24m_justselected_one	492
12.64.4.25m_move_delta_x	492
12.64.4.26m_move_delta_y	492
12.64.4.27m_move_snap_offset_x	492
12.64.4.28m_progress_x	492
12.64.4.29m_scroll_offset_ticks	492
12.64.4.30m_scroll_offset_key	492
12.64.4.31m_scroll_offset_x	492
12.64.4.32m_scroll_offset_y	492
12.64.4.33m_background_sequence	492
12.64.4.34m_drawing_background_seq	492
12.64.4.35m_status	492
12.64.4.36m_cc	493
12.65seq64::seqtime Class Reference	493
12.65.1 Constructor & Destructor Documentation	495
12.65.1.1 seqtime(sequence &seq, perform &p, int zoom, Gtk::Adjustment &hadjust, int ppqn=SEQ64_USE_DEFAULT_PPQN)	495
12.65.1.2 ~seqtime()	495
12.65.2 Member Function Documentation	495
12.65.2.1 reset()	495
12.65.2.2 redraw()	495
12.65.2.3 set_zoom(int zoom)	495
12.65.2.4 draw_pixmap_on_window()	495
12.65.2.5 draw_progress_on_window()	495
12.65.2.6 update_pixmap()	495

12.65.2.7 change_horz()	496
12.65.2.8 update_sizes()	496
12.65.2.9 idle_progress()	496
12.65.2.10 on_realize()	496
12.65.2.11 on_expose_event(GdkEventExpose *a_ev)	496
12.65.2.12 on_size_allocate(Gtk::Allocation &)	496
12.65.2.13 on_button_press_event(GdkEventButton *)	496
12.65.2.14 on_button_release_event(GdkEventButton *)	496
12.65.3 Field Documentation	496
12.65.3.1 m_seq	496
12.65.3.2 m_scroll_offset_ticks	496
12.65.3.3 m_scroll_offset_x	496
12.65.3.4 m_zoom	496
12.65.3.5 m_ppqn	496
12.66 seq64::sequence Class Reference	496
12.66.1 Detailed Description	506
12.66.2 Member Typedef Documentation	506
12.66.2.1 EventStack	506
12.66.3 Member Enumeration Documentation	506
12.66.3.1 select_action_e	506
12.66.4 Constructor & Destructor Documentation	506
12.66.4.1 sequence(int ppqn=SEQ64_USE_DEFAULT_PPQN)	506
12.66.4.2 ~sequence()	506
12.66.5 Member Function Documentation	506
12.66.5.1 operator=(const sequence &rhs)	506
12.66.5.2 partial_assign(const sequence &rhs)	506
12.66.5.3 events()	507
12.66.5.4 events() const	507
12.66.5.5 any_selected_notes() const	507
12.66.5.6 triggerlist()	507

12.66.5.7 number() const	507
12.66.5.8 number(int seqnum)	507
12.66.5.9 event_count() const	507
12.66.5.10 push_undo()	507
12.66.5.11 pop_undo()	507
12.66.5.12 pop_redo()	507
12.66.5.13 push_trigger_undo()	507
12.66.5.14 pop_trigger_undo()	508
12.66.5.15 set_name(const std::string &name)	508
12.66.5.16 set_name(char *name)	508
12.66.5.17 set_measures(int lengthmeasures)	508
12.66.5.18 get_measures()	508
12.66.5.19 get_ppqn() const	508
12.66.5.20 set_beats_per_bar(int beatspermeasure)	508
12.66.5.21 get_beats_per_bar() const	508
12.66.5.22 set_beat_width(int beatwidth)	508
12.66.5.23 get_beat_width() const	508
12.66.5.24 clocks_per_metronome(int cpm)	508
12.66.5.25 clocks_per_metronome() const	508
12.66.5.26 set_32nds_per_quarter(int tpq)	508
12.66.5.27 get_32nds_per_quarter() const	508
12.66.5.28 us_per_quarter_note(int upqn)	508
12.66.5.29 us_per_quarter_note() const	509
12.66.5.30 set_rec_vol(int rec_vol)	509
12.66.5.31 set_song_mute(bool mute)	509
12.66.5.32 toggle_song_mute()	509
12.66.5.33 get_song_mute() const	509
12.66.5.34 get_name() const	509
12.66.5.35 name() const	509
12.66.5.36 set_editing(bool edit)	509

12.66.5.37	<code>get_editing()</code> const	509
12.66.5.38	<code>set_raise(bool edit)</code>	509
12.66.5.39	<code>get_raise(void)</code> const	509
12.66.5.40	<code>set_length(midipulse len, bool adjust_triggers=true)</code>	509
12.66.5.41	<code>get_length()</code> const	510
12.66.5.42	<code>get_last_tick()</code>	510
12.66.5.43	<code>set_last_tick(midipulse tick)</code>	510
12.66.5.44	<code>mod_last_tick()</code>	510
12.66.5.45	<code>set_playing(bool)</code>	510
12.66.5.46	<code>get_playing()</code> const	510
12.66.5.47	<code>toggle_playing()</code>	510
12.66.5.48	<code>toggle_queued()</code>	510
12.66.5.49	<code>off_queued()</code>	510
12.66.5.50	<code>on_queued()</code>	511
12.66.5.51	<code>get_queued()</code> const	511
12.66.5.52	<code>get_queued_tick()</code> const	511
12.66.5.53	<code>check_queued_tick(midipulse tick)</code> const	511
12.66.5.54	<code>set_recording(bool)</code>	511
12.66.5.55	<code>get_recording()</code> const	511
12.66.5.56	<code>set_snap_tick(int st)</code>	511
12.66.5.57	<code>set_quantized_rec(bool qr)</code>	511
12.66.5.58	<code>get_quantized_rec()</code> const	511
12.66.5.59	<code>set_thru(bool)</code>	511
12.66.5.60	<code>get_thru()</code> const	511
12.66.5.61	<code>is_dirty_main()</code>	511
12.66.5.62	<code>s_dirty_edit()</code>	512
12.66.5.63	<code>s_dirty_perf()</code>	512
12.66.5.64	<code>s_dirty_names()</code>	512
12.66.5.65	<code>set_dirty_mp()</code>	512
12.66.5.66	<code>set_dirty()</code>	512

12.66.5.67	<code>get_midi_channel()</code> const	512
12.66.5.68	<code>smf_0()</code> const	512
12.66.5.69	<code>set_midi_channel(midibyte ch)</code>	512
12.66.5.70	<code>print()</code> const	513
12.66.5.71	<code>print_triggers()</code> const	513
12.66.5.72	<code>play(midipulse tick, bool playback_mode)</code>	513
12.66.5.73	<code>add_event(const event &er)</code>	513
12.66.5.74	<code>add_trigger(midipulse tick, midipulse len, midipulse offset=0, bool adjust_offset=true)</code>	514
12.66.5.75	<code>split_trigger(midipulse tick)</code>	514
12.66.5.76	<code>grow_trigger(midipulse tick_from, midipulse tick_to, midipulse len)</code>	514
12.66.5.77	<code>del_trigger(midipulse tick)</code>	515
12.66.5.78	<code>get_trigger_state(midipulse tick)</code>	515
12.66.5.79	<code>select_trigger(midipulse tick)</code>	515
12.66.5.80	<code>unselect_triggers()</code>	515
12.66.5.81	<code>intersect_triggers(midipulse position, midipulse &start, midipulse &ender)</code>	515
12.66.5.82	<code>intersect_notes(midipulse position, midipulse position_note, midipulse &start, midipulse &ender, int &note)</code>	516
12.66.5.83	<code>intersect_events(midipulse posstart, midipulse posend, midibyte status, midipulse &start)</code>	516
12.66.5.84	<code>del_selected_trigger()</code>	517
12.66.5.85	<code>cut_selected_trigger()</code>	517
12.66.5.86	<code>copy_selected_trigger()</code>	517
12.66.5.87	<code>paste_trigger()</code>	517
12.66.5.88	<code>move_selected_triggers_to(midipulse tick, bool adjust_offset, int which=2)</code>	517
12.66.5.89	<code>selected_trigger_start()</code>	517
12.66.5.90	<code>selected_trigger_end()</code>	518
12.66.5.91	<code>get_max_trigger()</code>	518
12.66.5.92	<code>move_triggers(midipulse start_tick, midipulse distance, bool direction)</code>	518
12.66.5.93	<code>copy_triggers(midipulse start_tick, midipulse distance)</code>	518
12.66.5.94	<code>clear_triggers()</code>	518
12.66.5.95	<code>get_trigger_offset()</code> const	518

12.66.5.96	set_midi_bus(char mb)	518
12.66.5.97	get_midi_bus() const	519
12.66.5.98	set_master_midi_bus(mastermidibus *mmb)	519
12.66.5.99	select_note_events(midipulse tick_s, int note_h, midipulse tick_f, int note_l, select_action_e action)	519
12.66.5.100	select_events(midipulse tick_s, midipulse tick_f, midibyte status, midibyte cc, select_action_e action)	519
12.66.5.101	select_events(midibyte status, midibyte cc, bool inverse=false)	520
12.66.5.102	select_all_notes(bool inverse=false)	520
12.66.5.103	get_num_selected_notes() const	520
12.66.5.104	get_num_selected_events(midibyte status, midibyte cc) const	520
12.66.5.105	select_all()	521
12.66.5.106	copy_selected()	521
12.66.5.107	cut_selected(bool copyevents=true)	521
12.66.5.108	paste_selected(midipulse tick, int note)	521
12.66.5.109	get_selected_box(midipulse &tick_s, int ¬e_h, midipulse &tick_f, int ¬e_l)	522
12.66.5.110	get_clipboard_box(midipulse &tick_s, int ¬e_h, midipulse &tick_f, int ¬e_l)	522
12.66.5.111	adjust_timestamp(midipulse t, bool isnoteoff=false)	523
12.66.5.112	clip_timestamp(midipulse ontime, midipulse offtime)	523
12.66.5.113	move_selected_notes(midipulse deltatick, int deltanote)	523
12.66.5.114	add_note(midipulse tick, midipulse len, int note, bool paint=false)	524
12.66.5.115	add_event(midipulse tick, midibyte status, midibyte d0, midibyte d1, bool paint=false)	524
12.66.5.116	stream_event(event &ev)	525
12.66.5.117	change_event_data_range(midipulse tick_s, midipulse tick_f, midibyte status, midibyte cc, int d_s, int d_f)	525
12.66.5.118	increment_selected(midibyte status, midibyte)	526
12.66.5.119	decrement_selected(midibyte status, midibyte)	526
12.66.5.120	grow_selected(midipulse deltatick)	527
12.66.5.121	stretch_selected(midipulse deltatick)	527
12.66.5.122	move_marked()	528
12.66.5.123	mark_selected()	528

12.66.5.124	remove_selected()	528
12.66.5.125	repaint_all()	528
12.66.5.126	unselect()	528
12.66.5.127	verify_and_link()	528
12.66.5.128	link_new()	528
12.66.5.129	zero_markers()	528
12.66.5.130	play_note_on(int note)	528
12.66.5.131	play_note_off(int note)	529
12.66.5.132	stop_playing_notes()	529
12.66.5.133	pause()	529
12.66.5.134	reset(bool live_mode)	529
12.66.5.135	reset_draw_marker()	529
12.66.5.136	reset_draw_trigger_marker()	529
12.66.5.137	get_next_note_event(midipulse *tick_s, midipulse *tick_f, int *note, bool *selected, int *velocity)	530
12.66.5.138	get_minmax_note_events(int &lowest, int &highest)	530
12.66.5.139	get_next_event(midibyte status, midibyte cc, midipulse *tick, midibyte *d0, midibyte *d1, bool *selected)	530
12.66.5.140	get_next_event(midibyte *status, midibyte *cc)	531
12.66.5.141	get_next_trigger(midipulse *tick_on, midipulse *tick_off, bool *selected, midipulse *tick_offset)	531
12.66.5.142	midi_container(midi_container &c, int tracknumber)	531
12.66.5.143	quantize_events(midibyte status, midibyte cc, midipulse snap_tick, int divide, bool linked=false)	531
12.66.5.144	flush_quantize(midibyte status, midibyte cc, midipulse snap_tick, int divide, bool linked=false)	531
12.66.5.145	transpose_notes(int steps, int scale)	531
12.66.5.146	musical_key() const	532
12.66.5.147	musical_key(int key)	532
12.66.5.148	musical_scale() const	532
12.66.5.149	musical_scale(int scale)	532
12.66.5.150	background_sequence() const	532
12.66.5.151	background_sequence(int bs)	532

12.66.5.152	show_events() const	532
12.66.5.153	copy_events(const event_list &newevents)	532
12.66.5.154	note_off_margin() const	532
12.66.5.155	set_parent(perform *p)	532
12.66.5.156	put_event_on_bus(event &ev)	533
12.66.5.157	set_trigger_offset(midipulse trigger_offset)	533
12.66.5.158	split_trigger(trigger &trig, midipulse splittick)	533
12.66.5.159	adjust_trigger_offsets_to_length(midipulse newlen)	533
12.66.5.160	adjust_offset(midipulse offset)	534
12.66.5.161	remove(event_list::iterator i)	534
12.66.5.162	remove(event &e)	534
12.66.5.163	remove_all()	534
12.66.6	Friends And Related Function Documentation	534
12.66.6.1	perform	534
12.66.6.2	triggers	534
12.66.7	Field Documentation	534
12.66.7.1	m_events_clipboard	534
12.66.7.2	m_parent	534
12.66.7.3	m_events	535
12.66.7.4	m_triggers	535
12.66.7.5	m_events_undo	535
12.66.7.6	m_events_redo	535
12.66.7.7	m_iterator_draw	535
12.66.7.8	m_midi_channel	535
12.66.7.9	m_bus	535
12.66.7.10	m_song_mute	535
12.66.7.11	m_notes_on	535
12.66.7.12	m_masterbus	535
12.66.7.13	m_playing_notes	535
12.66.7.14	m_was_playing	535

12.66.7.15m_playing	535
12.66.7.16m_recording	535
12.66.7.17m_quantized_rec	535
12.66.7.18m_thru	535
12.66.7.19m_queued	535
12.66.7.20m_dirty_main	535
12.66.7.21m_dirty_edit	536
12.66.7.22m_dirty_perf	536
12.66.7.23m_dirty_names	536
12.66.7.24m_editing	536
12.66.7.25m_raise	536
12.66.7.26m_name	536
12.66.7.27m_last_tick	536
12.66.7.28m_queued_tick	536
12.66.7.29m_trigger_offset	536
12.66.7.30m_maxbeats	536
12.66.7.31m_ppqn	536
12.66.7.32m_seq_number	536
12.66.7.33m_length	536
12.66.7.34m_snap_tick	536
12.66.7.35m_time_beats_per_measure	536
12.66.7.36m_time_beat_width	537
12.66.7.37m_clocks_per_metronome	537
12.66.7.38m_32nds_per_quarter	537
12.66.7.39m_us_per_quarter_note	537
12.66.7.40m_rec_vol	537
12.66.7.41m_musical_key	537
12.66.7.42m_musical_scale	537
12.66.7.43m_background_sequence	537
12.66.7.44m_mutex	537

12.66.7.45m_note_off_margin	537
12.67seq64::trigger Class Reference	538
12.67.1 Detailed Description	539
12.67.2 Constructor & Destructor Documentation	539
12.67.2.1 trigger()	539
12.67.3 Member Function Documentation	539
12.67.3.1 operator<(const trigger &rhs)	539
12.67.3.2 tick_start() const	539
12.67.3.3 tick_start(midipulse s)	539
12.67.3.4 increment_tick_start(midipulse s)	539
12.67.3.5 decrement_tick_start(midipulse s)	539
12.67.3.6 tick_end() const	539
12.67.3.7 tick_end(midipulse e)	539
12.67.3.8 increment_tick_end(midipulse s)	539
12.67.3.9 decrement_tick_end(midipulse s)	539
12.67.3.10offset() const	539
12.67.3.11offset(midipulse o)	539
12.67.3.12increment_offset(midipulse s)	539
12.67.3.13decrement_offset(midipulse s)	539
12.67.3.14selected() const	540
12.67.3.15selected(bool s)	540
12.67.4 Field Documentation	540
12.67.4.1 m_tick_start	540
12.67.4.2 m_tick_end	540
12.67.4.3 m_offset	540
12.67.4.4 m_selected	540
12.68seq64::triggers Class Reference	540
12.68.1 Member Typedef Documentation	542
12.68.1.1 List	542
12.68.1.2 Stack	542

12.68.2 Constructor & Destructor Documentation	542
12.68.2.1 triggers(sequence &parent)	542
12.68.2.2 ~triggers()	542
12.68.3 Member Function Documentation	542
12.68.3.1 operator=(const triggers &rhs)	543
12.68.3.2 set_ppqn(int ppqn)	543
12.68.3.3 set_length(int len)	543
12.68.3.4 triggerlist()	543
12.68.3.5 push_undo()	543
12.68.3.6 pop_undo()	543
12.68.3.7 print(const std::string &seqname) const	543
12.68.3.8 play(midipulse &starttick, midipulse &endtick)	543
12.68.3.9 add(midipulse tick, midipulse len, midipulse offset=0, bool adjustoffset=true)	544
12.68.3.10adjust_offsets_to_length(midipulse newlen)	544
12.68.3.11split(midipulse tick)	544
12.68.3.12split(trigger &trig, midipulse splittick)	545
12.68.3.13grow(midipulse tickfrom, midipulse tickto, midipulse length)	545
12.68.3.14remove(midipulse tick)	545
12.68.3.15get_state(midipulse tick)	545
12.68.3.16select(midipulse tick)	546
12.68.3.17unselect()	546
12.68.3.18intersect(midipulse position, midipulse &start, midipulse &end)	546
12.68.3.19remove_selected()	546
12.68.3.20copy_selected()	546
12.68.3.21paste()	546
12.68.3.22move_selected(midipulse tick, bool adjustoffset, int which=2)	547
12.68.3.23get_selected_start()	547
12.68.3.24get_selected_end()	547
12.68.3.25get_maximum()	547
12.68.3.26move(midipulse starttick, midipulse distance, bool direction)	547

12.68.3.27	copy(midipulse starttick, midipulse distance)	548
12.68.3.28	clear()	548
12.68.3.29	next(midipulse *tick_on, midipulse *tick_off, bool *selected, midipulse *tick_offset)	548
12.68.3.30	next_trigger()	549
12.68.3.31	reset_draw_trigger_marker()	549
12.68.3.32	adjust_offset(midipulse offset)	549
12.68.4	Field Documentation	549
12.68.4.1	m_parent	549
12.68.4.2	m_triggers	549
12.68.4.3	m_clipboard	549
12.68.4.4	m_undo_stack	549
12.68.4.5	m_redo_stack	549
12.68.4.6	m_iterator_play_trigger	549
12.68.4.7	m_iterator_draw_trigger	549
12.68.4.8	m_trigger_copied	549
12.68.4.9	m_ppqn	549
12.68.4.10	m_length	550
12.69	seq64::user_instrument Class Reference	550
12.69.1	Detailed Description	551
12.69.2	Constructor & Destructor Documentation	551
12.69.2.1	user_instrument(const std::string &name="")	551
12.69.2.2	user_instrument(const user_instrument &rhs)	551
12.69.3	Member Function Documentation	551
12.69.3.1	operator=(const user_instrument &rhs)	551
12.69.3.2	is_valid() const	551
12.69.3.3	set_defaults()	551
12.69.3.4	name() const	552
12.69.3.5	controller_count() const	552
12.69.3.6	controller_max() const	552
12.69.3.7	controller_name(int c) const	552

12.69.3.8 controller_active(int c) const	552
12.69.3.9 set_controller(int c, const std::string &cname, bool isactive)	552
12.69.3.10set_name(const std::string &instname)	552
12.69.3.11copy_definitions(const user_instrument &rhs)	553
12.69.4 Field Documentation	553
12.69.4.1 m_is_valid	553
12.69.4.2 m_controller_count	553
12.69.4.3 m_instrument_def	553
12.70seq64::user_instrument_t Struct Reference	553
12.70.1 Field Documentation	553
12.70.1.1 instrument	553
12.70.1.2 controllers	554
12.70.1.3 controllers_active	554
12.71seq64::user_midi_bus Class Reference	554
12.71.1 Detailed Description	555
12.71.2 Constructor & Destructor Documentation	555
12.71.2.1 user_midi_bus(const std::string &name="")	555
12.71.2.2 user_midi_bus(const user_midi_bus &rhs)	555
12.71.3 Member Function Documentation	555
12.71.3.1 operator=(const user_midi_bus &rhs)	555
12.71.3.2 is_valid() const	555
12.71.3.3 set_defaults()	555
12.71.3.4 name() const	556
12.71.3.5 channel_count() const	556
12.71.3.6 channel_max() const	556
12.71.3.7 instrument(int channel) const	556
12.71.3.8 set_instrument(int channel, int instrum)	556
12.71.3.9 set_name(const std::string &name)	556
12.71.3.10copy_definitions(const user_midi_bus &rhs)	556
12.71.4 Field Documentation	556

12.71.4.1 m_is_valid	557
12.71.4.2 m_channel_count	557
12.71.4.3 m_midi_bus_def	557
12.72seq64::user_midi_bus_t Struct Reference	557
12.72.1 Field Documentation	557
12.72.1.1 alias	557
12.72.1.2 instrument	557
12.73seq64::user_settings Class Reference	557
12.73.1 Detailed Description	564
12.73.2 Member Typedef Documentation	564
12.73.2.1 Busses	564
12.73.2.2 Bussliterator	565
12.73.2.3 BussConstliterator	565
12.73.2.4 Instruments	565
12.73.2.5 Instrumentliterator	565
12.73.2.6 InstrumentConstliterator	565
12.73.3 Member Enumeration Documentation	565
12.73.3.1 mainwid_grid_style_t	565
12.73.4 Constructor & Destructor Documentation	565
12.73.4.1 user_settings()	565
12.73.4.2 user_settings(const user_settings &rhs)	565
12.73.5 Member Function Documentation	565
12.73.5.1 operator=(const user_settings &rhs)	565
12.73.5.2 set_defaults()	565
12.73.5.3 normalize()	566
12.73.5.4 add_bus(const std::string &alias)	566
12.73.5.5 add_instrument(const std::string &instname)	566
12.73.5.6 bus(int index)	566
12.73.5.7 instrument(int index)	566
12.73.5.8 bus_count() const	566

12.73.5.9	<code>set_bus_instrument(int index, int channel, int instrum)</code>	566
12.73.5.10	<code>bus_instrument(int buss, int channel)</code>	566
12.73.5.11	<code>bus_name(int buss)</code>	566
12.73.5.12	<code>instrument_count() const</code>	566
12.73.5.13	<code>set_instrument_controllers(int index, int cc, const std::string &ccname, bool isactive)</code>	566
12.73.5.14	<code>instrument_name(int instrum)</code>	566
12.73.5.15	<code>instrument_name(int buss, int channel)</code>	566
12.73.5.16	<code>instrument_controller_active(int instrum, int cc)</code>	566
12.73.5.17	<code>controller_active(int buss, int channel, int cc)</code>	566
12.73.5.18	<code>instrument_controller_name(int instrum, int cc)</code>	566
12.73.5.19	<code>controller_name(int buss, int channel, int cc)</code>	566
12.73.5.20	<code>grid_style() const</code>	567
12.73.5.21	<code>grid_is_normal() const</code>	567
12.73.5.22	<code>grid_is_white() const</code>	567
12.73.5.23	<code>grid_is_black() const</code>	567
12.73.5.24	<code>grid_brackets() const</code>	567
12.73.5.25	<code>mainwnd_rows() const</code>	567
12.73.5.26	<code>mainwnd_cols() const</code>	567
12.73.5.27	<code>seqs_in_set() const</code>	567
12.73.5.28	<code>mute_tracks() const</code>	567
12.73.5.29	<code>max_sets() const</code>	567
12.73.5.30	<code>max_sequence() const</code>	567
12.73.5.31	<code>text_x() const</code>	567
12.73.5.32	<code>text_y() const</code>	567
12.73.5.33	<code>seqchars_x() const</code>	567
12.73.5.34	<code>seqchars_y() const</code>	567
12.73.5.35	<code>seqarea_x() const</code>	567
12.73.5.36	<code>seqarea_y() const</code>	567
12.73.5.37	<code>seqarea_seq_x() const</code>	567
12.73.5.38	<code>seqarea_seq_y() const</code>	567

12.73.5.39	<code>mainwid_border()</code> const	567
12.73.5.40	<code>mainwid_spacing()</code> const	567
12.73.5.41	<code>mainwid_x()</code> const	567
12.73.5.42	<code>mainwid_y()</code> const	567
12.73.5.43	<code>control_height()</code> const	567
12.73.5.44	<code>zoom()</code> const	567
12.73.5.45	<code>zoom(int value)</code>	567
12.73.5.46	<code>global_seq_feature()</code> const	568
12.73.5.47	<code>global_seq_feature(bool flag)</code>	568
12.73.5.48	<code>seqedit_scale()</code> const	568
12.73.5.49	<code>seqedit_scale(int scale)</code>	568
12.73.5.50	<code>seqedit_key()</code> const	568
12.73.5.51	<code>seqedit_key(int key)</code>	568
12.73.5.52	<code>seqedit_bgsequence()</code> const	568
12.73.5.53	<code>seqedit_bgsequence(int seqnum)</code>	568
12.73.5.54	<code>use_new_font()</code> const	568
12.73.5.55	<code>allow_two_perfedits()</code> const	568
12.73.5.56	<code>perf_h_page_increment()</code> const	568
12.73.5.57	<code>perf_v_page_increment()</code> const	568
12.73.5.58	<code>progress_bar_colored()</code> const	568
12.73.5.59	<code>progress_bar_thick()</code> const	568
12.73.5.60	<code>window_redraw_rate()</code> const	568
12.73.5.61	<code>save_user_config()</code> const	568
12.73.5.62	<code>save_user_config(bool flag)</code>	568
12.73.5.63	<code>grid_brackets(int thickness)</code>	568
12.73.5.64	<code>grid_style(int gridstyle)</code>	568
12.73.5.65	<code>mainwnd_rows(int value)</code>	568
12.73.5.66	<code>mainwnd_cols(int value)</code>	568
12.73.5.67	<code>max_sets(int value)</code>	569
12.73.5.68	<code>ext_x(int value)</code>	569

12.73.5.69	text_y(int value)	569
12.73.5.70	seqchars_x(int value)	569
12.73.5.71	seqchars_y(int value)	569
12.73.5.72	seqarea_x(int value)	569
12.73.5.73	seqarea_y(int value)	569
12.73.5.74	seqarea_seq_x(int value)	569
12.73.5.75	seqarea_seq_y(int value)	569
12.73.5.76	mainwid_border(int value)	569
12.73.5.77	mainwid_spacing(int value)	569
12.73.5.78	control_height(int value)	569
12.73.5.79	dump_summary()	569
12.73.5.80	midi_ppqn() const	570
12.73.5.81	midi_beats_per_bar() const	570
12.73.5.82	midi_beats_per_minute() const	570
12.73.5.83	midi_beat_width() const	570
12.73.5.84	midi_buss_override() const	570
12.73.5.85	min_zoom() const	570
12.73.5.86	max_zoom() const	570
12.73.5.87	baseline_ppqn() const	570
12.73.5.88	use_new_font(bool flag)	570
12.73.5.89	allow_two_perfedits(bool flag)	570
12.73.5.90	perf_h_page_increment(int inc)	570
12.73.5.91	perf_v_page_increment(int inc)	570
12.73.5.92	progress_bar_colored(bool flag)	570
12.73.5.93	progress_bar_thick(bool flag)	570
12.73.5.94	window_redraw_rate(int ms)	570
12.73.5.95	midi_ppqn(int ppqn)	570
12.73.5.96	midi_buss_override(char buss)	570
12.73.5.97	midi_beats_per_bar(int beatsperbar)	571
12.73.5.98	midi_beats_per_minute(int beatsperminute)	571

12.73.5.99midi_beat_width(int beatwidth)	571
12.73.5.100private_bus(int buss)	571
12.73.5.101private_instrument(int instrum)	571
12.73.6 Friends And Related Function Documentation	571
12.73.6.1 userfile	571
12.73.7 Field Documentation	571
12.73.7.1 m_midi_buses	571
12.73.7.2 m_instruments	571
12.73.7.3 m_grid_style	572
12.73.7.4 m_grid_brackets	572
12.73.7.5 m_mainwnd_rows	572
12.73.7.6 m_mainwnd_cols	572
12.73.7.7 m_max_sets	572
12.73.7.8 m_mainwid_border	572
12.73.7.9 m_mainwid_spacing	573
12.73.7.10m_control_height	573
12.73.7.11m_current_zoom	573
12.73.7.12m_global_seq_feature_save	573
12.73.7.13m_seqedit_scale	573
12.73.7.14m_seqedit_key	573
12.73.7.15m_seqedit_bgsequence	573
12.73.7.16m_use_new_font	574
12.73.7.17m_allow_two_perfedits	574
12.73.7.18m_h_perf_page_increment	574
12.73.7.19m_v_perf_page_increment	574
12.73.7.20m_progress_bar_colored	574
12.73.7.21m_progress_bar_thick	574
12.73.7.22m_window_redraw_rate_ms	574
12.73.7.23m_text_x	574
12.73.7.24m_text_y	575

12.73.7.25m_seqchars_x	575
12.73.7.26m_seqchars_y	575
12.73.7.27m_midi_ppqn	575
12.73.7.28m_midi_beats_per_measure	575
12.73.7.29m_midi_beats_per_minute	575
12.73.7.30m_midi_beat_width	575
12.73.7.31m_midi_buss_override	575
12.73.7.32m_total_seqs	576
12.73.7.33m_seqs_in_set	576
12.73.7.34m_gmute_tracks	576
12.73.7.35m_max_sequence	576
12.73.7.36m_seqarea_x	576
12.73.7.37m_seqarea_y	576
12.73.7.38m_seqarea_seq_x	576
12.73.7.39m_seqarea_seq_y	576
12.73.7.40m_mainwid_x	576
12.73.7.41m_mainwid_y	577
12.73.7.42m_save_user_config	577
12.73.7.43mc_min_zoom	577
12.73.7.44mc_max_zoom	577
12.73.7.45mc_baseline_ppqn	577
12.74seq64::userfile Class Reference	577
12.74.1 Constructor & Destructor Documentation	578
12.74.1.1 userfile(const std::string &a_name)	578
12.74.1.2 ~userfile()	579
12.74.2 Member Function Documentation	579
12.74.2.1 parse(perform &a_perf)	579
12.74.2.2 write(const perform &a_perf)	579
12.74.2.3 dump_setting_summary()	579

Chapter 1

Sequencer64

Author(s) Chris Ahlstrom 2015-11-27

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 fair amount of side-code to cut down on the size of the document. For example, the main module, redundant Windows support, utility headers like `easy_macros.h`, standard stuff like the mutex module, the fruity variants (at least the ones already refactored into their own modules), etc., are all left out. Still, the resulting PDF is over 300 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 value 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 2016-01-23

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.

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

(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`
- `jack_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`
- `jack_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`
- `jack_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

Licenses

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

Author(s) Chris Ahlstrom 2015-09-10

5.1 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.

5.2 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>

5.3 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>

5.4 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>

5.5 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.

5.6 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 6

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::mainwnd::timeout ()`

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

Global `seq64::mainwnd::mainwnd (perform &a_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 *a_ev)`

Test this functionality in old and new application.

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`?

Global `seq64::perform::is_active (int seq) const`

We should have the sequence object keep track of its own activity and access that via a reference or pointer.

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

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

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

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

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.

Support the highlight feature in one or both perfedit windows in the same way it is done in the mainwid.

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_clear_perf` ()

All of `seq_paste()` can be offloaded to a (new) perform member function.

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::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)

Consider adding a feature where event's are rejected if their channel doesn't match that of the sequence. This has been a complaint of some people. Would modify the `add_event()` and `add_note()` functions.

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 7

Deprecated List

Global [seq64::clock_tick_duration_bogus](#) (int 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 8

Namespace Index

8.1 Namespace List

Here is a list of all namespaces with brief descriptions:

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

Chapter 9

Hierarchical Index

9.1 Class Hierarchy

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

seq64::AbstractPerfInput	83
seq64::FruityPerfInput	158
seq64::Seq24PerfInput	415
seq64::automutex	84
seq64::click	86
seq64::configfile	90
seq64::optionsfile	315
seq64::userfile	577
Dialog	
seq64::options	311
DrawingArea	
seq64::gui_palette_gtk2	182
seq64::gui_drawingarea_gtk2	170
seq64::eventslots	142
seq64::maintime	233
seq64::mainwid	238
seq64::perfnames	329
seq64::perfroll	383
seq64::perftime	397
seq64::seqdata	420
seq64::seqevent	446
seq64::seqkeys	456
seq64::seqroll	471
seq64::seqtime	493
seq64::editable_events	103
Entry	
seq64::keybindentry	206
seq64::event	109
seq64::editable_event	93
seq64::event_list::event_key	121
seq64::event_list	123
seq64::font	154
seq64::FruitySeqEventInput	162
seq64::FruitySeqRollInput	165

seq64::gui_assistant	167
seq64::gui_assistant_gtk2	169
seq64::jack_assistant	191
seq64::jack_scratchpad	205
seq64::jack_status_pair_t	206
seq64::keys_perform	209
seq64::keys_perform_gtk2	224
seq64::keys_perform_transfer	226
seq64::keystroke	228
seq64::lash	231
seq64::mastermidibus	260
seq64::midi_container	269
seq64::midi_list	275
seq64::midi_vector	286
seq64::midi_control	272
seq64::midi_measures	278
seq64::midi_splitter	280
seq64::midi_timing	283
seq64::midibus	288
seq64::midifile	294
seq64::mutex	310
seq64::condition_var	89
seq64::editable_event::name_value_t	311
ObjectBase	
seq64::seqmenu	464
seq64::mainwid	238
seq64::perfnames	329
seq64::perform	336
seq64::performcallback	381
seq64::mainwnd	248
seq64::rc_settings	405
seq64::rect	414
seq64::gui_drawingarea_gtk2::rect	414
seq64::Seq24SeqEventInput	418
seq64::sequence	496
seq64::trigger	538
seq64::triggers	540
seq64::user_instrument	550
seq64::user_instrument_t	553
seq64::user_midi_bus	554
seq64::user_midi_bus_t	557
seq64::user_settings	557
Window	
seq64::gui_window_gtk2	187
seq64::eventedit	131
seq64::mainwnd	248
seq64::perfeddit	318
seq64::seqedit	428

Chapter 10

Data Structure Index

10.1 Data Structures

Here are the data structures with brief descriptions:

seq64::AbstractPerfInput	Provides an abstract base class to provide the minimal interface for the various "perf input" classes	??
seq64::automutex	Provides a mutex that locks automatically when created, and unlocks when destroyed	??
seq64::click	Encapsulates any possible mouse click	??
seq64::condition_var	A mutex works best in conjunction with a condition variable	??
seq64::configfile	This class is the abstract base class for optionsfile and userfile	??
seq64::editable_event	Provides for the management of MIDI editable events	??
seq64::editable_events	Provides for the management of an ordered collection MIDI editable events	??
seq64::event	Provides events for management of MIDI events	??
seq64::event_list::event_key	Provides a key value for an event map	??
seq64::event_list	Receptable for MIDI events	??
seq64::eventedit	This class supports an Event Editor that is used to tweak the details of events and get a better idea of the mix of events in a sequence	??
seq64::eventslots	This class implements the left-side list of events in the pattern event-edit window	??
seq64::font	This class provides a wrapper for rendering fonts that are encoded as a 16 x 16 pixmap file in XPM format	??
seq64::FruityPerfInput	Implements the performance input of that certain fruity sequencer that people seem to like . . .	??
seq64::FruitySeqEventInput	This structure implements the interaction methods for the "fruity" mode of operation	??
seq64::FruitySeqRollInput	Implements the fruity mouse interaction paradigm for the seqroll	??

seq64::gui_assistant	This class provides an interface for some of the GUI support needed in Sequencer64	??
seq64::gui_assistant_gtk2	This class provides an interface for some of the Gtk/Gdk/Glib support needed in Sequencer64	??
seq64::gui_drawingarea_gtk2	Implements the basic drawing areas of the application	??
seq64::gui_palette_gtk2	Implements a stock palette of Gdk::Color elements	??
seq64::gui_window_gtk2	This class supports a basic interface for Gtk::Window-derived objects	??
seq64::jack_assistant	This class provides the performance mode JACK support	??
seq64::jack_scratchpad	Provide a temporary structure for passing data and results between a perform and jack_assistant object	??
seq64::jack_status_pair_t	Provides an internal type to make it easier to display a specific and accurate human-readable message when a JACK operation fails	??
seq64::keybindentry	Class for management of application key-bindings	??
seq64::keys_perform	This class supports the performance mode	??
seq64::keys_perform_gtk2	This class supports the performance mode	??
seq64::keys_perform_transfer	Provides a data-transfer structure to make it easier to fill in a keys_perform object's members using sscanf()	??
seq64::keystroke	Encapsulates any practical keystroke	??
seq64::lash	This class supports LASH operations, if compiled with LASH support (i.e	??
seq64::maintime	This class provides the drawing of the progress bar at the top of the main window, along with two "pills" that move in time with the beat and measure	??
seq64::mainwid	This class implements the piano roll area of the application	??
seq64::mainwnd	This class implements the functionality of the main window of the application, except for the Patterns Panel functionality, which is implemented in the mainwid class	??
seq64::mastermidibus	The class that "supervises" all of the midibus objects?	??
seq64::midi_container	This class is the abstract base class for a container of MIDI track information	??
seq64::midi_control	This class (formerly a struct) contains the control information for sequences that make up a live set	??
seq64::midi_list	This class is the std::list implementation of the midi_container	??
seq64::midi_measures	Provides a data structure to hold the numeric equivalent of the measures string "measures←:beats:divisions" ("m:b:d")	??
seq64::midi_splitter	This class handles the parsing and writing of MIDI files	??
seq64::midi_timing	We anticipate the need to have a small structure holding the parameters needed to calculate MIDI times within an arbitrary song	??
seq64::midi_vector	This class is the std::vector implementation of the midi_container	??

seq64::midibus	Provides a class for handling the MIDI buss on Linux	??
seq64::midifile	This class handles the parsing and writing of MIDI files	??
seq64::mutex	Simple wrapper for the pthread_mutex_t type used as a recursive mutex	??
seq64::editable_event::name_value_t	Provides a type that contains the pair of values needed for the various lookup maps that are needed to manage editable events	??
seq64::options	This class supports a full tabbed options dialog	??
seq64::optionsfile	Provides a file for reading and writing the application' main configuration file	??
seq64::perfedit	This class supports a Performance Editor that is used to arrange the patterns/sequences defined in the patterns panel	??
seq64::perfnames	This class implements the left-side keyboard in the patterns window	??
seq64::perform	This class supports the performance mode	??
seq64::performcallback	Provides for notification of events	??
seq64::perfroll	This class implements the performance roll user interface	??
seq64::perftime	This class implements drawing the piano time at the top of the "performance window" (the "song editor")	??
seq64::rc_settings	This class contains the options formerly named "global_XXXXXX"	??
seq64::rect	A small helper class representing a rectangle	??
seq64::gui_drawingarea_gtk2::rect	A small helper structure representing a rectangle	??
seq64::Seq24PerfInput	Implements the default (Seq24) performance input characteristics of this application	??
seq64::Seq24SeqEventInput	This structure implement the normal interaction methods for Seq24	??
seq64::seqdata	This class supports drawing piano-roll events on a window	??
seq64::seqedit	Implements the Pattern Editor, which has references to:	??
seq64::seqevent	Implements the piano event drawing area	??
seq64::seqkeys	This class implements the left side piano of the pattern/sequence editor	??
seq64::seqmenu	This class handles the right-click menu of the sequence slots in the pattern window	??
seq64::seqroll	Implements the piano roll section of the pattern editor	??
seq64::seqtime	This class implements the piano time, whatever that is	??
seq64::sequence	Firstly a receptable for a single track of MIDI data read from a MIDI file or edited into a pattern	??
seq64::trigger	This class hold a single trigger for a sequence object	??
seq64::triggers	Receptable the triggers that can be used with a sequence object	??

seq64::user_instrument	Provides data about the MIDI instruments, readable from the "user" configuration file	??
seq64::user_instrument_t	This structure corresponds to [user-instrument-N] definitions in the ~/.seq24usr or ~/.config/sequencer64/sequencer64.usr file	??
seq64::user_midi_bus	Provides data about the MIDI busses, readable from the "user" configuration file	??
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)	??
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	??
seq64::userfile	Supports the user's ~/.config/sequencer64/sequencer64.usr and ~/.seq24usr configuration file	??

Chapter 11

Namespace Documentation

11.1 Gtk Namespace Reference

11.2 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 [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 [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 [mastermidibus](#)
The class that "supervises" all of the midibus objects?
- 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_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_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 [midibus](#)
Provides a class for handling the MIDI buss on Linux.
- 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' 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 [perfname](#)
This class implements the left-side keyboard in the patterns window.
- class [perform](#)
This class supports the performance mode.
- struct [performcallback](#)
Provides for notification of events.
- class [perfull](#)
This class implements the performance roll user interface.
- class [perftime](#)
This class implements drawing the piano time at the top of the "performance window" (the "song editor").
- class [rc_settings](#)
This class contains the options formerly named "global_XXXXXX".
- class [rect](#)
A small helper class representing a rectangle.
- 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.

Enumerations

Functions

- 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, int 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 ×tring, int bpm, int ppqn)
- Converts a string that represents "hours:minutes:seconds.fraction" into a MIDI pulse/ticks/clock value.*

 - 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_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 [beats_per_minute_from_tempo](#) (double tempo)
- This function calculates the effective beats-per-minute based on the value of a Tempo meta-event.*

 - double [tempo_from_beats_per_minute](#) (double bpm)
- This function is the inverse of `beats_per_minute_from_tempo()`.*

 - double [pulse_length_us](#) (int bpm, int ppqn)
- Calculates pulse-length from the BPM (beats-per-minute) and PPQN (pulses-per-quarter-note) values.*

 - double [delta_time_us_to_ticks](#) (unsigned long us, int bpm, int ppqn)
- Converts delta time in microseconds to ticks.*

 - double [ticks_to_delta_time_us](#) (midipulse delta_ticks, int bpm, int ppqn)
- Converts the time in ticks ("clocks") to delta time in microseconds.*

 - double [clock_tick_duration_bogus](#) (int 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 [measures_to_ticks](#) (int bpm, int ppqn, int bw, int measures=1)

- Calculates the length of an integral number of measures, in ticks.*

 - 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, 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](#) (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 [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 shutdown 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_process_callback](#) (jack_nframes_t nframes, void *arg)
 - void [jack_session_callback](#) (jack_session_event_t *ev, void *arg)

Set the m_session_ev (event) value of the perform object.
 - 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 * [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.
- long [min](#) (long a, long b)

[min\(\)](#) for long values.
- [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.
- static std::string [make_section_name](#) (const std::string &label, int value)

Provides a purely internal, ad hoc helper function to create numbered section names for the userfile class.
- [font](#) & [font_render](#) ()

The p_font_renderer pointer was once created in the main module, sequencer64.cpp.
- Gtk::Adjustment & [adjustment_dummy](#) ()

Provides a way to provide a dummy Gtk::Adjustment object, but not create one until it is actually needed, so that the Glib/Gtk infrastructure is ready for it.
- 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.
- 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.

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_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_SYSEX](#)

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.

- const [midibyte EVENT_SYSEX_END](#)
- const [midibyte EVENT_SYSEX_CONTINUE](#)
- 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 [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_midictl](#)

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 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 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_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_highlight_empty](#)

This section of variables provide static information about the options enabled or disabled during the build.

- 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_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](#)
- 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 [perfedit](#) * [gs_perfedit_pointer_0](#)

Holds a pointer to the first instance of perfedit for the entire application, once it is created.

- static [perfedit](#) * [gs_perfedit_pointer_1](#)

Holds a pointer to the second instance of perfedit for the entire application, once it is created.

- static const int [c_select_all_notes](#)

Actions.

- static const int [c_select_all_events](#)
- static const int [c_select_inverse_notes](#)
- static const int [c_select_inverse_events](#)
- static const int [c_quantize_notes](#)
- static const int [c_quantize_events](#)
- static const int [c_tighten_events](#)
- static const int [c_tighten_notes](#)
- static const int [c_transpose_notes](#)

- static const int `c_reserved`
- static const int `c_transpose_h`
- static const int `c_swing_notes`
- static const long `s_handlesize`

An internal variable for handle size.

11.2.1 Detailed Description

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.

11.2.2 Typedef Documentation

11.2.2.1 typedef unsigned char `seq64::midibyte`

11.2.2.2 typedef unsigned char `seq64::bussbyte`

11.2.2.3 typedef unsigned short `seq64::midishort`

11.2.2.4 typedef unsigned long `seq64::midilong`

11.2.2.5 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.

11.2.3 Enumeration Type Documentation

11.2.3.1 enum `seq64::seq_modifier_t`

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

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

Enumerator

```
SEQ64_NO_MASK
SEQ64_SHIFT_MASK
SEQ64_LOCK_MASK
SEQ64_CONTROL_MASK
SEQ64_MOD1_MASK
SEQ64_MOD2_MASK
SEQ64_MOD3_MASK
```

```

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

```

11.2.3.2 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

```

11.2.3.3 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

```

11.2.3.4 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.

11.2.3.5 enum seq64::interaction_method_t

Moved here from the globals.h module.

Enumerator

e_seq24_interaction

e_fruity_interaction

e_number_of_interactions

11.2.3.6 enum seq64::c_music_scales

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

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

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

Enumerator

c_scale_off

c_scale_major

c_scale_minor

c_scale_harmonic_minor

c_scale_melodic_minor

c_scale_c_whole_tone

c_scale_blues

c_scale_major_pentatonic

c_scale_minor_pentatonic

c_scale_size

11.2.3.7 enum seq64::draw_type

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

Enumerator

- DRAW_FIN*** Indicates that drawing is finished?
- DRAW_NORMAL_LINKED*** Probably 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?

11.2.3.8 enum seq64::mouse_action_e

Enumerator

- e_action_select***
- e_action_draw***
- e_action_grow***

11.2.4 Function Documentation

11.2.4.1 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>part₁</i>	The destination reference for the first part of the time.
out	<i>part₂</i>	The destination reference for the second part of the time.
out	<i>part₃</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.

11.2.4.2 `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.

11.2.4.3 `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_ILLEGAL_PULSE, 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.

11.2.4.4 `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.

11.2.4.5 `std::string seq64::pulses_to_timestring (midipulse p, int 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.

11.2.4.6 `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.

11.2.4.7 `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.

11.2.4.8 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_ILLEGAL_PULSE is returned.

11.2.4.9 midipulse seq64::timestring_to_pulses (const std::string & *timestring*, int *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.

This conversion assumes that the fractional parts of the seconds is padded with zeroes on the left or right to 6 digits.

11.2.4.10 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).

11.2.4.11 `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.

11.2.4.12 `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.

11.2.4.13 `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.

11.2.4.14 `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.

11.2.4.15 `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.

11.2.4.16 `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.

11.2.4.17 `void seq64::tempo_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.

11.2.4.18 `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.

11.2.4.19 `double seq64::beats_per_minute_from_tempo (double tempo)` `[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>tempo</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 too small, then this function will crash. :-D

11.2.4.20 `double seq64::tempo_from_beats_per_minute (double bpm)` `[inline]`

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 too small, then this function will crash. :-D

11.2.4.21 `double seq64::pulse_length_us (int 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

11.2.4.22 `double seq64::delta_time_us_to_ticks (unsigned long us, int 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.

11.2.4.23 `double seq64::ticks_to_delta_time_us (midipulse delta_ticks, int 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 * double(delta_ticks) / (double(bpm) * double(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.

11.2.4.24 `double seq64::clock_tick_duration_bogus (int 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.

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

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

11.2.4.25 `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.

11.2.4.26 `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.

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

This function is called in [segedit::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. Seems small.

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.

11.2.4.28 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`, `--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.

11.2.4.29 bool seq64::parse_options_files (perform & p, 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.
<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.

11.2.4.30 `int seq64::parse_command_line_options (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>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.

11.2.4.31 `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 <code>perform</code> 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.

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

Returns

Returns an ordered, human-readable string enumerating the features.

11.2.4.33 `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.

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

11.2.4.35 `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.

11.2.4.36 `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.

11.2.4.37 `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.

11.2.4.38 `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.

11.2.4.39 `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.

11.2.4.40 `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.

11.2.4.41 `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.

11.2.4.42 `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 SEQ64_USE_DEFAULT_PPQN (-1).

11.2.4.43 `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.

Parameters

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

Returns

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

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

Parameters

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

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

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

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

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

The 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.

```

Parameters

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

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

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

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

```

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

```

Parameters

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

11.2.4.48 `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.

11.2.4.49 `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.
----------	---

11.2.4.50 `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 in 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.

11.2.4.51 `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.

11.2.4.52 `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.

11.2.4.53 `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.

11.2.4.54 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.

11.2.4.55 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*.

11.2.4.56 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.

11.2.4.57 user_settings& seq64::usr ()

Returns

Returns the global object g_user_settings.

11.2.4.58 int seq64::choose_ppqn (int *ppqn*)

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

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.

11.2.4.59 `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]".

11.2.4.60 `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`.

11.2.4.61 `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.

11.2.4.62 `void seq64::update_mainwid_sequences ()`

It is used by other objects that can modify the currently-edited sequence shown in the mainwid (main window).

11.2.4.63 `void seq64::update_perfedited_sequences ()`

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

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

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

11.2.5 Variable Documentation

11.2.5.1 `std::string seq64::c_controller_names`

This array is used only by the seqedit class.

11.2.5.2 `const midibyte seq64::EVENT_STATUS_BIT`

11.2.5.3 `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.

11.2.5.4 `const midibyte seq64::EVENT_NOTE_OFF`

11.2.5.5 `const midibyte seq64::EVENT_NOTE_ON`

11.2.5.6 `const midibyte seq64::EVENT_AFTERTOUCH`

11.2.5.7 `const midibyte seq64::EVENT_CONTROL_CHANGE`

11.2.5.8 `const midibyte seq64::EVENT_PROGRAM_CHANGE`

11.2.5.9 `const midibyte seq64::EVENT_CHANNEL_PRESSURE`

11.2.5.10 `const midibyte seq64::EVENT_PITCH_WHEEL`

11.2.5.11 `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
```

11.2.5.12 `const midibyte seq64::EVENT_MIDI_QUARTER_FRAME`

11.2.5.13 `const midibyte seq64::EVENT_MIDI_SONG_POS`

11.2.5.14 `const midibyte seq64::EVENT_MIDI_SONG_SELECT`

11.2.5.15 `const midibyte seq64::EVENT_MIDI_SONG_F4`

11.2.5.16 `const midibyte seq64::EVENT_MIDI_SONG_F5`

11.2.5.17 `const midibyte seq64::EVENT_MIDI_TUNE_SELECT`

11.2.5.18 `const midibyte seq64::EVENT_MIDI_SYSEX_END`

11.2.5.19 `const midibyte seq64::EVENT_MIDI_CLOCK`

11.2.5.20 `const midibyte seq64::EVENT_MIDI_SONG_F9`

11.2.5.21 `const midibyte seq64::EVENT_MIDI_START`

11.2.5.22 `const midibyte seq64::EVENT_MIDI_CONTINUE`

11.2.5.23 `const midibyte seq64::EVENT_MIDI_STOP`

11.2.5.24 `const midibyte seq64::EVENT_MIDI_SONG_FD`

11.2.5.25 `const midibyte seq64::EVENT_MIDI_ACTIVE_SENS`

11.2.5.26 `const midibyte seq64::EVENT_MIDI_RESET`

11.2.5.27 `const midibyte seq64::EVENT_MIDI_META`

11.2.5.28 `const midibyte seq64::EVENT_SYSEX`

11.2.5.29 `const midibyte seq64::EVENT_SYSEX_END`

11.2.5.30 `const midibyte seq64::EVENT_SYSEX_CONTINUE`

11.2.5.31 `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.

11.2.5.32 `const midibyte seq64::EVENT_GET_CHAN_MASK`

11.2.5.33 `const midibyte seq64::EVENT_CLEAR_CHAN_MASK`

11.2.5.34 `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.

11.2.5.35 `const int seq64::c_midibus_input_size`

Not sure if the value needs to be so large.

11.2.5.36 `const int seq64::c_midibus_sysex_chunk`

11.2.5.37 `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
```

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.

Track buss number.

11.2.5.38 `const midilong seq64::c_midich`

11.2.5.39 `const midilong seq64::c_midiclocks`

11.2.5.40 `const midilong seq64::c_triggers`

11.2.5.41 `const midilong seq64::c_notes`

11.2.5.42 `const midilong seq64::c_timesig`

11.2.5.43 `const midilong seq64::c_bpmtag`

11.2.5.44 `const midilong seq64::c_triggers_new`

11.2.5.45 `const midilong seq64::c_mutegroups`

11.2.5.46 `const midilong seq64::c_midictrl`

11.2.5.47 `const midilong seq64::c_musickey`

11.2.5.48 `const midilong seq64::c_musicscale`

11.2.5.49 `const midilong seq64::c_backsequence`

11.2.5.50 `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.

11.2.5.51 `const int seq64::c_midi_control_bpm_up`

11.2.5.52 `const int seq64::c_midi_control_bpm_dn`

11.2.5.53 `const int seq64::c_midi_control_ss_up`

11.2.5.54 `const int seq64::c_midi_control_ss_dn`

11.2.5.55 `const int seq64::c_midi_control_mod_replace`

11.2.5.56 `const int seq64::c_midi_control_mod_snapshot`

11.2.5.57 `const int seq64::c_midi_control_mod_queue`

11.2.5.58 `const int seq64::c_midi_control_mod_gmute`

11.2.5.59 `const int seq64::c_midi_control_mod_glearn`

11.2.5.60 `const int seq64::c_midi_control_play_ss`

11.2.5.61 `const int seq64::c_midi_controls`

11.2.5.62 `const bool seq64::c_scales_policy[c_scale_size][SEQ64_OCTAVE_SIZE]`

See the following sites for more information:

- <http://method-behind-the-music.com/theory/scalesandkeys/>
- https://en.wikipedia.org/wiki/Heptatonic_scale
- https://en.wikibooks.org/wiki/Music_Theory/Scales_and_Intervals

Note that melodic minor descends in the same way as the natural minor scale, so it descends differently than it ascends. We don't deal with that trick, at all. In the following table, the scales all start with C, but seq24/sequencer64 allow other starting notes (e.g. "keys").

Chromatic	C	C#	D	D#	E	F	F#	G	G#	A	A#	B	Notes, chord
Major	C	.	D	.	E	F	.	G	.	A	.	B	
Minor	C	.	D	Eb	.	F	.	G	Ab	.	Bb	.	
Harmonic Minor	C	.	D	Eb	.	F	.	G	Ab	.	.	B	
Melodic Minor	C	.	D	Eb	.	F	.	G	.	A	.	B	Descending diff.
C Whole Tone	C	.	D	.	E	.	F#	.	G#	.	A#	.	C+7 chord
Blues	C	.	.	Eb	.	F	Gb	G	.	.	Bb	.	
Major Pentatonic	C	.	D	.	E	.	.	G	.	A	.	.	
Minor Pentatonic	C	.	.	Eb	.	F	.	G	.	.	Bb	.	
Octatonic 1	C	.	D	Eb	.	F	Gb	.	Ab	A	.	B	Unimplemented
Octatonic 2	C	Db	.	Eb	E	F	F#	G	.	A	Bb	.	Unimplemented

11.2.5.63 `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	.	

11.2.5.64 `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 .

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

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

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

11.2.5.68 `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.

11.2.5.69 `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.

11.2.5.70 `const int seq64::c_max_busses`

11.2.5.71 `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"
```

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

Note the terminating null structure..

11.2.5.73 `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.

```
0123456789 @AaBbCcDdEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz
ooooooooo oxxxxxxx          xx xxx xxxxx x  xx xxxxx  xxx    x
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

11.2.5.91 `struct charpair_t seq64::s_character_mapping[]`

11.2.5.92 `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.

11.2.5.93 `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.

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

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

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

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

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

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

11.2.5.100 `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.

11.2.5.101 `const int seq64::c_mainwid_x`

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

11.2.5.102 `const int seq64::c_mainwid_y`

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

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

11.2.5.105 `const int seq64::c_select_all_notes` `[static]`

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.

11.2.5.106 `const int seq64::c_select_all_events` `[static]`

11.2.5.107 `const int seq64::c_select_inverse_notes` `[static]`

11.2.5.108 `const int seq64::c_select_inverse_events` `[static]`

11.2.5.109 `const int seq64::c_quantize_notes` `[static]`

11.2.5.110 `const int seq64::c_quantize_events` `[static]`

11.2.5.111 `const int seq64::c_tighten_events` `[static]`

11.2.5.112 `const int seq64::c_tighten_notes` `[static]`

11.2.5.113 `const int seq64::c_transpose_notes` `[static]`

11.2.5.114 `const int seq64::c_reserved` `[static]`

11.2.5.115 `const int seq64::c_transpose_h` `[static]`

11.2.5.116 `const int seq64::c_swing_notes` `[static]`

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

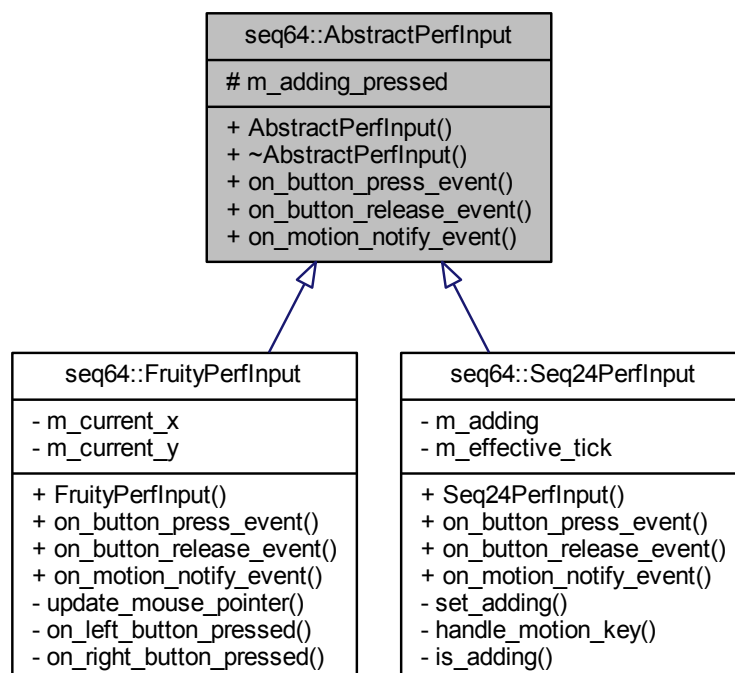
Chapter 12

Data Structure Documentation

12.1 seq64::AbstractPerfInput Class Reference

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

Inheritance diagram for seq64::AbstractPerfInput:



Public Member Functions

- [AbstractPerInput](#) ()
Default constructor.
- virtual [~AbstractPerInput](#) ()
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

Protected Attributes

- bool [m_adding_pressed](#)
Indicates if the left mouse button is pressed while in adding mode.

12.1.1 Constructor & Destructor Documentation

12.1.1.1 `seq64::AbstractPerInput::AbstractPerInput ()` [inline]

12.1.1.2 `virtual seq64::AbstractPerInput::~~AbstractPerInput ()` [inline], [virtual]

12.1.2 Member Function Documentation

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

Implemented in [seq64::Seq24PerInput](#), and [seq64::FruityPerInput](#).

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

Implemented in [seq64::Seq24PerInput](#), and [seq64::FruityPerInput](#).

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

Implemented in [seq64::Seq24PerInput](#), and [seq64::FruityPerInput](#).

12.1.3 Field Documentation

12.1.3.1 `bool seq64::AbstractPerInput::m_adding_pressed` [protected]

12.2 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.

12.2.1 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.

12.2.2 Constructor & Destructor Documentation

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

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

12.2.2.3 `seq64::automutex::automutex (mutex & my_mutex) [inline]`

Parameters

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

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

12.2.3 Member Function Documentation

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

12.2.4 Field Documentation

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

12.3 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.

12.3.1 Detailed Description

Useful in passing more generic events to non-GUI classes.

12.3.2 Constructor & Destructor Documentation

12.3.2.1 seq64::click::click ()

Sets all members to false, zero, or the lowest good value.

12.3.2.2 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.

12.3.2.3 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.
------------	---

12.3.3 Member Function Documentation

12.3.3.1 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.

12.3.3.2 `bool seq64::click::is_press () const [inline]`

12.3.3.3 `bool seq64::click::is_left () const [inline]`

12.3.3.4 `bool seq64::click::is_middle () const [inline]`

12.3.3.5 `bool seq64::click::is_right () const [inline]`

12.3.3.6 `int seq64::click::x () const [inline]`

12.3.3.7 `int seq64::click::y () const [inline]`

12.3.3.8 `int seq64::click::button () const [inline]`

12.3.3.9 `seq_modifier_t seq64::click::modifier () const [inline]`

12.3.3.10 `bool seq64::click::mod_control () const [inline]`

12.3.3.11 `bool seq64::click::mod_control_shift () const [inline]`

12.3.3.12 `bool seq64::click::mod_super () const [inline]`

12.3.4 Field Documentation

12.3.4.1 `bool seq64::click::m_is_press [private]`

12.3.4.2 `int seq64::click::m_x [private]`

0 is the left-most coordinate.

12.3.4.3 `int seq64::click::m_y [private]`

0 is the top-most coordinate.

12.3.4.4 `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.

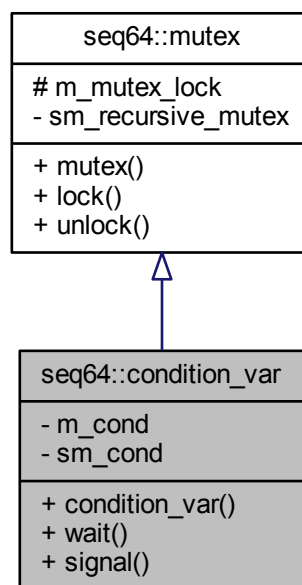
12.3.4.5 seq_modifier_t seq64::click::m_modifier [private]

Note that SEQ64_NO_MASK is our word for 0, meaning "no modifier".

12.4 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

12.4.1 Detailed Description

Therefore this class derives from the mutex class. A "has-a" relationship might be more logical than this "is-a" relationship.

12.4.2 Constructor & Destructor Documentation

12.4.2.1 `seq64::condition_var::condition_var ()`

12.4.3 Member Function Documentation

12.4.3.1 `void seq64::condition_var::wait ()`

12.4.3.2 `void seq64::condition_var::signal ()`

12.4.4 Field Documentation

12.4.4.1 `const pthread_cond_t seq64::condition_var::sm_cond` `[static], [private]`

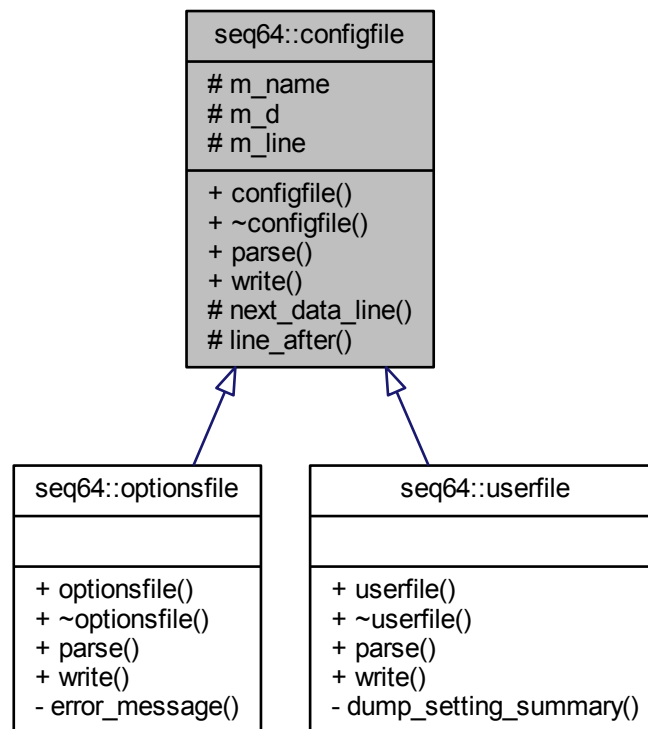
Define the static condition variable used by all mutex locks.

12.4.4.2 `pthread_cond_t seq64::condition_var::m_cond` `[private]`

12.5 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

Protected Member Functions

- bool [next_data_line](#) (std::ifstream &file)
Gets the next line of data from an input stream.
- void [line_after](#) (std::ifstream &file, const std::string &tag)
This function gets a specific line of text, specified as a tag.

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.

12.5.1 Constructor & Destructor Documentation

12.5.1.1 `seq64::configfile::configfile (const std::string & name)`

Parameters

<i>name</i>	The name of the configuration file.
-------------	-------------------------------------

12.5.1.2 `virtual seq64::configfile::~~configfile () [inline], [virtual]`

12.5.2 Member Function Documentation

12.5.2.1 `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.

12.5.2.2 `void 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.

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.

12.5.2.3 `virtual bool seq64::configfile::parse (perform & perf)` [pure virtual]

Implemented in [seq64::optionsfile](#), and [seq64::userfile](#).

12.5.2.4 `virtual bool seq64::configfile::write (const perform & perf)` [pure virtual]

Implemented in [seq64::optionsfile](#), and [seq64::userfile](#).

12.5.3 Field Documentation

12.5.3.1 `std::string seq64::configfile::m_name` [protected]

12.5.3.2 `char* seq64::configfile::m_d` [protected]

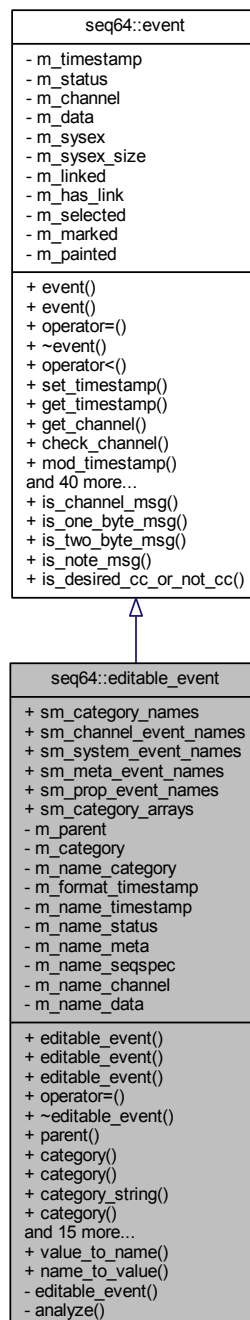
12.5.3.3 `char seq64::configfile::m_line[SEQ64_LINE_MAX]` [protected]

This member receives an input line, and so needs to be a character buffer.

12.6 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

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.

12.6.1 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.

12.6.2 Member Enumeration Documentation

12.6.2.1 `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.

12.6.2.2 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.

12.6.3 Constructor & Destructor Documentation

12.6.3.1 seq64::editable_event::editable_event () [private]

12.6.3.2 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.
---------------	---

12.6.3.3 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.

12.6.3.4 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.
------------	---

12.6.3.5 `virtual seq64::editable_event::~~editable_event () [inline],[virtual]`

12.6.4 Member Function Documentation

12.6.4.1 `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.

12.6.4.2 `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.

12.6.4.3 `editable_event & seq64::editable_event::operator= (const editable_event & rhs)`

12.6.4.4 `const editable_events& seq64::editable_event::parent () const [inline]`

12.6.4.5 `category_t seq64::editable_event::category () const [inline]`

12.6.4.6 `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.
----------	-------------------------------------

12.6.4.7 `const std::string& seq64::editable_event::category_string () const` `[inline]`

12.6.4.8 `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.
-------------	---

12.6.4.9 `const std::string& seq64::editable_event::timestamp_string () const` `[inline]`

12.6.4.10 `midipulse seq64::editable_event::timestamp () const` `[inline]`

12.6.4.11 `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.
-----------	---

12.6.4.12 `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.
------------------	---

12.6.4.13 `std::string seq64::editable_event::time_as_pulses ()` `[inline]`

12.6.4.14 `std::string seq64::editable_event::time_as_measures ()`

Cannot be inlined because of a circular dependency between the `editable_event` and `editable_events` classes.

12.6.4.15 `std::string seq64::editable_event::time_as_minutes ()`

Cannot be inlined because of a circular dependency between the `editable_event` and `editable_events` classes.

12.6.4.16 `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.

12.6.4.17 `std::string seq64::editable_event::format_timestamp ()`

The format of the string representation is of the format selected by the `m_format_timestamp` member.

12.6.4.18 `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.

12.6.4.19 `std::string seq64::editable_event::status_string () const [inline]`

12.6.4.20 `std::string seq64::editable_event::meta_string () const [inline]`

12.6.4.21 `std::string seq64::editable_event::seqspec_string () const [inline]`

12.6.4.22 `std::string seq64::editable_event::channel_string () const [inline]`

12.6.4.23 `std::string seq64::editable_event::data_string () const [inline]`

12.6.4.24 `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.

12.6.5 Field Documentation

12.6.5.1 `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.

12.6.5.2 `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.

12.6.5.3 `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.

12.6.5.4 `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.

12.6.5.5 `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.

12.6.5.6 `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.

12.6.5.7 `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.

12.6.5.8 `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.

12.6.5.9 `std::string seq64::editable_event::m_name_category` [private]

12.6.5.10 `timestamp_format_t seq64::editable_event::m_format_timestamp` [private]

The default is to display in timestamp_measures format.

12.6.5.11 `std::string seq64::editable_event::m_name_timestamp` [private]

12.6.5.12 `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.

12.6.5.13 `std::string seq64::editable_event::m_name_meta` [private]

If not applicable, this name will be empty.

12.6.5.14 `std::string seq64::editable_event::m_name_seqspect` [private]

12.6.5.15 `std::string seq64::editable_event::m_name_channel` [private]

12.6.5.16 `std::string seq64::editable_event::m_name_data` [private]

12.7 seq64::editable_events Class Reference

Provides for the management of an ordered collection MIDI editable events.

Public Member Functions

- [editable_events](#) ([sequence](#) &seq, int 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().

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](#)

12.7.1 Member Typedef Documentation

12.7.1.1 `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.

12.7.1.2 `typedef std::pair<Key, editable_event> seq64::editable_events::EventsPair` `[private]`

12.7.1.3 `typedef std::multimap<Key, editable_event> seq64::editable_events::Events` `[private]`

12.7.1.4 `typedef std::multimap<Key, editable_event>::iterator seq64::editable_events::iterator` `[private]`

12.7.1.5 `typedef std::multimap<Key, editable_event>::const_iterator seq64::editable_events::const_iterator`
`[private]`

12.7.2 Constructor & Destructor Documentation

12.7.2.1 `seq64::editable_events::editable_events ()` `[private]`

12.7.2.2 `seq64::editable_events::editable_events (sequence & seq, int 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.

12.7.2.3 `seq64::editable_events::editable_events (const editable_events & rhs)`

Note that we need to reconstitute the event links here, as well.

Parameters

<i>rhs</i>	Provides the editable_events object to be copied.
------------	---

12.7.2.4 `virtual seq64::editable_events::~~editable_events () [inline],[virtual]`

12.7.3 Member Function Documentation

12.7.3.1 `editable_events & seq64::editable_events::operator= (const editable_events & rhs)`

Note that we need to reconstitute the event links here, as well.

Parameters

<i>rhs</i>	Provides the editable_events object to be assigned.
------------	---

Returns

Returns a reference to "this" object, to support the serial assignment of `editable_eventss`.

12.7.3.2 `const midi_timing& seq64::editable_events::timing () const [inline]`12.7.3.3 `midipulse seq64::editable_events::string_to_pulses (const std::string & ts_string) const [inline]`12.7.3.4 `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.

12.7.3.5 `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.

12.7.3.6 `Events& seq64::editable_events::events ()` `[inline]`12.7.3.7 `iterator seq64::editable_events::begin ()` `[inline]`12.7.3.8 `const_iterator seq64::editable_events::begin () const` `[inline]`12.7.3.9 `iterator seq64::editable_events::end ()` `[inline]`12.7.3.10 `const_iterator seq64::editable_events::end () const` `[inline]`12.7.3.11 `int seq64::editable_events::count () const` `[inline]`

We like returning an integer instead of `size_t`, and rename the function so nobody is fooled.

12.7.3.12 `bool seq64::editable_events::add (const event & e)`**Parameters**

<code>e</code>	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.

12.7.3.13 `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.

12.7.3.14 `bool seq64::editable_events::replace (iterator ie, const editable_event & e)` `[inline]`

12.7.3.15 `void seq64::editable_events::remove (iterator ie)` `[inline]`

12.7.3.16 `void seq64::editable_events::clear ()` `[inline]`

12.7.3.17 `iterator seq64::editable_events::current_event () const` `[inline]`

12.7.3.18 `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.
------------	---

12.7.4 Friends And Related Function Documentation

12.7.4.1 `friend class eventslots` `[friend]`

12.7.5 Field Documentation

12.7.5.1 `Events seq64::editable_events::m_events` `[private]`

12.7.5.2 `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.

12.7.5.3 `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.

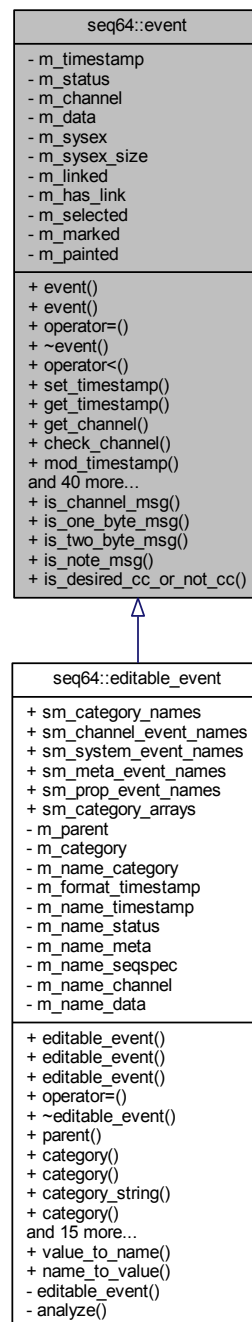
12.7.5.4 midi_timing seq64::editable_events::m_midi_parameters [private]

It holds the beats/minute, beats/measure, beat-width, and PPQN values needed to properly convert MIDI pulse timestamps to time and measure values.

12.8 seq64::event Class Reference

Provides events for management of MIDI events.

Inheritance diagram for seq64::event:



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 a_mod)`
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_channel (midibyte channel)`
Sets the channel "nybble", without modifying the status "nybble".
- `midibyte get_status () const`
'Getter' function for member `m_status`
- `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.
- `void restart_sysex ()`
Deletes and clears out the SYSEX buffer.
- `bool append_sysex (midibyte *data, int len)`
Appends SYSEX data to a new buffer.
- `midibyte * get_sysex () const`

- *'Getter' function for member m_sysex*
- void [set_sysex_size](#) (int len)
- *'Setter' function for member m_sysex_size*
- 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*
- 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].*
- [midibyte](#) [get_note_velocity](#) () const
- *'Getter' function for member m_data[1], the note velocity.*
- void [set_note_velocity](#) (int a_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
- bool [is_note_off](#) () const
- bool [is_note](#) () const
- *Returns true if m_status is a Note On, Note Off, or Aftertouch message.*
- 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_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.
- midibyte * m_sysex
Points to 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.".

12.8.1 Detailed Description

A MIDI event consists of 3 bytes:

- # Status byte, 1ssnnn, 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 xxxxxx values range from 0 to 127.
- # The second data byte, 0xxxxxxx.

This class may have too many member functions.

12.8.2 Constructor & Destructor Documentation

12.8.2.1 seq64::event::event ()

12.8.2.2 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.
------------	---

12.8.2.3 seq64::event::~~event () [virtual]

The [restart_sysex\(\)](#) function does what we need.

12.8.3 Member Function Documentation

12.8.3.1 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 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.

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.

12.8.3.2 `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.

12.8.3.3 `void seq64::event::set_timestamp (midipulse time) [inline]`

Parameters

<i>time</i>	Provides the time value, in ticks, to set as the timestamp.
-------------	---

12.8.3.4 `midipulse seq64::event::get_timestamp () const [inline]`

12.8.3.5 `midibyte seq64::event::get_channel () const [inline]`

12.8.3.6 `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.

12.8.3.7 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.
----------	--

Returns

Returns true if the byte represents a MIDI channel message.

12.8.3.8 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.
----------	--

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.

12.8.3.9 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.
----------	--

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.

12.8.3.10 static bool seq64::event::is_note_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.
----------	--

Returns

Returns true if the byte represents a MIDI note message.

12.8.3.11 `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.
```

12.8.3.12 `void seq64::event::mod_timestamp (midipulse a_mod)` `[inline]`

Parameters

<i>a_mod</i>	The tick value to mod the timestamp against.
--------------	--

Returns

Returns a value ranging from 0 to *a_mod*-1.

12.8.3.13 `void seq64::event::set_status (midibyte status)`

If *a_status* is a channel event, then the channel portion of the status is cleared using a bitwise AND against `EVENT_CLEAR_CHAN_MASK`.

Found in yet another fork of seq24:

```
// ORL fait de la merde
```

He also provided a very similar routine: `set_status_midibus()`.

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.
---------------	--

12.8.3.14 `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> .

12.8.3.15 `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.
----------------	-----------------------------

12.8.3.16 `midibyte seq64::event::get_status () const [inline]`

12.8.3.17 `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.
-----------	---

12.8.3.18 `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.

12.8.3.19 `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.

12.8.3.20 `void seq64::event::increment_data1 () [inline]`

12.8.3.21 `void seq64::event::decrement_data1 () [inline]`

12.8.3.22 `void seq64::event::increment_data2 () [inline]`

12.8.3.23 `void seq64::event::decrement_data2 () [inline]`

12.8.3.24 `void seq64::event::restart_sysex ()`

12.8.3.25 `bool seq64::event::append_sysex (midibyte * data, int dsize)`

First, a buffer of size `m_sysex_size+dsize` is created. The existing SYSEX data (stored in `m_sysex`) is copied to this buffer. Then the data represented by `data` and `dsize` is appended to that data buffer. Then the original SYSEX buffer, `m_sysex`, is deleted, and `m_sysex` is assigned to the new 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_SYSEX_END` byte in the appended data, or if an error occurred, and the caller needs to stop trying to process the data.

12.8.3.26 `midibyte* seq64::event::get_sysex () const [inline]`

12.8.3.27 `void seq64::event::set_sysex_size (int len) [inline]`

Parameters

<i>len</i>	Provides the length value to set as the size of the SYSEX data.
------------	---

12.8.3.28 `int seq64::event::get_sysex_size () const [inline]`

12.8.3.29 `void seq64::event::link (event * ev) [inline]`

Parameters

<i>a_event</i>	Provides a pointer to the event value to set. If null, then m_has_link is set to false, to guarantee that is_linked() is correct.
----------------	---

12.8.3.30 `event* seq64::event::get_linked () const` [inline]

12.8.3.31 `bool seq64::event::is_linked () const` [inline]

12.8.3.32 `void seq64::event::clear_link ()` [inline]

12.8.3.33 `void seq64::event::paint ()` [inline]

12.8.3.34 `void seq64::event::unpaint ()` [inline]

12.8.3.35 `bool seq64::event::is_painted () const` [inline]

12.8.3.36 `void seq64::event::mark ()` [inline]

12.8.3.37 `void seq64::event::unmark ()` [inline]

12.8.3.38 `bool seq64::event::is_marked () const` [inline]

12.8.3.39 `void seq64::event::select ()` [inline]

12.8.3.40 `void seq64::event::unselect ()` [inline]

12.8.3.41 `bool seq64::event::is_selected () const` [inline]

12.8.3.42 `void seq64::event::make_clock ()` [inline]

12.8.3.43 `midibyte seq64::event::data (int index) const` [inline]

12.8.3.44 `midibyte seq64::event::get_note () const` [inline]

12.8.3.45 `void seq64::event::set_note (midibyte note)` [inline]

Parameters

<i>note</i>	Provides the note value to set.
-------------	---------------------------------

12.8.3.46 `midibyte seq64::event::get_note_velocity () const` [inline]

12.8.3.47 `void seq64::event::set_note_velocity (int a_vel)` [inline]

Parameters

<code>a_vel</code>	Provides the velocity value to set.
--------------------	-------------------------------------

12.8.3.48 `bool seq64::event::is_note_on () const [inline]`

Returns

Returns true if `m_status` is `EVENT_NOTE_ON`.

12.8.3.49 `bool seq64::event::is_note_off () const [inline]`

Returns

Returns true if `m_status` is `EVENT_NOTE_OFF`.

12.8.3.50 `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.

12.8.3.51 `void seq64::event::print () const`

12.8.3.52 `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.

12.8.4 Field Documentation

12.8.4.1 `midipulse seq64::event::m_timestamp [private]`

12.8.4.2 `midibyte seq64::event::m_status [private]`

The channel will be appended on the MIDI bus. The high nibble = type of event; The low nibble = channel. Bit 7 is present in all status bytes.

12.8.4.3 `midibyte seq64::event::m_channel` `[private]`

This member adds another 4 bytes to the event object, most likely.

12.8.4.4 `midibyte seq64::event::m_data[SEQ64_MIDI_DATA_BYTE_COUNT]` `[private]`

Remember that the most-significant bit of a data byte is always 0.

12.8.4.5 `midibyte* seq64::event::m_sysex` `[private]`

This really ought to be a Boost or STD scoped pointer. Currently, it doesn't seem to be used.

12.8.4.6 `int seq64::event::m_sysex_size` `[private]`

12.8.4.7 `event* seq64::event::m_linked` `[private]`

12.8.4.8 `bool seq64::event::m_has_link` `[private]`

This item is used [via the `get_link()` and `link()` accessors] in the sequence class.

12.8.4.9 `bool seq64::event::m_selected` `[private]`

12.8.4.10 `bool seq64::event::m_marked` `[private]`

12.8.4.11 `bool seq64::event::m_painted` `[private]`

12.9 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.

12.9.1 Detailed Description

Its types match the `m_timestamp` and `get_rank()` function of this event class.

12.9.2 Constructor & Destructor Documentation

12.9.2.1 `seq64::event_list::event_key::event_key (midipulse tstamp, int rank)`

Parameters

<i>tstamp</i>	The time-stamp is the primary part of the key. It is the most important key item.
<i>rank</i>	Rank is an arbitrary number used to prioritize events that have the same time-stamp. See the event::get_rank() function for more information.

12.9.2.2 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.
------------	--------------------------------------

12.9.3 Member Function Documentation

12.9.3.1 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.

12.9.4 Field Documentation

12.9.4.1 midipulse seq64::event_list::event_key::m_timestamp [private]

12.9.4.2 int seq64::event_list::event_key::m_rank [private]

12.10 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, bool postsort=true)
Adds an event to the internal event list in an optionally sorted manner.
- 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](#) ()
Wrapper for std::list::sort(), or, since multimaps are always sorted, an empty function.

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.
- void [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 [midi_container](#)
- class [midi_splitter](#)
- class [sequence](#)

12.10.1 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.

12.10.2 Member Typedef Documentation

12.10.2.1 `typedef std::multimap<event_key, event> seq64::event_list::Events [private]`

12.10.2.2 `typedef std::pair<event_key, event> seq64::event_list::EventsPair [private]`

12.10.2.3 `typedef std::multimap<event_key, event>::iterator seq64::event_list::iterator [private]`

12.10.2.4 `typedef std::multimap<event_key, event>::const_iterator seq64::event_list::const_iterator [private]`

12.10.3 Constructor & Destructor Documentation

12.10.3.1 `seq64::event_list::event_list ()`

12.10.3.2 `seq64::event_list::event_list (const event_list & rhs)`

Parameters

<i>rhs</i>	Provides the event list to be copied.
------------	---------------------------------------

12.10.3.3 `seq64::event_list::~~event_list ()`

12.10.4 Member Function Documentation

12.10.4.1 `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.
------------	---

12.10.4.2 `iterator seq64::event_list::begin () [inline]`

12.10.4.3 `const_iterator seq64::event_list::begin () const [inline]`

12.10.4.4 `iterator seq64::event_list::end () [inline]`

12.10.4.5 `const_iterator seq64::event_list::end () const` `[inline]`

12.10.4.6 `int seq64::event_list::count () const` `[inline]`

We like returning an integer instead of `size_t`, and rename the function so nobody is fooled.

12.10.4.7 `bool seq64::event_list::empty () const` `[inline]`

12.10.4.8 `bool seq64::event_list::add (const event & e, bool postsort = true)`

It is a wrapper, wrapper for `insert()` or `push_front()`, with an option to call `sort()`.

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;
```

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.
<i>postsort</i>	If true, and the <code>std::list</code> implementation has been built in, then the event list is sorted after the addition. This is a time-consuming operation.

Returns

Returns true if the insertion succeeded, as evidenced by an increment in container size.

12.10.4.9 `bool seq64::event_list::is_modified () const` `[inline]`

12.10.4.10 `void seq64::event_list::unmodify ()` `[inline]`

But use it with great caution.

12.10.4.11 `void seq64::event_list::remove (iterator ie)` `[inline]`

Currently, no check on removal is performed. Set the modified-flag.

Parameters

<i>ie</i>	Provides the iterator to the event to be removed.
-----------	---

12.10.4.12 `void seq64::event_list::clear () [inline]`

Set the modified-flag.

12.10.4.13 `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.

12.10.4.14 `void seq64::event_list::sort () [inline]`

12.10.4.15 `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.
-----------	---

12.10.4.16 `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.
-----------	---

12.10.4.17 `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.

12.10.4.18 `void seq64::event_list::clear_links () [private]`

12.10.4.19 `void seq64::event_list::verify_and_link (midipulse slength) [private]`

Not threadsafe

Parameters

<i>slength</i>	Provides the length beyond which events will be pruned.
----------------	---

12.10.4.20 `void seq64::event_list::mark_selected () [private]`

12.10.4.21 `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.
----------------	---

12.10.4.22 `void seq64::event_list::mark_all () [private]`

Not yet used, but might come in handy with the event editor dialog.

12.10.4.23 `void seq64::event_list::unmark_all () [private]`

12.10.4.24 `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.

12.10.4.25 `void seq64::event_list::unpaint_all () [private]`

12.10.4.26 `int seq64::event_list::count_selected_notes () const [private]`

12.10.4.27 `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.

12.10.4.28 `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.

12.10.4.29 `void seq64::event_list::select_all () [private]`

12.10.4.30 `void seq64::event_list::unselect_all () [private]`

12.10.4.31 `void seq64::event_list::print () const [private]`

12.10.4.32 `const Events& seq64::event_list::events () const [inline], [private]`

12.10.5 Friends And Related Function Documentation

12.10.5.1 `friend class editable_events [friend]`

12.10.5.2 `friend class midi_container [friend]`

12.10.5.3 `friend class midi_splitter [friend]`

12.10.5.4 `friend class sequence [friend]`

12.10.6 Field Documentation

12.10.6.1 `Events seq64::event_list::m_events [private]`

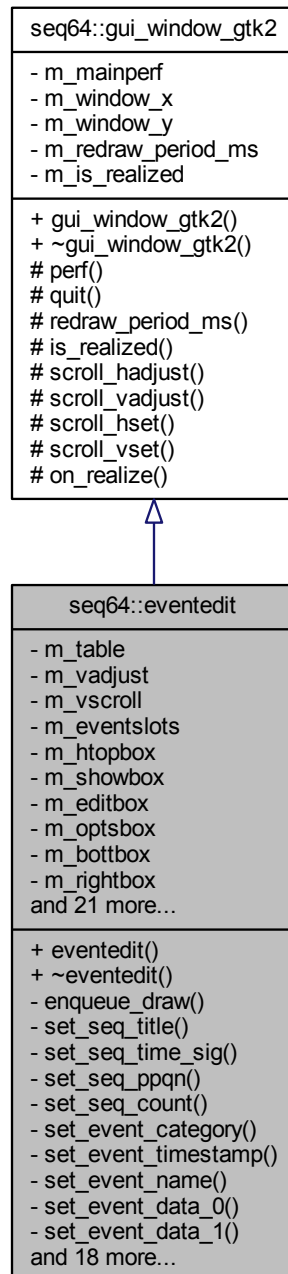
12.10.6.2 `bool seq64::event_list::m_is_modified [private]`

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.

12.11 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

12.11.1 Constructor & Destructor Documentation

12.11.1.1 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" | 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 ] | | 7
        | ...      ...      ...      | l | - - - - - - - - - - - - - | b |
        | ...      ...      ...      | | [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.

12.11.1.2 `seq64::eventedit::~eventedit ()` [virtual]

We're going to have to run the application through `valgrind` to make sure that nothing is left behind.

12.11.2 Member Function Documentation

12.11.2.1 `void seq64::eventedit::enqueue_draw ()` [private]

12.11.2.2 `void seq64::eventedit::set_seq_title (const std::string & title)` [private]

Parameters

<i>title</i>	The name of the sequence.
--------------	---------------------------

12.11.2.3 `void seq64::eventedit::set_seq_time_sig (const std::string & sig)` [private]

Parameters

<i>sig</i>	The time signature of the sequence.
------------	-------------------------------------

12.11.2.4 `void seq64::eventedit::set_seq_ppqn (const std::string & p)` [private]

Parameters

<i>p</i>	The parts-per-quarter-note string for the sequence.
----------	---

12.11.2.5 `void seq64::eventedit::set_seq_count ()` [private]

12.11.2.6 `void seq64::eventedit::set_event_category (const std::string & c)` [private]

Parameters

<i>c</i>	The category string for the current event.
----------	--

12.11.2.7 `void seq64::eventedit::set_event_timestamp (const std::string & ts)` [private]

Parameters

<i>ts</i>	The time-stamp string for the current event.
-----------	--

12.11.2.8 `void seq64::eventedit::set_event_name (const std::string & n)` [private]

Parameters

<i>n</i>	The name-of-event string for the current event.
----------	---

12.11.2.9 `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.
----------	---

12.11.2.10 `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.
----------	--

12.11.2.11 `void seq64::eventedit::perf_modify () [private]`

12.11.2.12 `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.
-------------	---

12.11.2.13 `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.
--------------	--

12.11.2.14 `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.

12.11.2.15 `void seq64::eventedit::change_focus (bool set_it = true) [private]`

Similar to the same function in `seqedit`.

Parameters

<code><i>set_it</i></code>	If true (the default value), indicates we want focus, otherwise we want to give up focus.
----------------------------	---

12.11.2.16 `void seq64::eventedit::close_out () [private]`

12.11.2.17 `void seq64::eventedit::handle_close () [private]`

Simply calls `close_out()`.

12.11.2.18 `void seq64::eventedit::handle_delete () [private]`

12.11.2.19 `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()`.

12.11.2.20 `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()`.

12.11.2.21 `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.

12.11.2.22 `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.

12.11.2.23 `void seq64::eventedit::on_realize () [private]`

Then it sets the vertical adjustment to account for the number of events in the eventslot.

12.11.2.24 `void seq64::eventedit::on_set_focus (Widget * focus) [private]`

Only works in certain circumstances.

Parameters

<i>focus</i>	The widget that has the focus. Merely passed on to gui_window_gtk2 's version of this function.
--------------	---

12.11.2.25 `bool seq64::eventedit::on_focus_in_event (GdkEventFocus *) [private]`

It sets the focus flag and calls [change_focus\(\)](#).

12.11.2.26 `bool seq64::eventedit::on_focus_out_event (GdkEventFocus *) [private]`

It resets the focus flag and calls [change_focus\(\)](#).

12.11.2.27 `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 some vi-like keys were supported, but they are needed for the edit fields, so cannot be used here. Also, 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 replace it by the slash. 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!

Parameters

<i>ev</i>	The key event to process.
-----------	---------------------------

Returns

Returns true if the event got handled somewhere along the line.

12.11.2.28 `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.

12.11.3 Friends And Related Function Documentation

12.11.3.1 friend class `eventslots` `[friend]`

12.11.4 Field Documentation

12.11.4.1 `Gtk::Table*` `seq64::eventedit::m_table` `[private]`

Provides the layout table for UI.

12.11.4.2 `Gtk::Adjustment*` `seq64::eventedit::m_vadjust` `[private]`

12.11.4.3 `Gtk::VScrollbar*` `seq64::eventedit::m_vscroll` `[private]`

12.11.4.4 `eventslots*` `seq64::eventedit::m_eventslots` `[private]`

12.11.4.5 `Gtk::HBox*` `seq64::eventedit::m_htopbox` `[private]`

12.11.4.6 `Gtk::VBox*` `seq64::eventedit::m_showbox` `[private]`

12.11.4.7 `Gtk::VBox*` `seq64::eventedit::m_editbox` `[private]`

12.11.4.8 `Gtk::VBox*` `seq64::eventedit::m_optsbox` `[private]`

12.11.4.9 `Gtk::HBox*` `seq64::eventedit::m_bottbox` `[private]`

12.11.4.10 `Gtk::VBox*` `seq64::eventedit::m_rightbox` `[private]`

12.11.4.11 `Gtk::Button*` `seq64::eventedit::m_button_del` `[private]`

12.11.4.12 `Gtk::Button*` `seq64::eventedit::m_button_ins` `[private]`

12.11.4.13 `Gtk::Button*` `seq64::eventedit::m_button_modify` `[private]`

12.11.4.14 `Gtk::Button*` `seq64::eventedit::m_button_save` `[private]`

12.11.4.15 `Gtk::Button*` `seq64::eventedit::m_button_cancel` `[private]`

12.11.4.16 `Gtk::Label*` `seq64::eventedit::m_label_seq_name` `[private]`

Shows the name of the pattern.

12.11.4.17 `Gtk::Label*` `seq64::eventedit::m_label_time_sig` [private]

12.11.4.18 `Gtk::Label*` `seq64::eventedit::m_label_ppqn` [private]

12.11.4.19 `Gtk::Label*` `seq64::eventedit::m_label_channel` [private]

12.11.4.20 `Gtk::Label*` `seq64::eventedit::m_label_ev_count` [private]

12.11.4.21 `Gtk::Label*` `seq64::eventedit::m_label_spacer` [private]

12.11.4.22 `Gtk::Label*` `seq64::eventedit::m_label_modified` [private]

12.11.4.23 `Gtk::Label*` `seq64::eventedit::m_label_category` [private]

Shows the type of MIDI event.

12.11.4.24 `Gtk::Entry*` `seq64::eventedit::m_entry_ev_timestamp` [private]

12.11.4.25 `Gtk::Entry*` `seq64::eventedit::m_entry_ev_name` [private]

12.11.4.26 `Gtk::Entry*` `seq64::eventedit::m_entry_ev_data_0` [private]

12.11.4.27 `Gtk::Entry*` `seq64::eventedit::m_entry_ev_data_1` [private]

12.11.4.28 `Gtk::Label*` `seq64::eventedit::m_label_time_fmt` [private]

12.11.4.29 `Gtk::Label*` `seq64::eventedit::m_label_right` [private]

12.11.4.30 `sequence&` `seq64::eventedit::m_seq` [private]

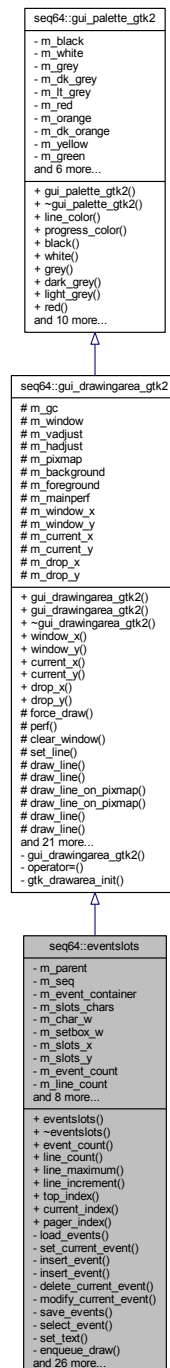
12.11.4.31 `bool` `seq64::eventedit::m_have_focus` [private]

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.

12.12 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)

Handles a button-release for the right button, bringing up a popup menu.
- 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

12.12.1 Constructor & Destructor Documentation

12.12.1.1 `seq64::eventslots::eventslots (perform & p, eventedit & parent, sequence & seq, Gtk::Adjustment & vadjust)`

12.12.1.2 `virtual seq64::eventslots::~~eventslots () [inline], [virtual]`

12.12.2 Member Function Documentation

12.12.2.1 `int seq64::eventslots::event_count () const [inline]`

12.12.2.2 `int seq64::eventslots::line_count () const [inline]`

12.12.2.3 `int seq64::eventslots::line_maximum () const [inline]`

12.12.2.4 `int seq64::eventslots::line_increment () const [inline]`

12.12.2.5 `int seq64::eventslots::top_index () const [inline]`

12.12.2.6 `int seq64::eventslots::current_index () const [inline]`

12.12.2.7 `int seq64::eventslots::pager_index () const [inline]`

12.12.2.8 `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.

12.12.2.9 `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.

12.12.2.10 `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

<code>edev</code>	The event to insert, prebuilt.
-------------------	--------------------------------

Returns

Returns true if the event was inserted.

12.12.2.11 `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

<code>evtimestamp</code>	The time-stamp of the new event, as obtained from the event-edit timestamp field.
<code>evname</code>	The type name (status name) of the new event, as obtained from the event-edit event-name field.
<code>evdata0</code>	The first data byte of the new event, as obtained from the event-edit data 1 field.
<code>evdata1</code>	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.

12.12.2.12 `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.

12.12.2.13 `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.

12.12.2.14 `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.

12.12.2.15 `void seq64::eventslots::select_event (int event_index = SEQ64_NULL_EVENT_INDEX, bool full_redraw = true) [private]`

The event index is provided by converting the y-coordinate of the mouse pointer into a slot number, and then an event index (actually the slot-distance from the `m_top_iterator`. Confusing, yes no?

Note that, if the event index is negative, then we just queue up a draw operation, which should paint an empty frame – the event container is empty.

Parameters

<i>event_index</i>	Provides the numeric index of the event in the event frame, or <code>SEQ64_NULL_EVENT</code> if there is no event to draw.
<i>full_redraw</i>	Defaulting to true, this parameter can be set to false in some case to reduce the flickering of the frame under fast movement.

12.12.2.16 `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.

12.12.2.17 `void seq64::eventslots::enqueue_draw () [private]`

12.12.2.18 `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`.

12.12.2.19 `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.

12.12.2.20 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 = ???

12.12.2.21 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.

12.12.2.22 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.
------------------	---

12.12.2.23 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.
-------------------	--

12.12.2.24 int seq64::eventslots::decrement_top () [private]

Returns

Returns 0, or SEQ64_NULL_EVENT_INDEX if the iterator could not be decremented.

12.12.2.25 `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.

12.12.2.26 `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.

12.12.2.27 `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.

12.12.2.28 `int seq64::eventslots::decrement_bottom () [private]`

Returns

Returns 0, or SEQ64_NULL_EVENT_INDEX if the iterator could not be decremented.

12.12.2.29 `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.

12.12.2.30 `void seq64::eventslots::on_realize () [private]`

It first calls the base-class version of [on_realize\(\)](#). Then it allocates any additional resources needed.

12.12.2.31 `bool seq64::eventslots::on_expose_event (GdkEventExpose * ev) [private]`

It draws all of the sequences.

12.12.2.32 `bool seq64::eventslots::on_button_press_event (GdkEventButton * ev) [private]`

12.12.2.33 `bool seq64::eventslots::on_button_release_event (GdkEventButton * ev) [private]`

12.12.2.34 `bool seq64::eventslots::on_focus_in_event (GdkEventFocus * ev) [private]`

See the same function in the perfrill module.

12.12.2.35 `bool seq64::eventslots::on_focus_out_event (GdkEventFocus * ev) [private]`

12.12.2.36 `bool seq64::eventslots::on_scroll_event (GdkEventScroll * ev) [private]`

12.12.2.37 `void seq64::eventslots::on_size_allocate (Gtk::Allocation & a) [private]`

It first calls the base-class version of this function.

12.12.2.38 `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\(\)](#).

12.12.2.39 `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\(\)](#).

12.12.2.40 `void seq64::eventslots::on_frame_up () [private]`

12.12.2.41 `void seq64::eventslots::on_frame_down () [private]`

12.12.2.42 `void seq64::eventslots::on_frame_home () [private]`

12.12.2.43 `void seq64::eventslots::on_frame_end () [private]`

12.12.3 Friends And Related Function Documentation

12.12.3.1 `friend class eventedit [friend]`

12.12.4 Field Documentation

12.12.4.1 `eventedit& seq64::eventslots::m_parent [private]`

12.12.4.2 `sequence& seq64::eventslots::m_seq [private]`

12.12.4.3 `editable_events seq64::eventslots::m_event_container [private]`

12.12.4.4 `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).

12.12.4.5 `int seq64::eventslots::m_char_w` `[private]`

This value is obtained from a font-renderer accessor function.

12.12.4.6 `int seq64::eventslots::m_setbox_w` `[private]`

This used to be hardwired to $6 * 2$ (character-width times two).

12.12.4.7 `int seq64::eventslots::m_slots_x` `[private]`

12.12.4.8 `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.

12.12.4.9 `int seq64::eventslots::m_event_count` `[private]`

12.12.4.10 `int seq64::eventslots::m_line_count` `[private]`

12.12.4.11 `int seq64::eventslots::m_line_maximum` `[private]`

12.12.4.12 `int seq64::eventslots::m_line_overlap` `[private]`

12.12.4.13 `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.

12.12.4.14 `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.

12.12.4.15 `editable_events::iterator seq64::eventslots::m_top_iterator` `[private]`

12.12.4.16 `editable_events::iterator seq64::eventslots::m_bottom_iterator` `[private]`

12.12.4.17 `editable_events::iterator seq64::eventslots::m_current_iterator` `[private]`

12.12.4.18 `int seq64::eventslots::m_pager_index` `[private]`

12.13 `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

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) 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.

12.13.1 Member Enumeration Documentation

12.13.1.1 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.

12.13.2 Constructor & Destructor Documentation

12.13.2.1 seq64::font::font ()

12.13.3 Member Function Documentation

12.13.3.1 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.
-----------	---

12.13.3.2 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) 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.

Parameters

<i>col</i>	The font color to use to draw the string. The supported values are <code>font::BLACK</code> , <code>font::WHITE</code> , <code>font::BLACK_ON_YELLOW</code> , <code>font::YELLOW_ON_BLACK</code> . The actual correct colors are provided by selecting one of four font pixmaps, as described in the <code>init()</code> function.
------------	--

12.13.3.3 `int seq64::font::char_width () const [inline]`

12.13.3.4 `int seq64::font::char_height () const [inline]`

12.13.3.5 `int seq64::font::padded_height () const [inline]`

12.13.4 Field Documentation

12.13.4.1 `bool seq64::font::m_use_new_font [private]`

12.13.4.2 `int seq64::font::m_cell_w [private]`

12.13.4.3 `int seq64::font::m_cell_h [private]`

12.13.4.4 `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.

12.13.4.5 `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.

12.13.4.6 `int seq64::font::m_offset [private]`

12.13.4.7 `int seq64::font::m_padded_h [private]`

12.13.4.8 `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).

12.13.4.9 `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.

12.13.4.10 `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.

12.13.4.11 `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.

12.13.4.12 `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.

12.13.4.13 `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.

12.13.4.14 `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.

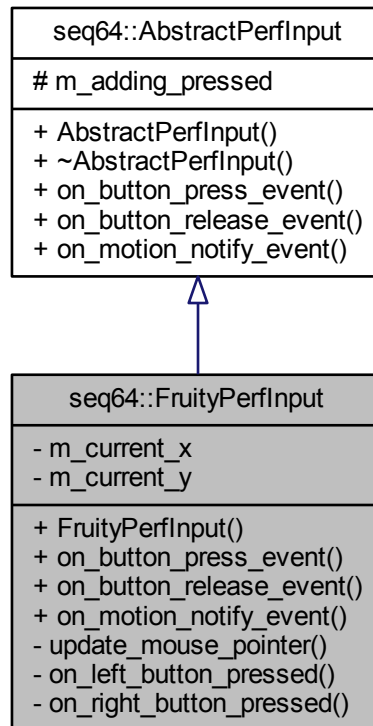
12.13.4.15 `Glib::RefPtr<Gdk::Bitmap> seq64::font::m_clip_mask` `[private]`

All we know is it seems to be a requirement for creating a pixmap object from an XMP file.

12.14 `seq64::FruityPerfInput` Class Reference

Implements the performance input of that certain fruity sequencer that people seem to like.

Inheritance diagram for seq64::FruityPerfInput:



Public Member Functions

- [FruityPerfInput \(\)](#)
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.

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 [perfroll](#)

Additional Inherited Members

12.14.1 Constructor & Destructor Documentation

12.14.1.1 `seq64::FruityPerfInput::FruityPerfInput ()` `[inline]`

12.14.2 Member Function Documentation

12.14.2.1 `bool seq64::FruityPerfInput::on_button_press_event (GdkEventButton * ev, perfroll & 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](#).

12.14.2.2 `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](#).

12.14.2.3 `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](#).

12.14.2.4 `void seq64::FruityPerfInput::update_mouse_pointer (perfroll & 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.
-------------	--

12.14.2.5 `bool seq64::FruityPerfInput::on_left_button_pressed (GdkEventButton * ev, perfroll & 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.

12.14.2.6 `bool seq64::FruityPerfInput::on_right_button_pressed (GdkEventButton * ev, perfroll & 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.

12.14.3 Friends And Related Function Documentation

12.14.3.1 friend class `perfroll` [`friend`]

12.14.4 Field Documentation

12.14.4.1 long `seq64::FruityPerfInput::m_current_x` [`private`]

12.14.4.2 long `seq64::FruityPerfInput::m_current_y` [`private`]

12.15 `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](#) ([sequevent](#) &ths)
Provides support for a context-sensitive mouse.
- 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 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.

12.15.1 Constructor & Destructor Documentation

12.15.1.1 seq64::FruitySeqEventInput::FruitySeqEventInput () `[inline]`

12.15.2 Member Function Documentation

12.15.2.1 void seq64::FruitySeqEventInput::update_mouse_pointer (*seqevent* & *seqev*)

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).
--------------	--

12.15.2.2 `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:

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.

12.15.2.3 `bool seq64::FruitySeqEventInput::on_button_release_event (GdkEventButton * ev, sequevent & segev)`

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.

12.15.2.4 `bool seq64::FruitySeqEventInput::on_motion_notify_event (GdkEventMotion * ev, seqevent & seqev)`

Parameters

<code>ev</code>	The button event for the press of a mouse button.
<code>seqev</code>	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.

12.15.3 Field Documentation

12.15.3.1 `bool seq64::FruitySeqEventInput::m_justselected_one`

12.15.3.2 `bool seq64::FruitySeqEventInput::m_is_drag_pasting_start`

12.15.3.3 `bool seq64::FruitySeqEventInput::m_is_drag_pasting`

12.16 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.

12.16.1 Constructor & Destructor Documentation

12.16.1.1 `seq64::FruitySeqRollInput::FruitySeqRollInput ()` `[inline]`

12.16.2 Member Function Documentation

12.16.2.1 `void seq64::FruitySeqRollInput::update_mouse_pointer (seqroll & sroll)`

Parameters

<i>scroll</i>	Provides the "parent" of this interaction class.
---------------	--

12.16.2.2 `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.

12.16.2.3 `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.

12.16.2.4 `bool seq64::FruitySeqRollInput::on_motion_notify_event (GdkEventMotion * ev, seqroll & scroll)`

Parameters

<i>ev</i>	The motion event.
<i>scroll</i>	The parent of this "fruity" interaction class. (Why not just inherit and save all these indirect accesses to the <code>seqroll</code> ? 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.

12.16.3 Field Documentation

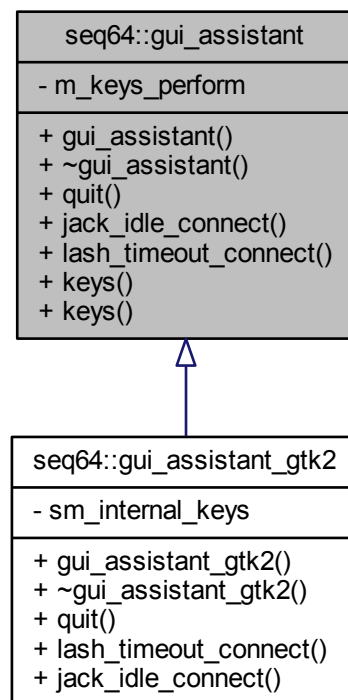
12.16.3.1 `bool seq64::FruitySeqRollInput::m_erase_painting` `[private]`

12.16.3.2 `int seq64::FruitySeqRollInput::m_drag_paste_start_pos[2]` `[private]`

12.17 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.

12.17.1 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).

12.17.2 Constructor & Destructor Documentation

12.17.2.1 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.
-----------	---

12.17.2.2 virtual seq64::gui_assistant::~~gui_assistant () [inline],[virtual]

12.17.3 Member Function Documentation

12.17.3.1 virtual void seq64::gui_assistant::quit () [pure virtual]

Implemented in [seq64::gui_assistant_gtk2](#).

12.17.3.2 virtual void seq64::gui_assistant::jack_idle_connect ([jack_assistant](#) & *jack*) [pure virtual]

Implemented in [seq64::gui_assistant_gtk2](#).

12.17.3.3 virtual void seq64::gui_assistant::lash_timeout_connect (*lash* * *lashobject*) [pure virtual]

Implemented in [seq64::gui_assistant_gtk2](#).

12.17.3.4 const [keys_perform](#)& seq64::gui_assistant::keys () const [inline]

12.17.3.5 [keys_perform](#)& seq64::gui_assistant::keys () [inline]

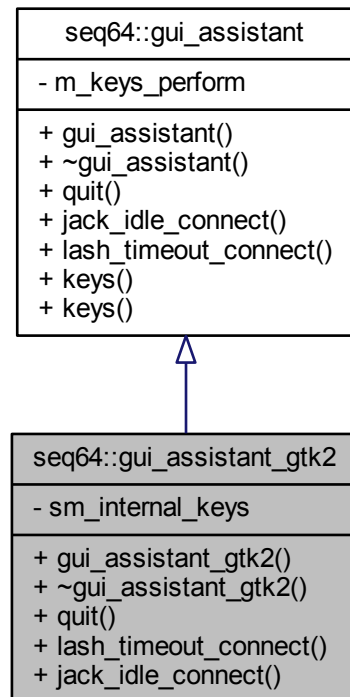
12.17.4 Field Documentation

12.17.4.1 [keys_perform](#)& seq64::gui_assistant::m_keys_perform [private]

12.18 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.

12.18.1 Constructor & Destructor Documentation

12.18.1.1 `seq64::gui_assistant_gtk2::gui_assistant_gtk2 ()`

12.18.1.2 `virtual seq64::gui_assistant_gtk2::~~gui_assistant_gtk2 () [inline], [virtual]`

12.18.2 Member Function Documentation

12.18.2.1 `void seq64::gui_assistant_gtk2::quit () [virtual]`

Implements [seq64::gui_assistant](#).

12.18.2.2 `void seq64::gui_assistant_gtk2::lash_timeout_connect (lash * lashobject) [virtual]`

The time-out value is set to 250 ms.

Implements [seq64::gui_assistant](#).

12.18.2.3 `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](#).

12.18.3 Field Documentation

12.18.3.1 `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.

12.19 seq64::gui_drawingarea_gtk2 Class Reference

Implements the basic drawing areas of the application.

- struct `rect`
A small helper structure representing a rectangle.

- `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)`
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)`
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

12.19.1 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.

12.19.2 Constructor & Destructor Documentation

12.19.2.1 [seq64::gui_drawingarea_gtk2::gui_drawingarea_gtk2](#) (const [gui_drawingarea_gtk2](#) &) [private]

12.19.2.2 [seq64::gui_drawingarea_gtk2::gui_drawingarea_gtk2](#) ([perform](#) & *p*, int *window_x* = 0, int *window_y* = 0)

12.19.2.3 [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)

12.19.2.4 [seq64::gui_drawingarea_gtk2::~~gui_drawingarea_gtk2](#) () [virtual]

12.19.3 Member Function Documentation

12.19.3.1 [gui_drawingarea_gtk2&](#) [seq64::gui_drawingarea_gtk2::operator=](#) (const [gui_drawingarea_gtk2](#) &) [private]

12.19.3.2 int [seq64::gui_drawingarea_gtk2::window_x](#) () const [inline]

12.19.3.3 int [seq64::gui_drawingarea_gtk2::window_y](#) () const [inline]

12.19.3.4 int [seq64::gui_drawingarea_gtk2::current_x](#) () const [inline]

12.19.3.5 int [seq64::gui_drawingarea_gtk2::current_y](#) () const [inline]

12.19.3.6 int [seq64::gui_drawingarea_gtk2::drop_x](#) () const [inline]

12.19.3.7 int [seq64::gui_drawingarea_gtk2::drop_y](#) () const [inline]

12.19.3.8 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](#).

12.19.3.9 `perform& seq64::gui_drawingarea_gtk2::perf ()` `[inline]`, `[protected]`

12.19.3.10 `void seq64::gui_drawingarea_gtk2::clear_window ()` `[inline]`, `[protected]`

One less need to access `m_window` directly.

12.19.3.11 `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.

12.19.3.12 `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.

12.19.3.13 `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.

12.19.3.14 `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.

12.19.3.15 `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.

12.19.3.16 `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.

12.19.3.17 `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.

12.19.3.18 `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.

12.19.3.19 `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.

12.19.3.20 `void seq64::gui_drawingarea_gtk2::render_string (int x, int y, const std::string & s, font::Color color)`
`[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.

12.19.3.21 `void seq64::gui_drawingarea_gtk2::render_string_on_pixmap (int x, int y, const std::string & s, font::Color color)`
`[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.

12.19.3.22 `void seq64::gui_drawingarea_gtk2::draw_rectangle (int x, int y, int lx, int ly, bool fill = true)` `[inline], [protected]`

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.23 `void seq64::gui_drawingarea_gtk2::draw_rectangle (const Color & c, int x, int y, int lx, int ly, bool fill = true)`
`[protected]`

It adds setting the foreground color to the [draw_rectangle\(\)](#) function.

Parameters

<i>c</i>	Provides the foreground color to set.
<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.24 `void seq64::gui_drawingarea_gtk2::draw_rectangle (Glib::RefPtr< Gdk::Drawable > & drawable, int x, int y, int lx, int ly, bool fill = true) [inline], [protected]`

Parameters

<i>drawable</i>	The object on which to draw the rectangle.
<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.25 `void seq64::gui_drawingarea_gtk2::draw_rectangle (Glib::RefPtr< Gdk::Drawable > & drawable, const Color & c, int x, int y, int lx, int ly, bool fill = true) [protected]`

It also supports setting the foreground color to the [draw_rectangle\(\)](#) function.

We have a number of such functions: for the main window, for the main pixmap, and for any drawing surface. Is the small bit of conciseness worth it?

Parameters

<i>drawable</i>	The surface on which to draw the box.
<i>c</i>	Provides the foreground color to set.
<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.26 `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.
---------------	--

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.27 `void seq64::gui_drawingarea_gtk2::draw_rectangle (Glib::RefPtr< Gdk::Pixmap > & pixmap, const Color & c, int x, int y, int lx, int ly, bool fill = true)` `[protected]`

It also supports setting the foreground color to the [draw_rectangle\(\)](#) function.

We have a number of such functions: for the main window, for the main pixmap, and for any drawing surface. Is the small bit of conciseness worth it?

Parameters

<i>pixmap</i>	The surface on which to draw the box.
<i>c</i>	Provides the foreground color to set.
<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.28 `void seq64::gui_drawingarea_gtk2::draw_rectangle_on_pixmap (int x, int y, int lx, int ly, bool fill = true)` `[inline], [protected]`

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.29 `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.

Parameters

<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.30 `void seq64::gui_drawingarea_gtk2::draw_normal_rectangle_on_pixmap (int x, int y, int lx, int ly, bool fill = true)` [protected]

It uses [Gtk](#) to get the proper background styling for the rectangle.

Parameters

<i>x</i>	The x-coordinate of the origin.
<i>y</i>	The y-coordinate of the origin.
<i>lx</i>	The width of the box.
<i>ly</i>	The height of the box.
<i>fill</i>	If true, fill the rectangle with the current foreground color, as set by <code>m_gc->set_foreground(color)</code> . Defaults to true.

12.19.3.31 `void seq64::gui_drawingarea_gtk2::draw_drawable (int xsrc, int ysrc, int xdest, int ydest, int width, int height)` [inline], [protected]

12.19.3.32 `void seq64::gui_drawingarea_gtk2::scroll_hadjust (Gtk::Adjustment & hadjust, double step)` [protected]

This function is currently duplicated in the [gui_drawingarea_gtk2](#) and [gui_window_gtk2](#) modules.

Parameters

<i>hadjust</i>	Provides a reference to the adjustment object to be adjusted. Do we really need this to be a parameter? Why not just use the <code>m_hadjust</code> member? (Note that this member is not present in the similar gui_window_gtk2 class.)
<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. If negative, the adjustment is leftward. If positive, the adjustment is rightward. It can be the value of <code>m_hadjust->get_step_increment()</code> , or provided especially to keep up with the progress bar.

12.19.3.33 `void seq64::gui_drawingarea_gtk2::scroll_vadjust (Gtk::Adjustment & vadjust, double step)` [protected]

Parameters

<i>vadjust</i>	Provides a reference to the adjustment object to be adjusted.
<i>step</i>	Provides the step value to use for adjusting the vertical scrollbar. If negative, the adjustment is upward. If positive, the adjustment is downward. It can be the value of <code>m_vadjust->get_step_increment()</code> .

12.19.3.34 void seq64::gui_drawingarea_gtk2::scroll_hset (Gtk::Adjustment & *hadjust*, double *value*) [protected]

12.19.3.35 void seq64::gui_drawingarea_gtk2::scroll_vset (Gtk::Adjustment & *vadjust*, double *value*) [protected]

12.19.3.36 void seq64::gui_drawingarea_gtk2::set_current_drop_x (int *x*) [inline], [protected]

Parameters

<i>x</i>	The x value to be set.
----------	------------------------

12.19.3.37 void seq64::gui_drawingarea_gtk2::set_current_drop_y (int *y*) [inline], [protected]

Parameters

<i>y</i>	The y value to be set.
----------	------------------------

12.19.3.38 void seq64::gui_drawingarea_gtk2::gtk_drawarea_init () [private]

12.19.3.39 void seq64::gui_drawingarea_gtk2::on_realize () [protected]

It allocates any additional resources that weren't initialized in the constructor.

12.19.4 Field Documentation

12.19.4.1 Glib::RefPtr<Gdk::GC> seq64::gui_drawingarea_gtk2::m_gc [protected]

12.19.4.2 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.

12.19.4.3 Gtk::Adjustment& seq64::gui_drawingarea_gtk2::m_vadjust [protected]

12.19.4.4 Gtk::Adjustment& seq64::gui_drawingarea_gtk2::m_hadjust [protected]

12.19.4.5 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.

12.19.4.6 Glib::RefPtr<Gdk::Pixmap> seq64::gui_drawingarea_gtk2::m_background [protected]

Our wrappers still leave this member exposed (giggle).

Public Member Functions

- [gui_palette_gtk2](#) ()
Principal constructor.
- [~gui_palette_gtk2](#) ()
Provides a destructor to delete allocated objects.
- const [Color](#) & [line_color](#) () const
'Getter' function for member m_line_color Provides an experimental way to change some line colors from black to something else.
- const [Color](#) & [progress_color](#) () const
'Getter' function for member m_progress_color Provides an experimental way to change the progress line color from black to something else.
- const [Color](#) & [black](#) () const
'Getter' function for member m_black
- 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](#) & [dark_orange](#) () const
'Getter' function for member m_dk_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](#) & [dark_cyan](#) () const
'Getter' function for member m_dk_cyan
- 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

Protected Types

- typedef Gdk::Color [Color](#)
Provides a type for the color object.

Private Attributes

- const [Color m_black](#)
Provides the black color.
- const [Color m_white](#)
Provides the white color.
- const [Color m_grey](#)
Provides the grey color.
- const [Color m_dk_grey](#)
Provides the dark grey color.
- const [Color m_lt_grey](#)
Provides the light grey color.
- const [Color m_red](#)
Provides the red color.
- const [Color m_orange](#)
Provides the orange color.
- const [Color m_dk_orange](#)
Provides a dark orange color.
- const [Color m_yellow](#)
Provides the yellow color.
- const [Color m_green](#)
Provides the green color.
- const [Color m_blue](#)
Provides the blue color.
- const [Color m_dk_cyan](#)
Provides the dark cyan color.
- const [Color m_line_color](#)
Provides the line color.
- const [Color m_progress_color](#)
Provides the progress color.
- [Color m_bg_color](#)
The background color.
- [Color m_fg_color](#)
The foreground color.

12.20.1 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.

12.20.2 Member Typedef Documentation

12.20.2.1 `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.

12.20.3 Constructor & Destructor Documentation

12.20.3.1 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.

12.20.3.2 seq64::gui_palette_gtk2::~~gui_palette_gtk2 ()

12.20.4 Member Function Documentation

12.20.4.1 const Color& seq64::gui_palette_gtk2::line_color () const [inline]

Might eventually be selectable from the "user" configuration file

12.20.4.2 const Color& seq64::gui_palette_gtk2::progress_color () const [inline]

Might eventually be selectable from the "user" configuration file

12.20.4.3 `const Color& seq64::gui_palette_gtk2::black () const` `[inline]`

12.20.4.4 `const Color& seq64::gui_palette_gtk2::white () const` `[inline]`

12.20.4.5 `const Color& seq64::gui_palette_gtk2::grey () const` `[inline]`

12.20.4.6 `const Color& seq64::gui_palette_gtk2::dark_grey () const` `[inline]`

12.20.4.7 `const Color& seq64::gui_palette_gtk2::light_grey () const` `[inline]`

12.20.4.8 `const Color& seq64::gui_palette_gtk2::red () const` `[inline]`

12.20.4.9 `const Color& seq64::gui_palette_gtk2::orange () const` `[inline]`

12.20.4.10 `const Color& seq64::gui_palette_gtk2::dark_orange () const` `[inline]`

12.20.4.11 `const Color& seq64::gui_palette_gtk2::yellow () const` `[inline]`

12.20.4.12 `const Color& seq64::gui_palette_gtk2::green () const` `[inline]`

12.20.4.13 `const Color& seq64::gui_palette_gtk2::blue () const` `[inline]`

12.20.4.14 `const Color& seq64::gui_palette_gtk2::dark_cyan () const` `[inline]`

12.20.4.15 `const Color& seq64::gui_palette_gtk2::bg_color () const` `[inline]`

12.20.4.16 `void seq64::gui_palette_gtk2::bg_color (const Color & c)` `[inline]`

12.20.4.17 `const Color& seq64::gui_palette_gtk2::fg_color () const` `[inline]`

12.20.4.18 `void seq64::gui_palette_gtk2::fg_color (const Color & c)` `[inline]`

12.20.5 Field Documentation

12.20.5.1 `const Color seq64::gui_palette_gtk2::m_black` `[private]`

12.20.5.2 `const Color seq64::gui_palette_gtk2::m_white` `[private]`

12.20.5.3 `const Color seq64::gui_palette_gtk2::m_grey` `[private]`

12.20.5.4 `const Color seq64::gui_palette_gtk2::m_dk_grey` `[private]`

12.20.5.5 `const Color seq64::gui_palette_gtk2::m_lt_grey` `[private]`

12.20.5.6 `const Color seq64::gui_palette_gtk2::m_red` `[private]`

12.20.5.7 `const Color seq64::gui_palette_gtk2::m_orange` [private]

12.20.5.8 `const Color seq64::gui_palette_gtk2::m_dk_orange` [private]

12.20.5.9 `const Color seq64::gui_palette_gtk2::m_yellow` [private]

12.20.5.10 `const Color seq64::gui_palette_gtk2::m_green` [private]

12.20.5.11 `const Color seq64::gui_palette_gtk2::m_blue` [private]

12.20.5.12 `const Color seq64::gui_palette_gtk2::m_dk_cyan` [private]

12.20.5.13 `const Color seq64::gui_palette_gtk2::m_line_color` [private]

12.20.5.14 `const Color seq64::gui_palette_gtk2::m_progress_color` [private]

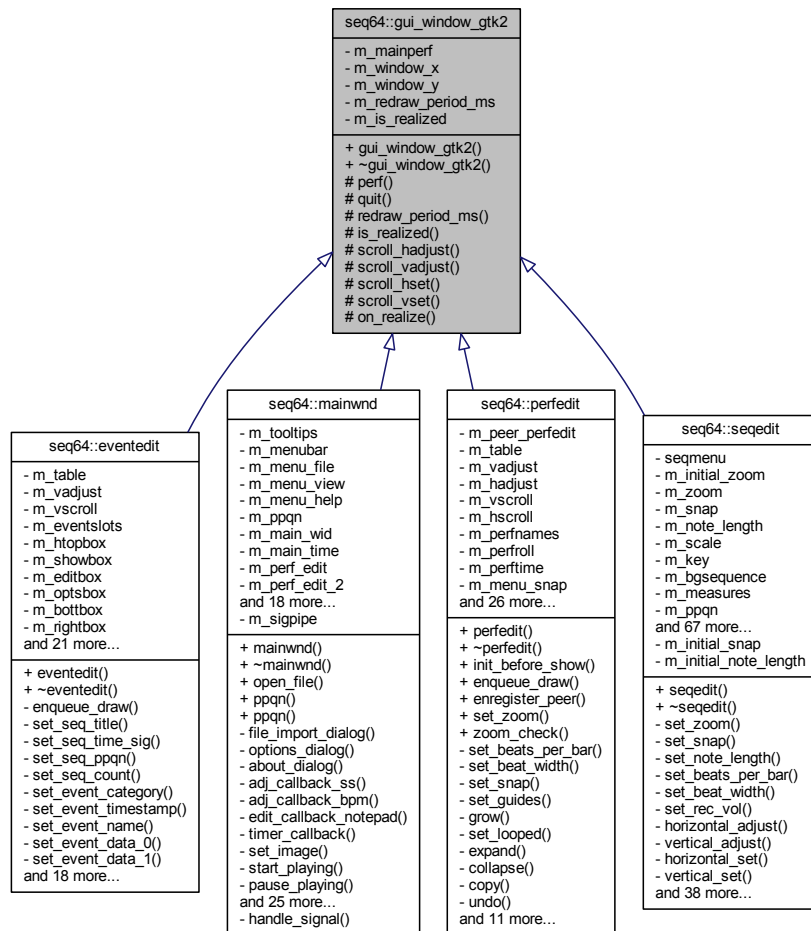
12.20.5.15 `Color seq64::gui_palette_gtk2::m_bg_color` [private]

12.20.5.16 `Color seq64::gui_palette_gtk2::m_fg_color` [private]

12.21 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.

12.21.1 Constructor & Destructor Documentation

12.21.1.1 `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.

12.21.1.2 `seq64::gui_window_gtk2::~~gui_window_gtk2 ()` [virtual]

12.21.2 Member Function Documentation

12.21.2.1 `perform& seq64::gui_window_gtk2::perf () [inline], [protected]`

12.21.2.2 `virtual void seq64::gui_window_gtk2::quit () [inline], [protected], [virtual]`

12.21.2.3 `int seq64::gui_window_gtk2::redraw_period_ms () const [inline], [protected]`

12.21.2.4 `bool seq64::gui_window_gtk2::is_realized () const [inline], [protected]`

12.21.2.5 `void seq64::gui_window_gtk2::scroll_hadjust (Gtk::Adjustment & hadjust, double step) [protected]`

This function is currently duplicated in the [gui_drawingarea_gtk2](#) and [gui_window_gtk2](#) modules.

Parameters

<i>hadjust</i>	Provides a reference to the adjustment object to be adjusted.
<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. If negative, the adjustment is leftward. If positive, the adjustment is rightward. It can be the value of <code>m_hadjust->get_step_increment()</code> , or provided especially to keep up with the progress bar.

12.21.2.6 `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.

12.21.2.7 `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.

12.21.2.8 `void seq64::gui_window_gtk2::scroll_vset (Gtk::Adjustment & vadjust, 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 vertical scrollbar.

12.21.2.9 `void seq64::gui_window_gtk2::on_realize () [protected]`

12.21.3 Field Documentation

12.21.3.1 `perform& seq64::gui_window_gtk2::m_mainperf` `[private]`

And a whole lot more than that.

12.21.3.2 `int seq64::gui_window_gtk2::m_window_x` `[private]`

Could make this constant, but some windows are resizable. The width of the window.

12.21.3.3 `int seq64::gui_window_gtk2::m_window_y` `[private]`

12.21.3.4 `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.

12.21.3.5 `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.

12.22 seq64::jack_assistant Class Reference

This class provides the performance mode JACK support.

Public Member Functions

- `jack_assistant` (`perform` & `parent`, `int 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.
- `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*
- int `get_beats_per_minute` () const
- *'Getter' function for member m_beats_per_minute*
- void `set_beats_per_minute` (int bpmminute)
- *'Setter' function for member m_beats_per_minute For the future, changing the BPM (beats/minute) internally.*
- 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.*
- void `start` ()
- *If JACK is supported, starts the JACK transport.*
- void `stop` ()
- *If JACK is supported, stops the JACK transport.*
- void `position` (bool to_left_tick, bool relocate=false)
- *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*

Private Member Functions

- void `set_jack_running` (bool flag)
- *'Setter' function for member m_jack_running*
- jack_client_t * `client` () const
- *'Getter' function for member m_jack_client*
- 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*
- bool `info_message` (const std::string &msg)
- *Common-code for console messages.*
- bool `error_message` (const std::string &msg)
- *Common-code for error messages.*
- jack_client_t * `client_open` (const std::string &clientname)
- *A more full-featured initialization for a JACK client, which is meant to be called by the `init()` function.*
- void `get_jack_client_info` ()
- *Tries to obtain the best information on the JACK client and the UUID assigned to this client.*
- void `show_statuses` (unsigned bits)
- *Loops through the full set of JACK bits, showing the information for any bits that are set in the given parameter.*
- 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.*

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_jsession_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.
- [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.
- [int](#) [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_process_callback](#) (jack_nframes_t nframes, void *arg)
- void [jack_shutdown_callback](#) (void *arg)
This callback is to shutdown 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.
- void [jack_session_callback](#) (jack_session_event_t *ev, void *arg)
Set the [m_jsession_ev](#) (event) value of the perform object.

12.22.1 Constructor & Destructor Documentation

- 12.22.1.1 **seq64::jack_assistant::jack_assistant (perform & parent, int bpmminute = SEQ64_DEFAULT_BPM, int ppqn = SEQ64_USE_DEFAULT_PPQN, int bpm = 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>bpm</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.

- 12.22.1.2 **seq64::jack_assistant::~~jack_assistant ()**

The perform object currently calls [jack_assistant::deinit\(\)](#), but that call could be made here instead.

12.22.2 Member Function Documentation

- 12.22.2.1 **perform& seq64::jack_assistant::parent ()** `[inline]`
- 12.22.2.2 **bool seq64::jack_assistant::is_running ()** `const [inline]`
- 12.22.2.3 **bool seq64::jack_assistant::is_master ()** `const [inline]`
- 12.22.2.4 **int seq64::jack_assistant::get_ppqn ()** `const [inline]`
- 12.22.2.5 **int seq64::jack_assistant::get_beat_width ()** `const [inline]`
- 12.22.2.6 **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.
-----------	---

12.22.2.7 `int seq64::jack_assistant::get_beats_per_measure () const [inline]`

12.22.2.8 `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.
------------	--

12.22.2.9 `int seq64::jack_assistant::get_beats_per_minute () const [inline]`

12.22.2.10 `void seq64::jack_assistant::set_beats_per_minute (int 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.
------------------	---

12.22.2.11 `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).
```

Returns

Returns true if JACK is now considered to be running (or if it was already running.)

12.22.2.12 `bool seq64::jack_assistant::deinit ()`

Returns

Returns the value of `m_jack_running`, which should be false.

12.22.2.13 `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.

12.22.2.14 `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.
```

12.22.2.15 `void seq64::jack_assistant::stop ()`

This function assumes that `m_jack_client` is not null, if `m_jack_running` is true.

12.22.2.16 void seq64::jack_assistant::position (bool *to_left_tick*, bool *relocate* = false)

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.

These repositions reset the progress bars. We don't always want that, it should be an option. TODO.

jack_transport_reposition():

Requests a new transport position. 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.

It's pos parameter provides the requested new transport position. Fill pos->valid to specify which fields should be taken into account. If you mark a set of fields as valid, you are expected to fill them all. Note that "frame" is always assumed, and generally needs to be set:

<http://comments.gmane.org/gmane.comp.audio.jackit/18705>

Returns 0 if a valid request, EINVAL if the position structure is rejected.

Warning

A lot of this code is effectively disabled by an early return statement.

Parameters

<i>to_left_tick</i>	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.
<i>relocate</i>	If true (it defaults to false), then we allow the relocation of the JACK transport to the current_tick or the left tick, rather than to frame 0. EXPERIMENTAL, enables dead code from seq24. Seems to work if set to true when we are the JACK Master. Enabling this code makes "klick -j -P" work, after a fashion. It clicks, but at a way too rapid rate.

12.22.2.17 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.

Parameters

<i>pad</i>	Provide a JACK scratchpad for sharing certain items between the perform object and the jack_assistant object.
------------	---

Returns

Returns true if JACK is running.

12.22.2.18 `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.
-------------	---------------------------------

12.22.2.19 `double seq64::jack_assistant::get_jack_tick () const [inline]`

12.22.2.20 `const jack_position_t& seq64::jack_assistant::get_jack_pos () const [inline]`

12.22.2.21 `void seq64::jack_assistant::set_jack_running (bool flag) [inline], [private]`

Parameters

<i>flag</i>	Provides the is-running value to set.
-------------	---------------------------------------

12.22.2.22 `jack_client_t* seq64::jack_assistant::client () const [inline], [private]`

12.22.2.23 `const std::string& seq64::jack_assistant::client_name () const [inline], [private]`

12.22.2.24 `const std::string& seq64::jack_assistant::client_uuid () const [inline], [private]`

12.22.2.25 `bool seq64::jack_assistant::info_message (const std::string & msg) [private]`

Adds markers and a newline.

Parameters

<i>msg</i>	The message to print, sans the newline.
------------	---

Returns

Returns true.

12.22.2.26 `bool seq64::jack_assistant::error_message (const std::string & msg) [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.

12.22.2.27 `jack_client_t * seq64::jack_assistant::client_open (const std::string & clientname) [private]`

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

Only the first is used at present.

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.
-------------------	--

Returns

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

12.22.2.28 `void seq64::jack_assistant::get_jack_client_info () [private]`

Sets `m_jack_client_name` and `m_jack_client_info` as side-effects.

12.22.2.29 `void seq64::jack_assistant::show_statuses (unsigned bits) [private]`

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.
-------------	---

12.22.2.30 `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:

```

nnnnn frame B:B:T N/D TPB BPM BBT
^   ^   ^   ^   ^   ^   ^
|   |   |   |   |   |   |
|   |   |   |   |   |   |----- bbt_offset (frame), even if invalid
|   |   |   |   |   |   |----- beats_per_minute
|   |   |   |   |   |   |----- ticks_per_beat (PPQN * 10?)
|   |   |   |   |   |   |----- beat_type (denominator)
|   |   |   |   |   |   |----- beats_per_bar (numerator)
|   |   |   |   |   |   |----- bar : beat : tick
|   |   |   |   |   |   |----- frame (number)
|   |   |   |   |   |   |----- the "valid" bits
-----

```

The "valid" field is shown as bits in the same bit order as shown here, but represented as a five-character string, "nnnnn", n = 0 or 1:

```

JackVideoFrameOffset = 0x100
JackAudioVideoRatio  = 0x080
JackBBTFrameOffset   = 0x040
JackPositionTimecode = 0x020
JackPositionBBT       = 0x010

```

We care most about nnnnn = "00101" in our experiments (the most common output will be "00001"). And we don't worry about non-integer measurements... we truncate them to integers. Change the output format if you want to play with non-Western timings.

Parameters

<i>pos</i>	The JACK position structure to dump.
------------	--------------------------------------

```

12.22.231 int seq64::jack_assistant::sync ( jack_transport_state_t state = (jack_transport_state_t) (-1) )
[private]

```

Sequencer64 is not a slow-sync client, so that callback is not really needed, but we probably need this sub-function here to start out with the right values for interacting with JACK.

Note the call to `jack_transport_query()`. This call is *not* in `seq24`, but seems to be needed in `sequencer64` because we put `m_jack_pos` in the initializer list, which sets all its fields to 0. `Seq24` accesses `m_jack_pos` before it ever gets set, but its fields have values. These values are bogus, but are consistent from run to run on my computer, and allow `seq24` to follow another JACK Master, on some computers. It explains why people had different experiences with JACK sync.

If we explicitly call `jack_transport_query()` here, without changing the `state` parameter, then `sequencer64` also can follow another JACK Master. (CURRENTLY BUGGY!)

Note that we should consider massaging the following `jack_position_t` members to set them to 0 (or 0.0) if less than 1.0 or 0.5:

- `bar_start_tick`
- `ticks_per_beat`
- `beats_per_minute`
- `frame_time`
- `next_time`
- `audio_frames_per_video_frame`

Also, why does `bbt_offset` start at 2128362496?

Parameters

<i>state</i>	The JACK transport state to be set.
--------------	-------------------------------------

12.22.2.32 `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.
--------------------	--

12.22.3 Friends And Related Function Documentation

12.22.3.1 `int jack_process_callback (jack_nframes_t nframes, void * arg)` [friend]

12.22.3.2 `void jack_shutdown_callback (void * arg)` [friend]

Parameters

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

12.22.3.3 `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.

Parameters

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

Returns

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

12.22.3.4 `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.
```

Parameters

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

12.22.3.5 `void jack_session_callback (jack_session_event_t * ev, void * arg)` [friend]

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

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

Parameters

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

12.22.4 Field Documentation

12.22.4.1 `jack_status_pair_t seq64::jack_assistant::sm_status_pairs` `[static], [private]`

Provides a list of JACK status bits, and a brief string to explain the status bit.

Terminated by a 0 value and an empty string.

12.22.4.2 `perform& seq64::jack_assistant::m_jack_parent` `[private]`

12.22.4.3 `jack_client_t* seq64::jack_assistant::m_jack_client` `[private]`

12.22.4.4 `std::string seq64::jack_assistant::m_jack_client_name` `[private]`

We might show this in the user-interface at some point.

12.22.4.5 `std::string seq64::jack_assistant::m_jack_client_uuid` `[private]`

We might show this in the user-interface at some point.

12.22.4.6 `jack_nframes_t seq64::jack_assistant::m_jack_frame_current` `[private]`

12.22.4.7 `jack_nframes_t seq64::jack_assistant::m_jack_frame_last` `[private]`

Also used in incrementing `m_jack_tick`.

12.22.4.8 `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.

12.22.4.9 `jack_transport_state_t seq64::jack_assistant::m_jack_transport_state` `[private]`

Common values are `JackTransportStopped`, `JackTransportRolling`, and `JackTransportLooping`.

12.22.4.10 `jack_transport_state_t seq64::jack_assistant::m_jack_transport_state_last` [private]

12.22.4.11 `double seq64::jack_assistant::m_jack_tick` [private]

12.22.4.12 `jack_session_event_t* seq64::jack_assistant::m_session_ev` [private]

Used in the [session_event\(\)](#) function.

12.22.4.13 `bool seq64::jack_assistant::m_jack_running` [private]

12.22.4.14 `bool seq64::jack_assistant::m_jack_master` [private]

12.22.4.15 `int seq64::jack_assistant::m_ppqn` [private]

It is used for calculating ticks/beat (pulses/beat) and for setting the tick position.

12.22.4.16 `int seq64::jack_assistant::m_beats_per_measure` [private]

12.22.4.17 `int seq64::jack_assistant::m_beat_width` [private]

12.22.4.18 `int seq64::jack_assistant::m_beats_per_minute` [private]

12.23 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_last`
Keeps track of position?

12.23.1 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.

12.23.2 Field Documentation

12.23.2.1 `double seq64::jack_scratchpad::js_current_tick`

12.23.2.2 `double seq64::jack_scratchpad::js_total_tick`

12.23.2.3 `double seq64::jack_scratchpad::js_clock_tick`

12.23.2.4 `bool seq64::jack_scratchpad::js_jack_stopped`

12.23.2.5 `bool seq64::jack_scratchpad::js_dumping`

12.23.2.6 `bool seq64::jack_scratchpad::js_init_clock`

12.23.2.7 `bool seq64::jack_scratchpad::js_looping`

12.23.2.8 `bool seq64::jack_scratchpad::js_playback_mode`

12.23.2.9 `double seq64::jack_scratchpad::js_ticks_converted_last`

12.24 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.

12.24.1 Field Documentation

12.24.1.1 `unsigned seq64::jack_status_pair_t::jf_bit`

12.24.1.2 `std::string seq64::jack_status_pair_t::jf_meaning`

12.25 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

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](#)

12.25.1 Member Enumeration Documentation

12.25.1.1 `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.

12.25.2 Constructor & Destructor Documentation

12.25.2.1 `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.

12.25.3 Member Function Documentation

12.25.3.1 `void seq64::keybindentry::set (unsigned int val)`

Then we call `set_text(buf)`. The `set_width_char()` function is then called.

12.25.3.2 `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()`.

12.25.4 Friends And Related Function Documentation

12.25.4.1 `friend class options` `[friend]`

12.25.5 Field Documentation

12.25.5.1 `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.

12.25.5.2 `type seq64::keybindentry::m_type` `[private]`

12.25.5.3 `perform* seq64::keybindentry::m_perf` `[private]`

12.25.5.4 `long seq64::keybindentry::m_slot` `[private]`

(This item should be changed to unsigned int, though.)

12.26 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 `screensset_up () const`
'Getter' function for member m_key_screensset_up
- void `screensset_up (unsigned int x)`
'Setter' function for member m_key_screensset_up
- unsigned int `screensset_dn () const`
'Getter' function for member m_key_screensset_dn
- void `screensset_dn (unsigned int x)`
'Setter' function for member m_key_screensset_dn
- unsigned int `set_playing_screensset () const`
'Getter' function for member m_key_playing_screensset
- void `set_playing_screensset (unsigned int x)`
'Setter' function for member m_key_playing_screensset
- 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*
- 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_sequency_number*
- void [show_ui_sequence_number](#) (bool flag)
- 'Setter' function for member m_key_show_ui_sequency_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_screenset_up](#) ()
'Getter' function for member m_key_screenset_up
- unsigned int * [at_screenset_dn](#) ()
'Getter' function for member m_key_screenset_dn
- unsigned int * [at_set_playing_screenset](#) ()
'Getter' function for member m_key_playing_screenset
- 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_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_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](#)

12.26.1 Detailed Description

It provides a way a mapping keystrokes to sequencer actions and song settings.

12.26.2 Member Typedef Documentation

12.26.2.1 `typedef std::map<unsigned int, long> seq64::keys_perform::SlotMap` `[protected]`

12.26.2.2 `typedef std::map<long, unsigned int> seq64::keys_perform::RevSlotMap` `[protected]`

12.26.3 Constructor & Destructor Documentation

12.26.3.1 `seq64::keys_perform::keys_perform ()`

12.26.3.2 `seq64::keys_perform::~~keys_perform ()` `[virtual]`

Finally, any active patterns/sequences are deleted.

12.26.4 Member Function Documentation

12.26.4.1 `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.
------------	--

12.26.4.2 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.
------------	--

12.26.4.3 unsigned int seq64::keys_perform::bpm_up () const [inline]

12.26.4.4 void seq64::keys_perform::bpm_up (unsigned int *x*) [inline]

Parameters

<i>x</i>	The key value to assign to the operation.
----------	---

12.26.4.5 unsigned int seq64::keys_perform::bpm_dn () const [inline]

12.26.4.6 void seq64::keys_perform::bpm_dn (unsigned int *x*) [inline]

Parameters

<i>x</i>	The key value to assign to the operation.
----------	---

12.26.4.7 unsigned int seq64::keys_perform::replace () const [inline]

12.26.4.8 void seq64::keys_perform::replace (unsigned int *x*) [inline]

Parameters

<i>x</i>	The key value to assign to the operation.
----------	---

12.26.4.9 unsigned int seq64::keys_perform::queue () const [inline]

12.26.4.10 void seq64::keys_perform::queue (unsigned int *x*) [inline]

Parameters

<i>x</i>	The key value to assign to the operation.
----------	---

12.26.4.11 `unsigned int seq64::keys_perform::keep_queue () const` `[inline]`

12.26.4.12 `void seq64::keys_perform::keep_queue (unsigned int x)` `[inline]`

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

12.26.4.13 `unsigned int seq64::keys_perform::snapshot_1 () const` `[inline]`

12.26.4.14 `void seq64::keys_perform::snapshot_1 (unsigned int x)` `[inline]`

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

12.26.4.15 `unsigned int seq64::keys_perform::snapshot_2 () const` `[inline]`

12.26.4.16 `void seq64::keys_perform::snapshot_2 (unsigned int x)` `[inline]`

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

12.26.4.17 `unsigned int seq64::keys_perform::screenset_up () const` `[inline]`

12.26.4.18 `void seq64::keys_perform::screenset_up (unsigned int x)` `[inline]`

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

12.26.4.19 `unsigned int seq64::keys_perform::screenset_dn () const` `[inline]`

12.26.4.20 `void seq64::keys_perform::screenset_dn (unsigned int x)` `[inline]`

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

12.26.4.21 `unsigned int seq64::keys_perform::set_playing_screenset () const` `[inline]`

12.26.4.22 `void seq64::keys_perform::set_playing_screenset (unsigned int x)` `[inline]`

Parameters

x	The key value to assign to the operation.
---	---

12.26.4.23 `unsigned int seq64::keys_perform::group_on () const` `[inline]`

12.26.4.24 `void seq64::keys_perform::group_on (unsigned int x)` `[inline]`

Parameters

x	The key value to assign to the operation.
---	---

12.26.4.25 `unsigned int seq64::keys_perform::group_off () const` `[inline]`

12.26.4.26 `void seq64::keys_perform::group_off (unsigned int x)` `[inline]`

Parameters

x	The key value to assign to the operation.
---	---

12.26.4.27 `unsigned int seq64::keys_perform::group_learn () const` `[inline]`

12.26.4.28 `void seq64::keys_perform::group_learn (unsigned int x)` `[inline]`

Parameters

x	The key value to assign to the operation.
---	---

12.26.4.29 `unsigned int seq64::keys_perform::start () const` `[inline]`

12.26.4.30 `void seq64::keys_perform::start (unsigned int x)` `[inline]`

Parameters

x	The key value to assign to the operation.
---	---

12.26.4.31 `unsigned int seq64::keys_perform::pause () const` `[inline]`

12.26.4.32 `void seq64::keys_perform::pause (unsigned int x)` `[inline]`

Parameters

x	The key value to assign to the operation.
---	---

12.26.4.33 `unsigned int seq64::keys_perform::pattern_edit () const` `[inline]`

12.26.4.34 `void seq64::keys_perform::pattern_edit (unsigned int x)` `[inline]`

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

12.26.4.35 `unsigned int seq64::keys_perform::event_edit () const` `[inline]`

12.26.4.36 `void seq64::keys_perform::event_edit (unsigned int x)` `[inline]`

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

12.26.4.37 `unsigned int seq64::keys_perform::stop () const` `[inline]`

12.26.4.38 `void seq64::keys_perform::stop (unsigned int x)` `[inline]`

Parameters

<code>x</code>	The key value to assign to the operation.
----------------	---

12.26.4.39 `bool seq64::keys_perform::show_ui_sequence_key () const` `[inline]`

Used in mainwid, options, optionsfile, userfile, and perform.

12.26.4.40 `void seq64::keys_perform::show_ui_sequence_key (bool flag)` `[inline]`

Parameters

<code><i>flag</i></code>	The flag for showing the sequence key characters in each pattern slot.
--------------------------	--

12.26.4.41 `bool seq64::keys_perform::show_ui_sequence_number () const` `[inline]`

Used in mainwid, options, optionsfile, userfile, and perform.

12.26.4.42 `void seq64::keys_perform::show_ui_sequence_number (bool flag)` `[inline]`

Parameters

<code><i>flag</i></code>	The flag for showing the sequence number in each pattern slot.
--------------------------	--

12.26.4.43 SlotMap& seq64::keys_perform::get_key_events () [inline]

12.26.4.44 SlotMap& seq64::keys_perform::get_key_groups () [inline]

12.26.4.45 RevSlotMap& seq64::keys_perform::get_key_events_rev () [inline]

12.26.4.46 RevSlotMap& seq64::keys_perform::get_key_groups_rev () [inline]

12.26.4.47 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!
---------------	---

12.26.4.48 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.
----------------	--

12.26.4.49 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.
-----------------	---

12.26.4.50 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.
----------------	--

12.26.4.51 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](#).

12.26.4.52 `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](#).

12.26.4.53 `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](#).

12.26.4.54 `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.

12.26.4.55 `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.

12.26.4.56 `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.

12.26.4.57 `unsigned int* seq64::keys_perform::at_bpm_dn () [inline],[protected]`

Address getter for the bpm_dn operation.

12.26.4.58 `unsigned int* seq64::keys_perform::at_replace () [inline], [protected]`

Address getter for the replace operation.

12.26.4.59 `unsigned int* seq64::keys_perform::at_queue () [inline], [protected]`

Address getter for the queue operation.

12.26.4.60 `unsigned int* seq64::keys_perform::at_keep_queue () [inline], [protected]`

Address getter for the keep_queue operation.

12.26.4.61 `unsigned int* seq64::keys_perform::at_snapshot_1 () [inline], [protected]`

Address getter for the snapshot_1 operation.

12.26.4.62 `unsigned int* seq64::keys_perform::at_snapshot_2 () [inline], [protected]`

Address getter for the snapshot_2 operation.

12.26.4.63 `unsigned int* seq64::keys_perform::at_screenset_up () [inline], [protected]`

Address getter for the screenset_up operation.

12.26.4.64 `unsigned int* seq64::keys_perform::at_screenset_dn () [inline], [protected]`

Address getter for the screenset_dn operation.

12.26.4.65 `unsigned int* seq64::keys_perform::at_set_playing_screenset () [inline], [protected]`

Address getter for the set_playing_screenset operation.

12.26.4.66 `unsigned int* seq64::keys_perform::at_group_on () [inline], [protected]`

Address getter for the group_on operation.

12.26.4.67 `unsigned int* seq64::keys_perform::at_group_off () [inline], [protected]`

Address getter for the group_off operation.

12.26.4.68 `unsigned int* seq64::keys_perform::at_group_learn () [inline],[protected]`

Address getter for the group_learn operation.

12.26.4.69 `unsigned int* seq64::keys_perform::at_start () [inline],[protected]`

Address getter for the start operation.

12.26.4.70 `unsigned int* seq64::keys_perform::at_pause () [inline],[protected]`

Address getter for the pause operation.

12.26.4.71 `unsigned int* seq64::keys_perform::at_pattern_edit () [inline],[protected]`

Address getter for the pattern edit operation.

12.26.4.72 `unsigned int* seq64::keys_perform::at_event_edit () [inline],[protected]`

Address getter for the event edit operation.

12.26.4.73 `unsigned int* seq64::keys_perform::at_stop () [inline],[protected]`

Address getter for the stop operation.

12.26.4.74 `bool* seq64::keys_perform::at_show_ui_sequence_key () [inline],[protected]`

Address getter for the show_ui_sequence_key value.

12.26.4.75 `bool* seq64::keys_perform::at_show_ui_sequence_number () [inline],[protected]`

Address getter for the show_ui_sequence_number value.

12.26.5 Friends And Related Function Documentation

12.26.5.1 `friend class options [friend]`

12.26.5.2 `friend class perform [friend]`

12.26.5.3 `friend class optionsfile [friend]`

12.26.6 Field Documentation

12.26.6.1 `bool seq64::keys_perform::m_key_show_ui_sequence_key [private]`

12.26.6.2 `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.

12.26.6.3 SlotMap seq64::keys_perform::m_key_events [private]

Do not access directly, use the set/lookup functions declared below.

12.26.6.4 SlotMap seq64::keys_perform::m_key_groups [private]

Do not access directly, use the set/lookup functions declared below.

12.26.6.5 RevSlotMap seq64::keys_perform::m_key_events_rev [private]

Do not access directly, use the set/lookup functions declared below.

12.26.6.6 RevSlotMap seq64::keys_perform::m_key_groups_rev [private]

Do not access directly, use the set/lookup functions declared below.

12.26.6.7 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!!!

12.26.6.8 unsigned int seq64::keys_perform::m_key_bpm_dn [private]

12.26.6.9 unsigned int seq64::keys_perform::m_key_replace [private]

12.26.6.10 unsigned int seq64::keys_perform::m_key_queue [private]

12.26.6.11 unsigned int seq64::keys_perform::m_key_keep_queue [private]

12.26.6.12 unsigned int seq64::keys_perform::m_key_snapshot_1 [private]

12.26.6.13 unsigned int seq64::keys_perform::m_key_snapshot_2 [private]

12.26.6.14 unsigned int seq64::keys_perform::m_key_screenset_up [private]

12.26.6.15 unsigned int seq64::keys_perform::m_key_screenset_dn [private]

12.26.6.16 unsigned int seq64::keys_perform::m_key_set_playing_screenset [private]

12.26.6.17 unsigned int seq64::keys_perform::m_key_group_on [private]

12.26.6.18 unsigned int seq64::keys_perform::m_key_group_off [private]

12.26.6.19 unsigned int seq64::keys_perform::m_key_group_learn [private]

12.26.6.20 unsigned int seq64::keys_perform::m_key_start [private]

12.26.6.21 unsigned int seq64::keys_perform::m_key_pause [private]

12.26.6.22 unsigned int seq64::keys_perform::m_key_pattern_edit [private]

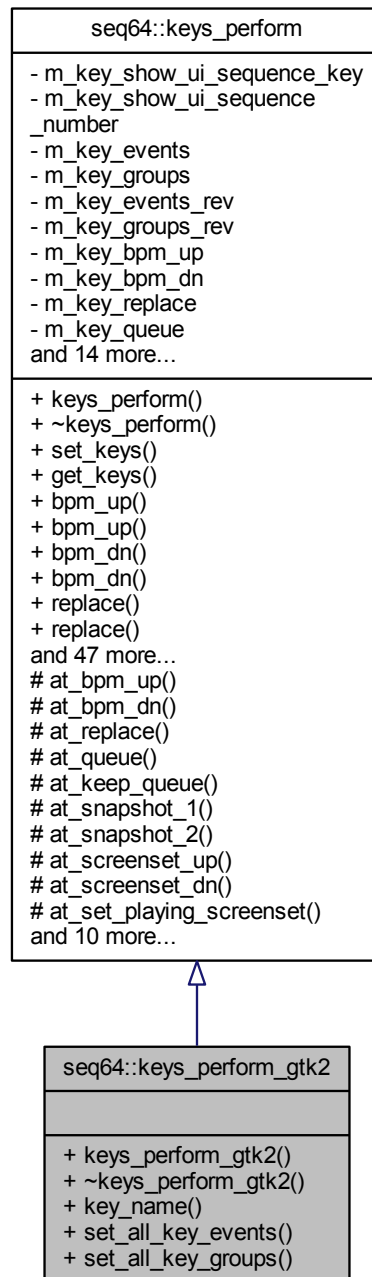
12.26.6.23 unsigned int seq64::keys_perform::m_key_event_edit [private]

12.26.6.24 unsigned int seq64::keys_perform::m_key_stop [private]

12.27 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

12.27.1 Detailed Description

It has way too many data members, many of the public. Might be ripe for refactoring.

12.27.2 Constructor & Destructor Documentation

12.27.2.1 `seq64::keys_perform_gtk2::keys_perform_gtk2 ()`

12.27.2.2 `seq64::keys_perform_gtk2::~~keys_perform_gtk2 ()` `[virtual]`

No action.

12.27.3 Member Function Documentation

12.27.3.1 `virtual std::string seq64::keys_perform_gtk2::key_name (unsigned int key) const` `[inline]`, `[virtual]`

Reimplemented from [seq64::keys_perform](#).

12.27.3.2 `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](#).

12.27.3.3 `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](#).

12.28 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_pause](#)

12.28.1 Field Documentation

12.28.1.1 unsigned int seq64::keys_perform_transfer::kpt_bpm_up

12.28.1.2 unsigned int seq64::keys_perform_transfer::kpt_bpm_dn

12.28.1.3 unsigned int seq64::keys_perform_transfer::kpt_screenset_up

12.28.1.4 unsigned int seq64::keys_perform_transfer::kpt_screenset_dn

12.28.1.5 unsigned int seq64::keys_perform_transfer::kpt_set_playing_screenset

12.28.1.6 unsigned int seq64::keys_perform_transfer::kpt_group_on

12.28.1.7 unsigned int seq64::keys_perform_transfer::kpt_group_off

12.28.1.8 unsigned int seq64::keys_perform_transfer::kpt_group_learn

12.28.1.9 unsigned int seq64::keys_perform_transfer::kpt_replace

12.28.1.10 unsigned int seq64::keys_perform_transfer::kpt_queue

12.28.1.11 unsigned int seq64::keys_perform_transfer::kpt_keep_queue

12.28.1.12 unsigned int seq64::keys_perform_transfer::kpt_snapshot_1

- 12.28.1.13 unsigned int seq64::keys_perform_transfer::kpt_snapshot_2
- 12.28.1.14 unsigned int seq64::keys_perform_transfer::kpt_start
- 12.28.1.15 unsigned int seq64::keys_perform_transfer::kpt_stop
- 12.28.1.16 bool seq64::keys_perform_transfer::kpt_show_ui_sequence_key
- 12.28.1.17 bool seq64::keys_perform_transfer::kpt_show_ui_sequence_number
- 12.28.1.18 unsigned int seq64::keys_perform_transfer::kpt_pattern_edit
- 12.28.1.19 unsigned int seq64::keys_perform_transfer::kpt_event_edit
- 12.28.1.20 unsigned int seq64::keys_perform_transfer::kpt_pause

12.29 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_MODKEY](#)))
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)
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.

12.29.1 Detailed Description

Useful in passing more generic events to non-GUI classes.

12.29.2 Constructor & Destructor Documentation

12.29.2.1 `seq64::keystroke::keystroke ()`

12.29.2.2 `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> .

12.29.2.3 `seq64::keystroke::keystroke (const keystroke & rhs)`

Parameters

<i>rhs</i>	The object to be copied.
------------	--------------------------

12.29.3 Member Function Documentation

12.29.3.1 `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.

12.29.3.2 `bool seq64::keystroke::is_press () const [inline]`

12.29.3.3 `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

12.29.3.4 `bool seq64::keystroke::is (unsigned int ch) [inline]`

Parameters

<i>ch</i>	The character to be tested.
-----------	-----------------------------

Returns

Returns true if `m_key == ch`.

12.29.3.5 `bool seq64::keystroke::is_delete () const [inline]`

12.29.3.6 `unsigned int seq64::keystroke::key () const [inline]`

12.29.3.7 `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.

12.29.3.8 `seq_modifier_t seq64::keystroke::modifier () const` `[inline]`

12.29.3.9 `bool seq64::keystroke::mod_control () const` `[inline]`

12.29.3.10 `bool seq64::keystroke::mod_control_shift () const` `[inline]`

12.29.3.11 `bool seq64::keystroke::mod_super () const` `[inline]`

12.29.4 Field Documentation

12.29.4.1 `bool seq64::keystroke::m_is_press` `[private]`

See the SEQ64_KEYSTROKE_PRESS and SEQ64_KEYSTROKE_RELEASE readability macros.

12.29.4.2 `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.

12.29.4.3 `seq_modifier_t seq64::keystroke::m_modifier` `[private]`

Note that SEQ64_NO_MASK is our word for 0, meaning "no modifier".

12.30 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.

12.30.1 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.

12.30.2 Constructor & Destructor Documentation

12.30.2.1 `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.

12.30.3 Member Function Documentation

12.30.3.1 `void seq64::lash::set_alsa_client_id (int id)`

/param id The ALSA client ID to be set.

12.30.3.2 `void seq64::lash::start ()`

12.30.3.3 `bool seq64::lash::process_events ()`

Returns

Always returns true.

12.30.3.4 `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.

12.30.3.5 `void seq64::lash::handle_event (lash_event_t* ev) [private]`

Parameters

<code><i>ev</i></code>	Provides the event to be handled.
------------------------	-----------------------------------

12.30.3.6 `void seq64::lash::handle_config (lash_config_t* conf) [private]`

Currently incomplete.

Parameters

<code><i>conf</i></code>	Provides the configuration item to handle.
--------------------------	--

12.30.4 Field Documentation

12.30.4.1 `perform& seq64::lash::m_perform [private]`

12.30.4.2 `lash_client_t* seq64::lash::m_client [private]`

12.30.4.3 `lash_args_t* seq64::lash::m_lash_args [private]`

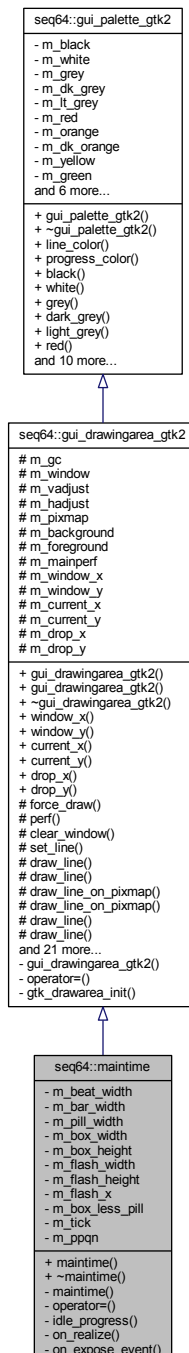
12.30.4.4 `bool seq64::lash::m_is_lash_supported [private]`

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.

12.31 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

12.31.1 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.

12.31.2 Constructor & Destructor Documentation

12.31.2.1 `seq64::maintime::maintime (const maintime &) [private]`

12.31.2.2 `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.

12.31.2.3 `virtual seq64::maintime::~~maintime () [inline],[virtual]`

12.31.3 Member Function Documentation

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

12.31.3.2 `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 mainwnd() , in its timer callback.
--------------	--

Returns

Always returns 1 (it used to return "true!").

12.31.3.3 `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.

12.31.3.4 `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.)

12.31.4 Friends And Related Function Documentation

12.31.4.1 `friend class mainwnd [friend]`

12.31.5 Field Documentation

12.31.5.1 `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\(\)](#).

12.31.5.2 `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()`.

12.31.5.3 `const int seq64::maintime::m_pill_width` [private]

12.31.5.4 `const int seq64::maintime::m_box_width` [private]

This item absolutely depends on the main window being non-resizable.

12.31.5.5 `const int seq64::maintime::m_box_height` [private]

This item absolutely depends on the main window being non-resizable.

12.31.5.6 `const int seq64::maintime::m_flash_width` [private]

Just a bit smaller than `m_box_width`.

12.31.5.7 `const int seq64::maintime::m_flash_height` [private]

Just a bit smaller than `m_box_width`.

12.31.5.8 `const int seq64::maintime::m_flash_x` [private]

12.31.5.9 `const int seq64::maintime::m_box_less_pill` [private]

12.31.5.10 `midipulse seq64::maintime::m_tick` [private]

It might actually be useful after all. And the overhead is tiny.

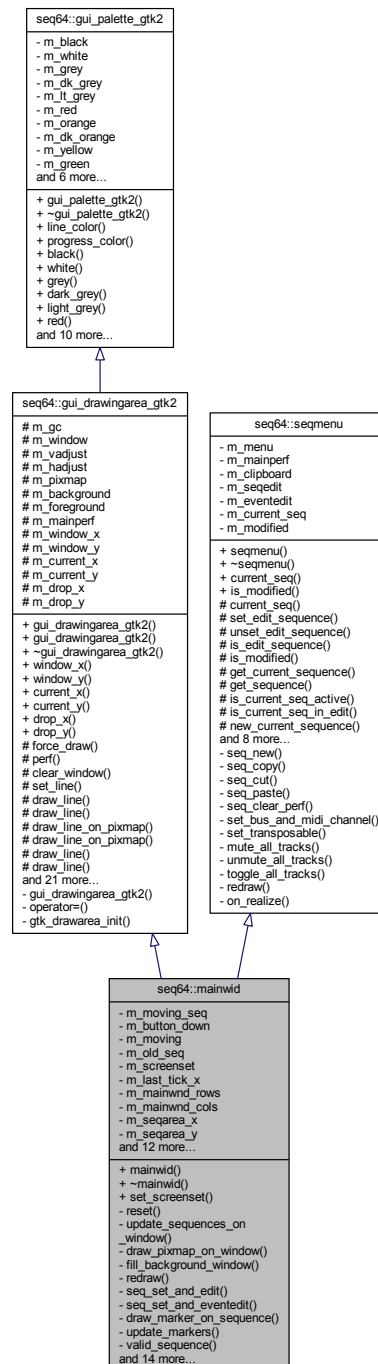
12.31.5.11 `int seq64::maintime::m_ppqn` [private]

While this is effectively a constant for the duration of a tune, it might change as different tunes are loaded.

12.32 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 coordinates in the Patterns Panel to a sequence number.

- int `timeout ()`

Provides a stock callback, because some kind of callback is needed.

- void `calculate_base_sizes` (int seq, int &basex, int &basey)

Provides a way to calculate the base x and y size values for the pattern map.

- void `select_fg_bg_colors` (int seqnum)

Picks the foreground and background colors based on the sequence in edit and the SEQ64_EDIT_SEQUENCE_↔ HIGHLIGHT macro.

- void `on_realize ()`

For this GTK callback, on realization of window, initialize the shiz.

- bool `on_expose_event` (GdkEventExpose *ev)

Implements the GTK expose event callback.

- bool [on_button_press_event](#) (GdkEventButton *ev)
Handles a press of a mouse button in one of the sequence/pattern slots.
- bool [on_button_release_event](#) (GdkEventButton *ev)
Handles a release of a mouse button.
- bool [on_motion_notify_event](#) (GdkEventMotion *p0)
Handle the motion of the mouse if a mouse button is down and in another sequence and if the current sequence is not in edit mode.
- bool [on_focus_in_event](#) (GdkEventFocus *)
Handles an on-focus event.
- bool [on_focus_out_event](#) (GdkEventFocus *)
Handles an out-of-focus event.

Private Attributes

- [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.
- 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

12.32.1 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.

12.32.2 Constructor & Destructor Documentation

12.32.2.1 `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.
----------	--

12.32.2.2 `seq64::mainwid::~~mainwid () [virtual]`

12.32.3 Member Function Documentation

12.32.3.1 `void seq64::mainwid::set_screenset (int ss, bool setperf = false)`

The clamping algorithm for the `screenset` 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

<code>ss</code>	Provides the screen-set number to set.
-----------------	--

12.32.3.2 `void seq64::mainwnd::reset () [inline], [private]`

12.32.3.3 `void seq64::mainwnd::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.

12.32.3.4 `void seq64::mainwnd::draw_pixmap_on_window () [inline], [private]`

12.32.3.5 `void seq64::mainwnd::fill_background_window () [inline], [private]`

12.32.3.6 `void seq64::mainwnd::redraw (int seqnum) [private], [virtual]`

Parameters

<code>seqnum</code>	Provides the number of the sequence to draw.
---------------------	--

Implements `seq64::seqmenu`.

12.32.3.7 `void seq64::mainwnd::seq_set_and_edit (int seqnum) [private], [virtual]`

Used with the '=' key selection, by default.

Reimplemented from `seq64::seqmenu`.

12.32.3.8 `void seq64::mainwnd::seq_set_and_eventedit (int seqnum) [private], [virtual]`

Used with the '-' key selection, by default.

Reimplemented from `seq64::seqmenu`.

12.32.3.9 `void seq64::mainwnd::draw_marker_on_sequence (int seqnum, int tick) [private]`

If the sequence has no events, this function doesn't bother even drawing a position marker.

Note that, when `Sequencer64` first comes up, and `perform::is_dirty_main()` is called, no sequences exist yet. Also, currently the `redraw()` is hit when `seq_edit()` is called, but not when `seq_event_edit()` is called, which makes the latter not paint the in-edit highlight colors (if enabled). Why?

Parameters

<i>seqnum</i>	Provides the number of the sequence to draw.
<i>tick</i>	Provides the location to draw the marker. If pause support is compiled in (i.e. no <code>--disable-pause</code> in the configuration), then this parameter is ignored, and is replaced by the sequences' <code>get_lask_tick()</code> value. This causes correct stop/pause/play progress-bar behavior in each pattern slot.

12.32.3.10 `void seq64::mainwid::update_markers (int tick) [private]`

Parameters

<i>tick</i>	Starting point for drawing the markers.
-------------	---

12.32.3.11 `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.

12.32.3.12 `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.
---------------	---

12.32.3.13 `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.

12.32.3.14 `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.
---------------	--

12.32.3.15 `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.

12.32.3.16 `int seq64::mainwid::timeout () [private]`

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

Returns

Always returns true.

12.32.3.17 `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.

12.32.3.18 `void seq64::mainwid::select_fg_bg_colors (int seqnum) [private]`

12.32.3.19 `void seq64::mainwid::on_realize () [private]`

It allocates any additional resources that weren't initialized in the constructor.

This function used to call `font::init()`, and was the only place where the `font::init()` function was called. The `init()` function gets a color-map from the window. We need a more fool-proof way to do this!

12.32.3.20 `bool seq64::mainwid::on_expose_event (GdkEventExpose * ev)` `[private]`

Parameters

<code>ev</code>	The expose event.
-----------------	-------------------

Returns

Always returns true.

12.32.3.21 `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

<code>ev</code>	Provides the parameters of the button event.
-----------------	--

Returns

Always returns true.

12.32.3.22 `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).

12.32.3.23 `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.

12.32.3.24 `bool seq64::mainwid::on_focus_in_event (GdkEventFocus *)` `[private]`

Just sets the `Gtk::HAS_FOCUS` flag.

Returns

Always returns false.

12.32.3.25 `bool seq64::mainwid::on_focus_out_event (GdkEventFocus *)` `[private]`

Just unsets the `Gtk::HAS_FOCUS` flag.

Returns

Always returns false.

12.32.4 Friends And Related Function Documentation

12.32.4.1 `friend class mainwnd` `[friend]`

12.32.4.2 `void update_mainwid_sequences ()` `[friend]`

It is used by other objects that can modify the currently-edited sequence shown in the mainwid (main window).

12.32.5 Field Documentation

12.32.5.1 **sequence** seq64::mainwid::m_moving_seq [private]

The assignment is made by `sequence::partial_copy()`, which behaves like the legacy `seq24` code.

12.32.5.2 **bool** seq64::mainwid::m_button_down [private]

Used in the drag-and-drop functionality.

12.32.5.3 **bool** seq64::mainwid::m_moving [private]

12.32.5.4 **int** seq64::mainwid::m_old_seq [private]

12.32.5.5 **int** seq64::mainwid::m_screenset [private]

12.32.5.6 **long** seq64::mainwid::m_last_tick_x[c_max_sequence] [private]

12.32.5.7 **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.

12.32.5.8 **int** seq64::mainwid::m_mainwnd_cols [private]

12.32.5.9 **int** seq64::mainwid::m_seqarea_x [private]

12.32.5.10 **int** seq64::mainwid::m_seqarea_y [private]

12.32.5.11 **int** seq64::mainwid::m_seqarea_seq_x [private]

12.32.5.12 **int** seq64::mainwid::m_seqarea_seq_y [private]

12.32.5.13 **int** seq64::mainwid::m_mainwid_x [private]

12.32.5.14 **int** seq64::mainwid::m_mainwid_y [private]

12.32.5.15 **int** seq64::mainwid::m_mainwid_border [private]

12.32.5.16 **int** seq64::mainwid::m_mainwid_spacing [private]

12.32.5.17 **int** seq64::mainwid::m_text_size_x [private]

12.32.5.18 **int** seq64::mainwid::m_text_size_y [private]

12.32.5.19 **int** seq64::mainwid::m_max_sets [private]

12.32.5.20 **int** seq64::mainwid::m_screenset_slots [private]

It is equal to `m_mainwnd_rows * m_mainwnd_cols`.

12.32.5.21 `int seq64::mainwid::m_screenset_offset` `[private]`

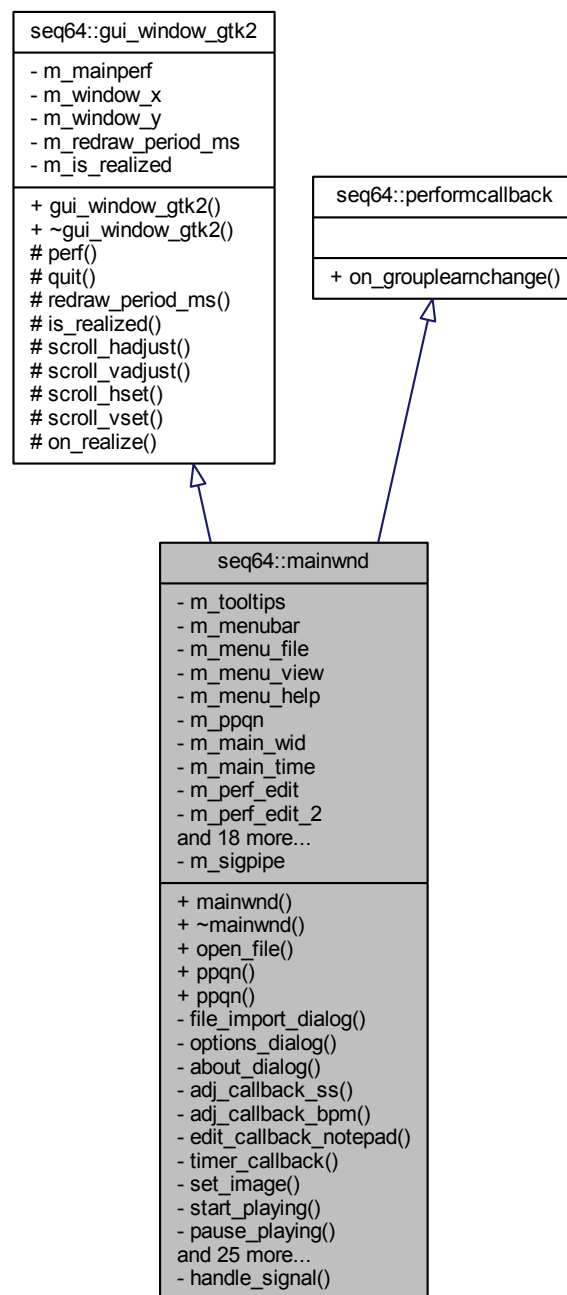
It is equally to `m_screenset_slots * m_screenset`.

12.32.5.22 `int seq64::mainwid::m_progress_height` `[private]`

12.33 `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` &a_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.*
- `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 file_import_dialog ()`
Presents a file dialog to import a MIDI file.
- `void options_dialog ()`
Opens the File / Options dialog.
- `void about_dialog ()`
Presents a Help / About dialog.
- `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_image (bool isrunning)`
Changes the image used for the pause/play 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 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 [file_save_as](#) ()
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.
- int [query_save_changes](#) ()
Queries the user to save the changes made while the application was running.
- 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 [on_delete_event](#) (GdkEventAny *a_e)
This callback function handles a delete event from ...?
- bool [on_key_press_event](#) (GdkEventKey *a_ev)
Handles a key press event.
- bool [on_key_release_event](#) (GdkEventKey *a_ev)
Handles a key release event.
- 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_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.
- `perfeddit * m_perf_edit`
A pointer to the first song/performance editor.
- `perfeddit * m_perf_edit_2`
A pointer to an optional second song/performance editor.
- `options * m_options`
A pointer to the program options.
- `Gdk::Cursor m_main_cursor`
Mouse cursor?
- `Gtk::Image * m_image_play`
Provides a pointer to hold the images for the pause/play button.
- `Gtk::Button * m_button_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_perfeddit`
The button for bringing up the Song Editor (Performance Editor).
- `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_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

12.33.1 Constructor & Destructor Documentation

12.33.1.1 `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.

View menu items and their hot keys.

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.

12.33.1.2 `seq64::mainwnd::~~mainwnd ()` [virtual]

12.33.2 Member Function Documentation

12.33.2.1 `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.
-----------	--

12.33.2.2 `int seq64::mainwnd::ppqn () const [inline]`

12.33.2.3 `void seq64::mainwnd::ppqn (int ppqn) [inline]`

`m_ppqn = choose_ppqn(ppqn);`

12.33.2.4 `void seq64::mainwnd::handle_signal (int sig) [static],[private]`

12.33.2.5 `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.

12.33.2.6 `void seq64::mainwnd::options_dialog () [private]`

12.33.2.7 `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.

12.33.2.8 `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.

12.33.2.9 `void seq64::mainwnd::adj_callback_bpm () [private]`

Let the perform object keep track of modifications.

12.33.2.10 void seq64::mainwnd::edit_callback_notepad () [private]

Let the perform object keep track of modifications.

12.33.2.11 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.

12.33.2.12 void seq64::mainwnd::set_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.
------------------	---

12.33.2.13 void seq64::mainwnd::start_playing () [private]

The [rc_settings::jack_start_mode\(\)](#) function is used (if jack is running) to determine if the playback mode is "live" (false) or "song" (true). An accessor to [perform::start_playing\(\)](#). This function is actually a callback for the pause/play button.

Note

This overrides the old behavior of playing live mode if the song is started from the main window. So let's go back to the way seq24 handles it. We could also make it dependent on the `-legacy` option, but that's too much trouble for now.

12.33.2.14 void seq64::mainwnd::pause_playing () [private]

Currently, it is just the same as [stop_playing\(\)](#), but we will get it to work.

12.33.2.15 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.

12.33.2.16 `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.

12.33.2.17 `void seq64::mainwnd::learn_toggle () [inline],[private]`

Simply forwards the call to [perform::learn_toggle\(\)](#).

12.33.2.18 `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.

12.33.2.19 `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.

12.33.2.20 `void seq64::mainwnd::enregister_perfedits () [private]`

12.33.2.21 `void seq64::mainwnd::sequence_key (int seq) [inline],[private]`

12.33.2.22 `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.

12.33.2.23 `void seq64::mainwnd::toLower (std::string & s) [private]`

12.33.2.24 `void seq64::mainwnd::file_new () [inline],[private]`

12.33.2.25 `void seq64::mainwnd::file_open () [inline],[private]`

12.33.2.26 `void seq64::mainwnd::file_save () [inline],[private]`

12.33.2.27 `void seq64::mainwnd::file_save_as () [private]`

12.33.2.28 `void seq64::mainwnd::file_exit () [private]`

12.33.2.29 `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.

12.33.2.30 `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`.

12.33.2.31 `void seq64::mainwnd::choose_file () [private]`

12.33.2.32 `int seq64::mainwnd::query_save_changes () [private]`

12.33.2.33 `bool seq64::mainwnd::is_save () [private]`

12.33.2.34 `bool seq64::mainwnd::install_signal_handlers () [private]`

12.33.2.35 `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.

12.33.2.36 `bool seq64::mainwnd::on_delete_event (GdkEventAny * a_e) [private]`

Any changed data is saved. If the pattern is playing, then it is stopped. We now use `is_running()`, instead of the global `rc().is_pattern_playing()` function.

12.33.2.37 `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.

12.33.2.38 `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.

Todo Test this functionality in old and new application.

Returns

Always returns false. This matches `seq24` behavior.

12.33.2.39 `void seq64::mainwnd::on_grouplearnchange (bool state)` `[private], [virtual]`

This handler responds to a learn-mode change from [perf\(\)](#).

Reimplemented from [seq64::performcallback](#).

12.33.3 Field Documentation

12.33.3.1 `int seq64::mainwnd::m_sigpipe` `[static], [private]`

This static member provides a couple of pipes for signalling/messaging.

12.33.3.2 `Gtk::Tooltips* seq64::mainwnd::m_tooltips` `[private]`

12.33.3.3 `Gtk::MenuBar* seq64::mainwnd::m_menubar` `[private]`

The whole menu bar.

12.33.3.4 `Gtk::Menu* seq64::mainwnd::m_menu_file` `[private]`

12.33.3.5 `Gtk::Menu* seq64::mainwnd::m_menu_view` `[private]`

12.33.3.6 `Gtk::Menu* seq64::mainwnd::m_menu_help` `[private]`

12.33.3.7 `int seq64::mainwnd::m_ppqn` `[private]`

We need it early here to be able to pass it along to child objects.

12.33.3.8 `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`, `perfname`s, and `seqedit` in order to allow the `seqedit` object to notify the `mainwid` (indirectly) of the currently-edited sequence.

12.33.3.9 `maintime* seq64::mainwnd::m_main_time` `[private]`

12.33.3.10 `perfedit* seq64::mainwnd::m_perf_edit` `[private]`

12.33.3.11 `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.

12.33.3.12 **options*** seq64::mainwnd::m_options [private]

12.33.3.13 **Gdk::Cursor** seq64::mainwnd::m_main_cursor [private]

12.33.3.14 **Gtk::Image*** seq64::mainwnd::m_image_play [private]

12.33.3.15 **Gtk::Button*** seq64::mainwnd::m_button_learn [private]

12.33.3.16 **Gtk::Button*** seq64::mainwnd::m_button_stop [private]

12.33.3.17 **Gtk::Button*** seq64::mainwnd::m_button_play [private]

If configured to support pause, it also supports the pause pixmap and functionality.

12.33.3.18 **Gtk::Button*** seq64::mainwnd::m_button_perfedit [private]

12.33.3.19 **Gtk::Adjustment*** seq64::mainwnd::m_adjust_bpm [private]

BPM adjustment object.

12.33.3.20 **Gtk::SpinButton*** seq64::mainwnd::m_spinbutton_bpm [private]

12.33.3.21 **Gtk::Adjustment*** seq64::mainwnd::m_adjust_ss [private]

Screenset adjustment.

12.33.3.22 **Gtk::SpinButton*** seq64::mainwnd::m_spinbutton_ss [private]

12.33.3.23 **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.

12.33.3.24 **Gtk::SpinButton*** seq64::mainwnd::m_spinbutton_load_offset [private]

12.33.3.25 **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.

12.33.3.26 `bool seq64::mainwnd::m_is_running` [private]

12.33.3.27 `sigc::connection seq64::mainwnd::m_timeout_connect` [private]

12.33.3.28 `bool seq64::mainwnd::m_call_seq_edit` [private]

Currently, the hard-wired key for this function is the equals key.

12.33.3.29 `bool seq64::mainwnd::m_call_seq_eventedit` [private]

Currently, the hard-wired key for this function is the minus key.

12.34 seq64::mastermidibus Class Reference

The class that "supervises" all of the midibus objects?

Public Member Functions

- [mastermidibus](#) (int ppqn=SEQ64_USE_DEFAULT_PPQN, int bpm=c_beats_per_minute)
The mastermidibus default constructor fills the array with our busses.
- [~mastermidibus](#) ()
The destructor deletes all of the output busses, clears out the ALSA events, stops and frees the queue, and closes ALSA for this application.
- void [init](#) (int ppqn)
Initialize the mastermidibus.
- `snd_seq_t *` [get_alsa_seq](#) () const
'Getter' function for member m_alsa_seq
- 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
- void [set_beats_per_minute](#) (int bpm)
Set the BPM value (beats per minute).
- void [set_ppqn](#) (int ppqn)
Set the PPQN value (parts per quarter note).
- int [get_beats_per_minute](#) () const
'Getter' function for member m_beats_per_minute
- int [get_ppqn](#) () const
'Getter' function for member m_ppqn
- std::string [get_midi_out_bus_name](#) (int bus)
Get the MIDI output buss name for the given (legal) buss number.
- std::string [get_midi_in_bus_name](#) (int bus)
Get the MIDI input buss name for the given (legal) buss number.
- void [print](#) ()
Print some information about the available MIDI output busses.
- void [flush](#) ()
Flushes our local queue events out into ALSA.

- void `start` ()
Starts all of the configured output busses up to `m_num_out_buses`.
- void `stop` ()
Stops each of the output busses.
- void `clock` (midipulse tick)
Generates the MIDI clock for each of the output busses.
- void `continue_from` (midipulse tick)
Gets the output busses running again, if ALSA support is enabled.
- void `init_clock` (midipulse tick)
Initializes the clock of each of the output busses.
- int `poll_for_midi` ()
Initiate a poll() on the existing poll descriptors.
- bool `is_more_input` ()
Test the ALSA sequencer to see if any more input is pending.
- bool `get_midi_event` (event *in)
Grab a MIDI event.
- void `set_sequence_input` (bool state, sequence *seq)
Set the input sequence object, and set the `m_dumping_input` value to the given state.
- bool `is_dumping` () const
'Getter' function for member `m_dumping_input`
- sequence * `get_sequence` () const
'Getter' function for member `m_seq`
- void `sysex` (event *event)
Handle the sending of SYSEX events.
- void `port_start` (int client, int port)
Start the given ALSA 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 `set_clock` (bussbyte bus, clock_e clock_type)
Set the clock for the given (legal) buss number.
- clock_e `get_clock` (bussbyte bus)
Gets the clock setting for the given (legal) buss number.
- void `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.

Private Attributes

- snd_seq_t * `m_alsa_seq`
The ALSA sequencer client handle.
- int `m_num_out_buses`
The number of output busses.
- int `m_num_in_buses`
The number of input busses.
- midibus * `m_buses_out` [`c_max_busses`]
Output MIDI busses.
- midibus * `m_buses_in` [`c_max_busses`]

- Input MIDI busses.*
- `midibus * m_bus_announce`
- MIDI buss announcer?*
- `bool m_buses_out_active [c_max_busses]`
- Active output MIDI busses.*
- `bool m_buses_in_active [c_max_busses]`
- Active input MIDI busses.*
- `bool m_buses_out_init [c_max_busses]`
- Output MIDI buss initialization.*
- `bool m_buses_in_init [c_max_busses]`
- Input MIDI buss initialization.*
- `clock_e m_init_clock [c_max_busses]`
- Clock initialization.*
- `bool m_init_input [c_max_busses]`
- Input initialization?*
- `int m_queue`
- The ID of the MIDI queue.*
- `int m_ppqn`
- Resolution in parts per quarter note.*
- `int m_beats_per_minute`
- BPM (beats per minute).*
- `int m_num_poll_descriptors`
- The number of descriptors for polling.*
- `struct pollfd * m_poll_descriptors`
- Points to the list of descriptors for polling.*
- `bool m_dumping_input`
- For dumping MIDI input to a sequence for recording.*
- `sequence * m_seq`
- Points to the sequence object.*
- `mutex m_mutex`
- The locking mutex.*

12.34.1 Constructor & Destructor Documentation

12.34.1.1 `seq64::mastermidibus::mastermidibus (int ppqn = SEQ64_USE_DEFAULT_PPQN, int bpm = c_beats_per_minute)`

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 <code>mastermidibus::set_ppqn()</code> to set up the proper PPQN value.
<i>bpm</i>	Provides the beats per minute value, which defaults to <code>c_beats_per_minute</code> .

12.34.1.2 `seq64::mastermidibus::~~mastermidibus ()`

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]

12.34.2 Member Function Documentation

12.34.2.1 `void seq64::mastermidibus::init (int ppqn)`

It initializes 16 MIDI output busses, a hardwired constant, `SEQ64_ALSA_OUTPUT_BUSS_MAX == 16`. Only one MIDI input buss is initialized.

Parameters

<i>ppqn</i>	The PPQN value to which to initialize the master MIDI buss.
-------------	---

12.34.2.2 `snd_seq_t* seq64::mastermidibus::get_alsa_seq () const` `[inline]`

12.34.2.3 `int seq64::mastermidibus::get_num_out_buses () const` `[inline]`

12.34.2.4 `int seq64::mastermidibus::get_num_in_buses () const` `[inline]`

12.34.2.5 `void seq64::mastermidibus::set_beats_per_minute (int bpm)`

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.

Threadsafe

Parameters

<i>bpm</i>	Provides the beats-per-minute value to set.
------------	---

12.34.2.6 `void seq64::mastermidibus::set_ppqn (int ppqn)`

This is done by creating an ALSA tempo structure, adding tempo information to it, and then setting the ALSA sequencer object with this information. Fills the tempo structure with the current tempo information. Then sets the `ppqn` value. Finally, gives the tempo structure to the ALSA queue.

Threadsafe

Parameters

<i>ppqn</i>	The PPQN value to be set.
-------------	---------------------------

12.34.2.7 `int seq64::mastermidibus::get_beats_per_minute () const [inline]`

12.34.2.8 `int seq64::mastermidibus::get_ppqn () const [inline]`

12.34.2.9 `std::string seq64::mastermidibus::get_midi_out_bus_name (int bus)`

Parameters

<i>bus</i>	Provides the output buss 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.

12.34.2.10 `std::string seq64::mastermidibus::get_midi_in_bus_name (int bus)`

Parameters

<i>bus</i>	Provides the input buss 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.

12.34.2.11 `void seq64::mastermidibus::print ()`

12.34.2.12 `void seq64::mastermidibus::flush ()`

Threadsafe

12.34.2.13 `void seq64::mastermidibus::start ()`

Threadsafe

12.34.2.14 `void seq64::mastermidibus::stop ()`

If ALSA support is enable, also drains the output, synchronizes the output queue, and then stop the queue.

Threadsafe

12.34.2.15 `void seq64::mastermidibus::clock (midipulse tick)`

Threadsafe

Parameters

<i>tick</i>	Provides the tick value with which to set the buss clock.
-------------	---

12.34.2.16 void seq64::mastermidibus::continue_from (midipulse *tick*)

Threadsafe

Parameters

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

12.34.2.17 void seq64::mastermidibus::init_clock (midipulse *tick*)

Threadsafe

Parameters

<i>tick</i>	Provides the tick value with which to initialize the buss clock.
-------------	--

12.34.2.18 int seq64::mastermidibus::poll_for_midi ()

Returns

Returns the result of the poll, or 0 if ALSA is not supported.

12.34.2.19 bool seq64::mastermidibus::is_more_input ()

Threadsafe

Returns

Returns true if ALSA is supported, and the returned size is greater than 0, or false otherwise.

12.34.2.20 bool seq64::mastermidibus::get_midi_event (event * *inev*)

Threadsafe

Parameters

<i>inev</i>	The event to be set based on the found input event.
-------------	---

12.34.2.21 `void seq64::mastermidibus::set_sequence_input (bool state, sequence * seq)`

Threadsafe

Parameters

<i>state</i>	Provides the dumping-input state to be set.
<i>seq</i>	Provides the sequence object to be logged as the mastermidibus's sequence. Can also be used to set a null pointer, to disable the sequence setting.

12.34.2.22 `bool seq64::mastermidibus::is_dumping () const [inline]`

12.34.2.23 `sequence* seq64::mastermidibus::get_sequence () const [inline]`

12.34.2.24 `void seq64::mastermidibus::sysex (event * ev)`

Threadsafe

Parameters

<i>ev</i>	Provides the event pointer to be set.
-----------	---------------------------------------

12.34.2.25 `void seq64::mastermidibus::port_start (int client, int port)`

Threadsafe Quite a lot is done during the lock!

Parameters

<i>client</i>	Provides the ALSA client number.
<i>port</i>	Provides the ALSA client port.

12.34.2.26 `void seq64::mastermidibus::port_exit (int client, int port)`

Both the input and output busses for the given client are stopped, and set to inactive.

Threadsafe

Parameters

<i>client</i>	The client to be matched and acted on.
<i>port</i>	The port to be acted on. Both parameter must be match before the buss is made inactive.

12.34.2.27 void seq64::mastermidibus::play (bussbyte bus, event * e24, midibyte channel)

Threadsafe

Parameters

<i>bus</i>	The buss to start play on.
<i>e24</i>	The seq24 event to play on the buss.
<i>channel</i>	The channel on which to play the event.

12.34.2.28 void seq64::mastermidibus::set_clock (bussbyte bus, clock_e clocktype)

The legality checks are a little loose, however.

Threadsafe

Parameters

<i>bus</i>	The buss to start play on.
<i>clocktype</i>	The type of clock to be set, either "off", "pos", or "mod", as noted in the midibus_common module.

12.34.2.29 clock_e seq64::mastermidibus::get_clock (bussbyte bus)

Parameters

<i>bus</i>	Provides the buss number to read.
------------	-----------------------------------

Returns

If the buss number is legal, and the buss is active, then its clock setting is returned. Otherwise, e_clock_off is returned.

12.34.2.30 void seq64::mastermidibus::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.

Threadsafe

Parameters

<i>bus</i>	Provides the buss number.
<i>inputing</i>	True if the input bus will be inputting MIDI data.

12.34.2.31 bool seq64::mastermidibus::get_input (bussbyte bus)

Parameters

<i>bus</i>	Provides the buss number.
------------	---------------------------

Returns

Always returns false.

12.34.3 Field Documentation

12.34.3.1 `snd_seq_t* seq64::mastermidibus::m_alsa_seq` [private]

12.34.3.2 `int seq64::mastermidibus::m_num_out_buses` [private]

12.34.3.3 `int seq64::mastermidibus::m_num_in_buses` [private]

12.34.3.4 `midibus* seq64::mastermidibus::m_buses_out[c_max_busses]` [private]

12.34.3.5 `midibus* seq64::mastermidibus::m_buses_in[c_max_busses]` [private]

12.34.3.6 `midibus* seq64::mastermidibus::m_bus_announce` [private]

12.34.3.7 `bool seq64::mastermidibus::m_buses_out_active[c_max_busses]` [private]

12.34.3.8 `bool seq64::mastermidibus::m_buses_in_active[c_max_busses]` [private]

12.34.3.9 `bool seq64::mastermidibus::m_buses_out_init[c_max_busses]` [private]

12.34.3.10 `bool seq64::mastermidibus::m_buses_in_init[c_max_busses]` [private]

12.34.3.11 `clock_e seq64::mastermidibus::m_init_clock[c_max_busses]` [private]

12.34.3.12 `bool seq64::mastermidibus::m_init_input[c_max_busses]` [private]

12.34.3.13 `int seq64::mastermidibus::m_queue` [private]

12.34.3.14 `int seq64::mastermidibus::m_ppqn` [private]

12.34.3.15 `int seq64::mastermidibus::m_beats_per_minute` [private]

We had to lengthen this name; way too easy to confuse it with "bpm" for "beats per measure".

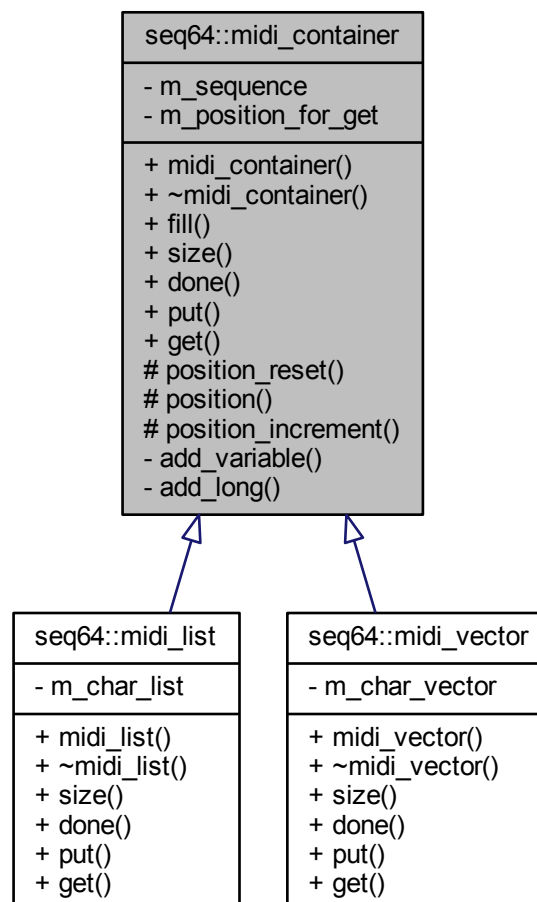
- 12.34.3.16 `int seq64::mastermidibus::m_num_poll_descriptors` [private]
- 12.34.3.17 `struct pollfd* seq64::mastermidibus::m_poll_descriptors` [private]
- 12.34.3.18 `bool seq64::mastermidibus::m_dumping_input` [private]
- 12.34.3.19 `sequence* seq64::mastermidibus::m_seq` [private]
- 12.34.3.20 `mutex seq64::mastermidibus::m_mutex` [private]

This object is passed to an automutex object that lends exception-safety to the mutex locking.

12.35 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)`
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 ()=0`
Provide a way to get the next byte from 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.

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.

12.35.1 Detailed Description

It is the base class for `midi_list` and `midi_vector`.

12.35.2 Constructor & Destructor Documentation

12.35.2.1 `seq64::midi_container::midi_container (sequence & seq)`

Parameters

<code>seq</code>	Provides a reference to the sequence/track for which this container holds MIDI data.
------------------	--

12.35.2.2 `virtual seq64::midi_container::~~midi_container () [inline],[virtual]`

12.35.3 Member Function Documentation

12.35.3.1 `void seq64::midi_container::fill (int tracknumber)`

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_container\(\)](#).

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 assume to increment. This application doesn't use with that shortcut.

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

<code>tracknumber</code>	Provides the track number. This number is masked into the track information.
--------------------------	--

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.

12.35.3.2 `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](#).

12.35.3.3 `virtual bool seq64::midi_container::done () const [inline],[virtual]`

Reimplemented in [seq64::midi_vector](#), and [seq64::midi_list](#).

12.35.3.4 `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](#).

12.35.3.5 `virtual midibyte seq64::midi_container::get () [pure virtual]`

It also increments `m_position_for_get`.

Implemented in [seq64::midi_vector](#), and [seq64::midi_list](#).

12.35.3.6 `unsigned int seq64::midi_container::position_reset () const [inline],[protected]`

12.35.3.7 `unsigned int seq64::midi_container::position () const [inline],[protected]`

12.35.3.8 `void seq64::midi_container::position_increment () const [inline],[protected]`

12.35.3.9 `void seq64::midi_container::add_variable (midipulse v) [private]`

This function "replaces" `sequence::add_list_var()`.

Parameters

<code>v</code>	The data value to be added to the current event in the MIDI container.
----------------	--

12.35.3.10 `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

<code>x</code>	Provides the timestamp (pulse value) to be added to the container.
----------------	--

12.35.4 Field Documentation

12.35.4.1 `sequence& seq64::midi_container::m_sequence [private]`

12.35.4.2 `unsigned int seq64::midi_container::m_position_for_get [mutable],[private]`

12.36 seq64::midi_control Class Reference

This class (formerly a struct) contains the control information for sequences that make up a live set.

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.

12.36.1 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. 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.

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*.

12.36.2 Constructor & Destructor Documentation

12.36.2.1 `seq64::midi_control::midi_control () [inline]`

Every member is either false or zero.

12.36.3 Member Function Documentation

12.36.3.1 `bool seq64::midi_control::active () const [inline]`

12.36.3.2 `bool seq64::midi_control::inverse_active () const [inline]`

12.36.3.3 `int seq64::midi_control::status () const [inline]`

12.36.3.4 `int seq64::midi_control::data () const [inline]`

12.36.3.5 `int seq64::midi_control::min_value () const [inline]`

12.36.3.6 `int seq64::midi_control::max_value () const [inline]`

12.36.3.7 `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: <i>m_active</i> , <i>m_inverse_active</i> , <i>m_status</i> , <i>m_data</i> , <i>m_min_value</i> , and <i>m_max_value</i> .
---------------	--

12.36.3.8 `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.
---------------	--

12.36.3.9 `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.

12.36.3.10 `bool seq64::midi_control::in_range (midibyte data) const` `[inline]`

12.36.4 Field Documentation

12.36.4.1 `bool seq64::midi_control::m_active` `[private]`

12.36.4.2 `bool seq64::midi_control::m_inverse_active` `[private]`

12.36.4.3 `int seq64::midi_control::m_status` `[private]`

12.36.4.4 `int seq64::midi_control::m_data` `[private]`

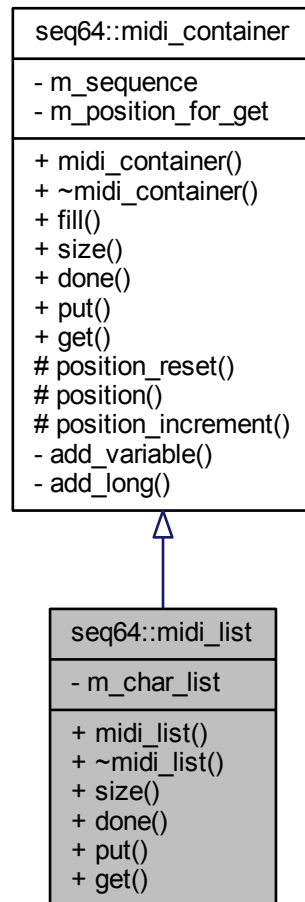
12.36.4.5 `int seq64::midi_control::m_min_value` `[private]`

12.36.4.6 `int seq64::midi_control::m_max_value` `[private]`

12.37 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 ()`

Provide a way to get the next byte from 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

12.37.1 Member Typedef Documentation

12.37.1.1 `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.

12.37.2 Constructor & Destructor Documentation

12.37.2.1 `seq64::midi_list::midi_list (sequence & seq)`

Parameters

<code>seq</code>	The sequence/track object that is using this container.
------------------	---

12.37.2.2 `virtual seq64::midi_list::~~midi_list ()` [inline],[virtual]

12.37.3 Member Function Documentation

12.37.3.1 `virtual std::size_t seq64::midi_list::size () const` [inline],[virtual]

Reimplemented from [seq64::midi_container](#).

12.37.3.2 `virtual bool seq64::midi_list::done () const` [inline],[virtual]

Reimplemented from [seq64::midi_container](#).

12.37.3.3 `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](#).

12.37.3.4 `virtual midibyte seq64::midi_list::get () [inline],[virtual]`

In this implementation, `m_position_for_get` is not used. The elements of the container are popped off backward!

Implements [seq64::midi_container](#).

12.37.4 Field Documentation

12.37.4.1 `CharList seq64::midi_list::m_char_list [private]`

12.38 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.

12.38.1 Detailed Description

More commonly known as "bars:beats:ticks", or "BBT".

12.38.2 Constructor & Destructor Documentation

12.38.2.1 `seq64::midi_measures::midi_measures ()`

12.38.2.2 `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.

12.38.3 Member Function Documentation

12.38.3.1 `int seq64::midi_measures::measures () const` `[inline]`

12.38.3.2 `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.
----------	--

12.38.3.3 `int seq64::midi_measures::beats () const` `[inline]`

12.38.3.4 `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.
----------	---

12.38.3.5 `int seq64::midi_measures::divisions () const` `[inline]`

12.38.3.6 `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.
----------	---

12.38.4 Field Documentation

12.38.4.1 `int seq64::midi_measures::m_measures` `[private]`

12.38.4.2 `int seq64::midi_measures::m_beats` `[private]`

12.38.4.3 `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.

12.39 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, 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.

12.39.1 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.

12.39.2 Constructor & Destructor Documentation

12.39.2.1 seq64::midi_splitter::midi_splitter (int *ppqn* = SEQ64_USE_DEFAULT_PPQN)

Parameters

<i>ppqn</i>	<p>Provides the initial value of the PPQN setting. It is handled differently for parsing (reading) versus writing the MIDI file.</p> <ul style="list-style-type: none"> • Reading. <ul style="list-style-type: none"> – If set to 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>. – Otherwise, <code>m_ppqn</code> is set to the value read from the MIDI file. No scaling is done. Since the value gets written, specify <code>ppqn</code> as 0, an obviously bogus value, to get this behavior. • Writing. This value is written to the MIDI file in the header chunk of the song. Note that the caller must query for the PPQN set during parsing, and pass it to the constructor when preparing to write the file. See how it is done in the <code>mainwnd</code> class.
-------------	---

12.39.2.2 seq64::midi_splitter::~~midi_splitter ()

12.39.3 Member Function Documentation

12.39.3.1 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.

12.39.3.2 void seq64::midi_splitter::initialize ()

12.39.3.3 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.
----------------	---

12.39.3.4 `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 patter 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.

12.39.3.5 `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.

12.39.3.6 `int seq64::midi_splitter::count () const [inline]`

12.39.3.7 `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*.

12.39.4 Field Documentation

12.39.4.1 `int seq64::midi_splitter::m_ppqn` [private]

12.39.4.2 `bool seq64::midi_splitter::m_use_default_ppqn` [private]

12.39.4.3 `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.

12.39.4.4 `bool seq64::midi_splitter::m_smf0_channels[16]` [private]

Obviously, we don't have to worry about multiple MIDI busses.

12.39.4.5 `sequence* seq64::midi_splitter::m_smf0_main_sequence` [private]

12.39.4.6 `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.

12.40 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 (int bpmminute, int bpmmeasure, int beatwidth, int ppqn)`
Principal constructor for [midi_timing](#).
- `int beats_per_minute () const`
'Getter' function for member `m_beats_per_minute`
- `void beats_per_minute (int 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

- int `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.

12.40.1 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.

12.40.2 Constructor & Destructor Documentation

12.40.2.1 `seq64::midi_timing::midi_timing ()`

12.40.2.2 `seq64::midi_timing::midi_timing (int bpmminute, int bpmeasure, int beatwidth, int ppqn)`

Parameters

<i>bpmminute</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.

12.40.3 Member Function Documentation

12.40.3.1 `int seq64::midi_timing::beats_per_minute () const` `[inline]`

12.40.3.2 `void seq64::midi_timing::beats_per_minute (int b)` `[inline]`

Parameters

<i>b</i>	The value to which to set the number of beats/minute. We can add validation later.
----------	--

12.40.3.3 `int seq64::midi_timing::beats_per_measure () const` `[inline]`

12.40.3.4 `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.
----------	---

12.40.3.5 `int seq64::midi_timing::beat_width () const` `[inline]`

12.40.3.6 `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.
-----------	--

12.40.3.7 `int seq64::midi_timing::ppqn () const` `[inline]`

12.40.3.8 `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.
----------	---

12.40.4 Field Documentation

12.40.4.1 `int 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.

12.40.4.2 `int seq64::midi_timing::m_beats_per_measure` `[private]`

This value is most commonly set to 4.

12.40.4.3 `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.

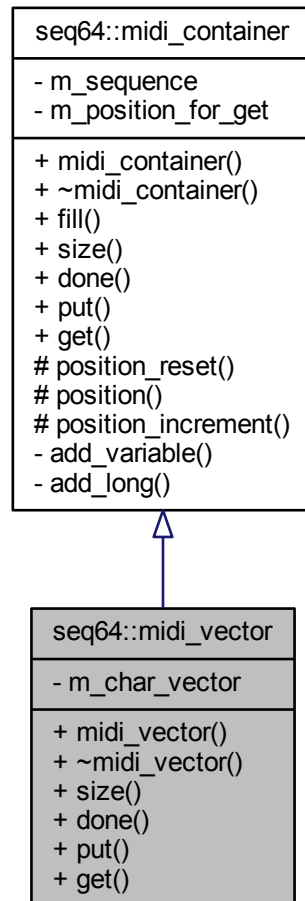
12.40.4.4 `int seq64::midi_timing::m_ppqn` `[private]`

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.

12.41 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](#) ()
Provide a way to get the next byte from 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

12.41.1 Member Typedef Documentation

12.41.1.1 typedef std::vector<midibyte> seq64::midi_vector::CharVector [private]

12.41.2 Constructor & Destructor Documentation

12.41.2.1 seq64::midi_vector::midi_vector ([sequence](#) & [seq](#))

Parameters

seq	Provides a reference to the sequence/track for which this container holds MIDI data.
---------------------	--

12.41.2.2 virtual seq64::midi_vector::~~midi_vector () [inline],[virtual]

12.41.3 Member Function Documentation

12.41.3.1 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](#).

12.41.3.2 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](#).

12.41.3.3 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](#).

12.41.3.4 **virtual midibyte seq64::midi_vector::get ()** [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](#).

12.41.4 Field Documentation

12.41.4.1 **CharVector seq64::midi_vector::m_char_vector** [private]

12.42 seq64::midibus Class Reference

Provides a class for handling the MIDI buss on Linux.

Public Member Functions

- [midibus](#) (int localclient, int destclient, int destport, snd_seq_t *seq, const char *client_name, const char *port_name, int id, int queue, int ppqn=SEQ64_USE_DEFAULT_PPQN)
Provides a constructor with client number, port number, ALSA sequencer support, name of client, name of port.
- [midibus](#) (int localclient, snd_seq_t *seq, int id, int queue, int ppqn=SEQ64_USE_DEFAULT_PPQN)
Secondary constructor.
- [~midibus](#) ()
A rote empty destructor.
- 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 [print](#) ()
Prints m_name.
- const std::string & [get_name](#) () const

- *'Getter' function for member n_name*
- int `get_id` () const
- *'Getter' function for member m_id*
- void `play` (event *e24, midibyte channel)
- *This `play()` function takes a native event, encodes it to an ALSA event, 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 `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 `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_input` (bool inputing)
- *Set status to of "inputting" to the given value.*
- bool `get_input` () const
- *'Getter' function for member m_inputing*
- void `flush` ()
- *Flushes our local queue events out into ALSA.*
- int `get_client` () const
- *'Getter' function for member m_dest_addr_client The address of client.*
- int `get_port` () const
- *'Getter' function for member m_dest_addr_port*

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.*

Private Attributes

- int `m_id`
- *The ID of the midibus object.*
- `clock_e` `m_clock_type`
- *The type of clock to use.*
- bool `m_inputing`
- *TBD.*
- int `m_ppqn`
- *Provides the PPQN value in force, currently a constant.*

- `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.
- `int m_queue`
Another ID of the MIDI queue?
- `std::string m_name`
The name of the MIDI buss.
- `midipulse m_lasttick`
The last (most recent? final?) tick.
- `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.

12.42.1 Constructor & Destructor Documentation

- 12.42.1.1 `seq64::midibus::midibus (int localclient, int destclient, int destport, snd_seq_t * seq, const char * client_name, const char * port_name, int id, int queue, int ppqn = SEQ64_USE_DEFAULT_PPQN)`

Parameters

<i>localclient</i>	Provides the local-client number.
<i>destclient</i>	Provides the destination-client number.
<i>destport</i>	Provides the destination-client port.
<i>seq</i>	Provides the sequence that will work with this buss.
<i>client_name</i>	Provides the client name, but this parameter is unused.
<i>port_name</i>	Provides the port name.
<i>id</i>	Provides the ID code for this bus. It is an index into the midibus definitions array, and is also used in the constructed human-readable buss name.
<i>queue</i>	Provides the queue ID.
<i>ppqn</i>	Provides the PPQN value.

12.42.1.2 `seq64::midibus::midibus (int localclient, snd_seq_t * seq, int id, int queue, int ppqn = SEQ64_USE_DEFAULT_PPQN)`

Similar to the principal constructor, but labels the buss by number more than by name.

Parameters

<i>localclient</i>	Provides the local-client number.
<i>seq</i>	Provides the sequence that will work with this buss.
<i>id</i>	Provides the ID code for this bus. It is an index into the midibus definitions array, and is also used in the constructed human-readable buss name.
<i>queue</i>	Provides the queue ID.
<i>ppqn</i>	Provides the PPQN value.

12.42.1.3 `seq64::midibus::~~midibus ()`

12.42.2 Member Function Documentation

12.42.2.1 `bool seq64::midibus::init_out ()`

Returns

Returns true unless setting up ALSA MIDI failed in some way.

12.42.2.2 `bool seq64::midibus::init_in ()`

Returns

Returns true unless setting up ALSA MIDI failed in some way.

12.42.2.3 `bool seq64::midibus::deinit_in ()`

Returns

Returns true, unless an error occurs.

12.42.2.4 `bool seq64::midibus::init_out_sub ()`

Returns

Returns true unless setting up the ALSA port failed in some way.

12.42.2.5 `bool seq64::midibus::init_in_sub ()`

Returns

Returns true unless setting up the ALSA port failed in some way.

12.42.2.6 void seq64::midibus::print ()

12.42.2.7 const std::string& seq64::midibus::get_name () const [inline]

12.42.2.8 int seq64::midibus::get_id () const [inline]

12.42.2.9 void seq64::midibus::play (event * e24, midibyte channel)

Threadsafe

Parameters

<i>e24</i>	The event to be played on this bus.
<i>channel</i>	The channel of the playback.

12.42.2.10 void seq64::midibus::sysex (event * e24)

Parameters

<i>e24</i>	The event to be handled.
------------	--------------------------

12.42.2.11 void seq64::midibus::start ()

12.42.2.12 void seq64::midibus::stop ()

12.42.2.13 void seq64::midibus::clock (midipulse tick)

Parameters

<i>tick</i>	Provides the starting tick.
-------------	-----------------------------

12.42.2.14 void seq64::midibus::continue_from (midipulse tick)

Parameters

<i>tick</i>	The continuing tick.
-------------	----------------------

12.42.2.15 void seq64::midibus::init_clock (midipulse tick)

Parameters

<i>tick</i>	The starting tick.
-------------	--------------------

12.42.2.16 void seq64::midibus::set_clock (clock_e clocktype) [inline]

Parameters

<i>clocktype</i>	The value used to set the clock-type.
------------------	---------------------------------------

12.42.2.17 `clock_e seq64::midibus::get_clock () const` `[inline]`

12.42.2.18 `void seq64::midibus::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.
-----------------	----------------------------

12.42.2.19 `bool seq64::midibus::get_input () const` `[inline]`

12.42.2.20 `void seq64::midibus::flush ()`

12.42.2.21 `int seq64::midibus::get_client () const` `[inline]`

12.42.2.22 `int seq64::midibus::get_port () const` `[inline]`

12.42.2.23 `static void seq64::midibus::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> .
-----------------	---

12.42.2.24 `static int seq64::midibus::get_clock_mod ()` `[inline],[static]`

12.42.3 Friends And Related Function Documentation

12.42.3.1 `friend class mastermidibus` `[friend]`

12.42.4 Field Documentation

12.42.4.1 `int seq64::midibus::m_clock_mod` `[static],[private]`

Initialize this static member.

```

12.42.4.2  int seq64::midibus::m_id    [private]

12.42.4.3  clock_e seq64::midibus::m_clock_type [private]

12.42.4.4  bool seq64::midibus::m_inputting [private]

12.42.4.5  int seq64::midibus::m_ppqn   [private]

12.42.4.6  snd_seq_t* const seq64::midibus::m_seq [private]

12.42.4.7  const int seq64::midibus::m_dest_addr_client [private]

12.42.4.8  const int seq64::midibus::m_dest_addr_port [private]

12.42.4.9  const int seq64::midibus::m_local_addr_client [private]

12.42.4.10 int seq64::midibus::m_local_addr_port [private]

12.42.4.11 int seq64::midibus::m_queue [private]

12.42.4.12 std::string seq64::midibus::m_name [private]

12.42.4.13 midipulse seq64::midibus::m_lasttick [private]

12.42.4.14 mutex seq64::midibus::m_mutex [private]

```

12.43 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 globalbgs=true)
Principal constructor.
- [~midifile](#) ()
A rote destructor.
- bool [parse](#) ([perform](#) &a_perf, int a_screen_set=0)
This function opens a binary MIDI file and parses it into sequences and other application objects.
- bool [write](#) ([perform](#) &a_perf)
Write the whole MIDI data and Seq24 information out to the file.
- 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 seq25 [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.
- void [write_prop_header](#) ([midilong](#) tag, long len)
We want to write:
- bool [write_proprietary_track](#) ([perform](#) &a_perf)
Writes out the 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.
- 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

- 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.

12.43.1 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.

12.43.2 Constructor & Destructor Documentation

12.43.2.1 `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>	Provides the initial value of the PPQN setting. It is handled differently for parsing (reading) versus writing the MIDI file. <ul style="list-style-type: none"> • Reading. <ul style="list-style-type: none"> – If set to SEQ64_USE_DEFAULT_PPQN, the legacy application behavior is used. The m_ppqn member is set to the default PPQN, DEFAULT_PPQN. The value read from the MIDI file, ppqn, is then use to scale the running-time of the sequence relative to DEFAULT_PPQN. – Otherwise, m_ppqn is set to the value read from the MIDI file. No scaling is done. Since the value gets written, specify ppqn as 0, an obviously bogus value, to get this behavior. • Writing. This value is written to the MIDI file in the header chunk of the song. Note that the caller must query for the PPQN set during parsing, and pass it to the constructor when preparing to write the file. See how it is done in the mainwnd class.
<i>oldformat</i>	If true, write out the MIDI file using the old Seq24 format, instead of the new MIDI-compliant sequencer-specific format, for the seq24-specific SeqSpec tags defined in the globals module. This option is false by default. Note that this option is only used in writing; reading can handle either format transparently.
<i>globalbgs</i>	If true, write any non-default values of the key, scale, and background sequence to the global "proprietary" section of the MIDI file, instead of to each sequence. Note that this option is only used in writing; reading can handle either format transparently.

12.43.2.2 `seq64::midifile::~~midifile ()`

12.43.3 Member Function Documentation

12.43.3.1 `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.

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`.

12.43.3.2 bool seq64::midifile::write (perform & p)

Parameters

<i>p</i>	Provides the object that will contain and manage the entire performance.
----------	--

Returns

Returns true if the write operations succeeded.

Note

Seq24 reverses the order of some events, due to popping from its container. Not an issue here.

12.43.3.3 `const std::string& seq64::midifile::error_message () const [inline]`

12.43.3.4 `bool seq64::midifile::error_is_fatal () const [inline]`

12.43.3.5 `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.

12.43.3.6 `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.

12.43.3.7 `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.

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.

12.43.3.8 `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.

12.43.3.9 `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.

Parameters

<i>p</i>	The performance object that is being set via the incoming MIDI file.
<i>file_size</i>	The file size as determined in the parse() function.

There are also implicit parameters, with the `m_pos` and `m_new_format` member variables.

12.43.3.10 `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.

12.43.3.11 `bool seq64::midifile::checklen (midilong len, midibyte type) [private]`

Parameters

<i>len</i>	The length value to be checked, and it should be greater than 0.
<i>type</i>	The type of meta event. Used for displaying an error.

Returns

Returns true if the length parameter is valid.

12.43.3.12 `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 <code>m_use_default_ppqn</code> is true. If 0, the ppqn value is not used.

12.43.3.13 `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.

12.43.3.14 `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.

12.43.3.15 `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.

12.43.3.16 `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.

12.43.3.17 `void seq64::midifile::write_long (midilong x) [private]`

Warning

This code looks endian-dependent.

Parameters

<i>x</i>	The long value to be written to the MIDI file.
----------	--

12.43.3.18 `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.
----------	---

12.43.3.19 `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.

12.43.3.20 `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".
----------	--------------------------------

12.43.3.21 `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".
- 0xFFFFFFFF (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.
--------------	---

12.43.3.22 `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.
------------------	--

12.43.3.23 `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.

12.43.3.24 `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.
---------------	-------------------------------

12.43.3.25 `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.

12.43.3.26 `void seq64::midifile::write_track_end ()` [private]

12.43.3.27 `void seq64::midifile::write_prop_header (midilong control_tag, long data_length)` [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.
```

Writes a "proprietary" (SeqSpec) Seq24 footer header in either the new MIDI-compliant format, or the legacy Seq24 format. 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.

12.43.3.28 `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.

12.43.3.29 `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

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.

12.43.3.30 `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.

12.43.3.31 `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".

12.43.3.32 `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.

12.43.3.33 `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.

12.43.3.34 `long seq64::midifile::seq_number_size () const [inline],[private]`

12.43.3.35 `long seq64::midifile::track_end_size () const [inline],[private]`

12.43.3.36 `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.

12.43.4 Field Documentation

12.43.4.1 `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.

12.43.4.2 `std::string seq64::midifile::m_error_message [private]`

If empty, there's no pending error. Currently most useful in the [parse\(\)](#) function.

12.43.4.3 `bool seq64::midifile::m_error_is_fatal [private]`

The caller can query for this value after getting the return value from [parse\(\)](#).

12.43.4.4 `bool seq64::midifile::m_disable_reported [private]`

Once is enough.

12.43.4.5 `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.

12.43.4.6 `const std::string seq64::midifile::m_name` `[private]`

12.43.4.7 `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.

12.43.4.8 `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.

12.43.4.9 `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.

12.43.4.10 `bool seq64::midifile::m_global_bgsequence` `[private]`

12.43.4.11 `int seq64::midifile::m_ppqn` `[private]`

12.43.4.12 `bool seq64::midifile::m_use_default_ppqn` `[private]`

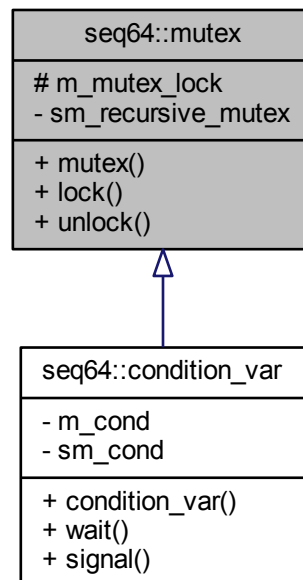
12.43.4.13 `midi_splitter seq64::midifile::m_smf0_splitter` `[private]`

This object holds all of the information needed to split a multi-channel sequence.

12.44 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.

12.44.1 Constructor & Destructor Documentation

12.44.1.1 seq64::mutex::mutex ()

12.44.2 Member Function Documentation

12.44.2.1 void seq64::mutex::lock () const

12.44.2.2 void seq64::mutex::unlock () const

12.44.3 Field Documentation

12.44.3.1 const pthread_mutex_t seq64::mutex::sm_recursive_mutex [static], [private]

Define the static recursive mutex and its condition variable.

12.44.3.2 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").

12.45 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.

12.45.1 Field Documentation

12.45.1.1 unsigned short seq64::editable_event::name_value_t::event_value

12.45.1.2 std::string seq64::editable_event::name_value_t::event_name

12.46 seq64::options Class Reference

This class supports a full tabbed options dialog.

Inherits Dialog.

Public Member Functions

- [options](#) (Gtk::Window &parent, [perform](#) &p)

Private Types

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 [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 [lash_support_callback](#) (Gtk::CheckButton *)
- void [add_midi_clock_page](#) ()
- void [add_midi_input_page](#) ()
- void [add_keyboard_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.

12.46.1 Member Enumeration Documentation

12.46.1.1 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, [rc_settings::with_jack_transport\(\)](#).
- e_jack_master*** Turns on the "with JACK Master" option, [rc_settings::with_jack_master\(\)](#). If another application is already JACK Master, this will fail.
- e_jack_master_cond*** Turns on the "with JACK Master" option [rc_settings::with_jack_master_cond\(\)](#). This option makes sequencer64 the JACK Master conditionally, that is, if no other application has claimed that role.
- e_jack_start_mode_live*** Doesn't directly do anything; the live mode versus song mode is set by the [e_jack_start_mode_song](#) 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 [rc_settings::jack_start_mode\(\)](#).
- e_jack_connect*** Causes the perform object's JACK initialization function, [perform::init_jack\(\)](#), to be called.
- e_jack_disconnect*** Causes the perform object's JACK deinitialization function, [perform::deinit_jack\(\)](#), to be called.

12.46.2 Constructor & Destructor Documentation

12.46.2.1 seq64::options::options (Gtk::Window & *parent*, *perform* & *p*)

12.46.3 Member Function Documentation

12.46.3.1 perform& seq64::options::perf () [inline], [private]

12.46.3.2 void seq64::options::clock_callback_off (int *bus*, Gtk::RadioButton * *button*) [private]

12.46.3.3 void seq64::options::clock_callback_on (int *bus*, Gtk::RadioButton * *button*) [private]

12.46.3.4 void seq64::options::clock_callback_mod (int *bus*, Gtk::RadioButton * *button*) [private]

12.46.3.5 void seq64::options::clock_mod_callback (Gtk::Adjustment * *adj*) [private]

12.46.3.6 void seq64::options::input_callback (int *bus*, Gtk::Button * *button*) [private]

12.46.3.7 void seq64::options::transport_callback (*button type*, Gtk::Button * *button*) [private]

12.46.3.8 void seq64::options::mouse_seq24_callback (Gtk::RadioButton *) [private]

12.46.3.9 void seq64::options::mouse_fruity_callback (Gtk::RadioButton *) [private]

12.46.3.10 void seq64::options::mouse_mod4_callback (Gtk::CheckButton *) [private]

12.46.3.11 void seq64::options::lash_support_callback (Gtk::CheckButton *) [private]

12.46.3.12 void seq64::options::add_midi_clock_page () [private]

12.46.3.13 void seq64::options::add_midi_input_page () [private]

12.46.3.14 void seq64::options::add_keyboard_page () [private]

12.46.3.15 void seq64::options::add_mouse_page () [private]

12.46.3.16 void seq64::options::add_jack_sync_page () [private]

12.46.4 Field Documentation

12.46.4.1 Gtk::Tooltips* seq64::options::m_tooltips [private]

12.46.4.2 perform& seq64::options::m_mainperf [private]

12.46.4.3 Gtk::Button* seq64::options::m_button_ok [private]

12.46.4.4 Gtk::CheckButton* seq64::options::m_button_jack_transport [private]

12.46.4.5 Gtk::CheckButton* seq64::options::m_button_jack_master [private]

12.46.4.6 Gtk::CheckButton* seq64::options::m_button_jack_master_cond [private]

12.46.4.7 Gtk::Button* seq64::options::m_button_jack_connect [private]

12.46.4.8 Gtk::Button* seq64::options::m_button_jack_disconnect [private]

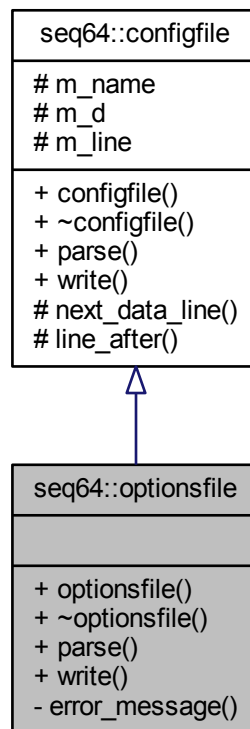
12.46.4.9 Gtk::Notebook* seq64::options::m_notebook [private]

Must be a GTK thang.

12.47 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 §ionname)
Helper function for error-handling.

Additional Inherited Members

12.47.1 Detailed Description

The settings that are passed around are provided or used by the perform class.

12.47.2 Constructor & Destructor Documentation

12.47.2.1 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.
-------------	--

12.47.2.2 seq64::optionsfile::~~optionsfile ()

12.47.3 Member Function Documentation

12.47.3.1 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.

[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.
- jack_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]

This section covers.... 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](#).

12.47.3.2 `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](#).

12.47.3.3 `bool seq64::optionsfile::error_message (const std::string & sectionname) [private]`

Parameters

<i>sectionname</i>	Provides the name of the section for reporting the error.
--------------------	---

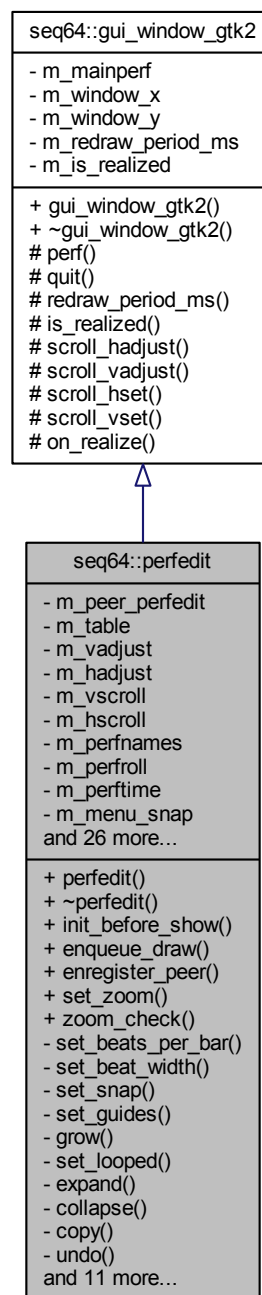
Returns

Always returns false.

12.48 seq64::perfedit Class Reference

This class supports a Performance Editor that is used to arrange the patterns/sequences defined in the patterns panel.

Inheritance diagram for seq64::perfedit:



Public Member Functions

- [perfedit](#) ([perform](#) &p, bool second_perfedit=false, int ppqn=SEQ64_USE_DEFAULT_PPQN)
Principal constructor, has a reference to a perform object.
- virtual [~perfedit](#) ()
This rote constructor does nothing.
- void [init_before_show](#) ()

This function forwards its call to the perfroll function of the same name.

- void [enqueue_draw](#) (bool forward=true)
Helper wrapper for calling perfroll::queue_draw() for one or both perfedits.
- void [enregister_peer](#) (perfedit *peer)
Register the peer perfedit object.
- void [set_zoom](#) (int z)
Implements the horizontal zoom feature.

Static Public Member Functions

- static bool [zoom_check](#) (int z)
Checks zoom values for the z/Z keystrokes used in perfroll and perftime.

Private Member Functions

- void [set_beats_per_bar](#) (int bpm)
Sets the beats-per-measure text and value to the given value, and then calls [set_guides\(\)](#).
- void [set_beat_width](#) (int bw)
Sets the BW (beat width, or the denominator in the time signature) text and values to the given value, and then calls [set_guides\(\)](#).
- void [set_snap](#) (int snap)
Sets the snap text and values to the given value, and then calls [set_guides\(\)](#).
- void [set_guides](#) ()
Sets the guides, which are the L and R user-interface elements.
- void [grow](#) ()
Increments the size of the perfroll and perftime objects.
- void [set_looped](#) ()
Set the looping in the perform object.
- void [expand](#) ()
Implement the expand action.
- void [collapse](#) ()
Implement the collapse action.
- void [copy](#) ()
Implement the copy (actually, expand-and-copy) action.
- void [undo](#) ()
Implement the undo feature (Ctrl-Z).
- void [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_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](#)
- Horizonatl 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::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_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 perfrill (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

12.48.1 Detailed Description

It has a seqroll and piano roll? No, it has a perform, a perfnames, a perfrill, and a perftime.

12.48.2 Constructor & Destructor Documentation

12.48.2.1 `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.

12.48.2.2 `seq64::perfedit::~~perfedit () [virtual]`

We're going to have to run the application through valgrind to make sure that nothing is left behind.

12.48.3 Member Function Documentation

12.48.3.1 `void seq64::perfedit::init_before_show ()`

It does not seem to need to also forward to the perftime function of the same name.

12.48.3.2 `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.
----------------	--

12.48.3.3 `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.
----------	-------------------------------------

12.48.3.4 `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.
-------------	--

12.48.3.5 `void seq64::perfedit::set_zoom (int z)`

12.48.3.6 `void seq64::perfedit::set_beats_per_bar (int bpm) [private]`

The usage of is modified was faulty. Offloaded it to the perform object to make it more foolproof. See the [perform↵::modify\(\)](#) function.

Parameters

<i>bpm</i>	Provides the beats/measure or beats/bar value to be set. This value is basically the numerator of the time signature.
------------	---

12.48.3.7 `void seq64::perfedit::set_beat_width (int bw) [private]`

The usage of is modified was faulty. Offloaded it to the perform object to make it more foolproof. See the [perform↵::modify\(\)](#) function.

Parameters

<i>bw</i>	Provides the beat width to be set. The beat width is basically the denominator of the time signature.
-----------	---

12.48.3.8 `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".
-------------	---

12.48.3.9 `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.

12.48.3.10 void seq64::perfedit::grow () [private]

Make sure that setting the modified flag makes sense for this operation. It doesn't seem to modify members.

12.48.3.11 void seq64::perfedit::set_looped () [private]

12.48.3.12 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 perfrroll to redraw.

12.48.3.13 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 perfrroll to redraw.

12.48.3.14 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 perfrroll to redraw.

12.48.3.15 void seq64::perfedit::undo () [private]

We pop an Undo trigger, and then ask the perfrroll to queue up a (re)drawing action.

12.48.3.16 void seq64::perfedit::popup_menu (Gtk::Menu * *menu*) [private]

12.48.3.17 void seq64::perfedit::draw_sequences () [private]

This is meant to be called when the focus of an open seqedit or eventedit window changes.

12.48.3.18 bool seq64::perfedit::timeout () [private]

It redraws "dirty" sequences in the perfrroll and the perfrnames objects, and shows draw progress on the perfrroll. 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.

12.48.3.19 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.
------------------	---

12.48.3.20 `void seq64::perfedit::start_playing () [private]`

JACK will be used if it is present and, in the application, enabled and working.

12.48.3.21 `void seq64::perfedit::pause_playing () [private]`

Currently, it is just the same as [stop_playing\(\)](#), but we will get it to work. Keeps the stop button enabled as a kind of rewind for ALSA.

12.48.3.22 `void seq64::perfedit::stop_playing () [private]`

We need to make the progress line move back to the beginning right away here.

12.48.3.23 `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.

12.48.3.24 `void seq64::perfedit::on_realize () [private]`

12.48.3.25 `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.
-----------	--------------------------------------

12.48.3.26 `bool seq64::perfedit::on_delete_event (GdkEventAny *) [inline],[private]`

12.48.4 Friends And Related Function Documentation

12.48.4.1 `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).

12.48.5 Field Documentation

12.48.5.1 **perfeddit*** seq64::perfeddit::m_peer_perfeddit [private]

12.48.5.2 **Gtk::Table*** seq64::perfeddit::m_table [private]

Layout table for song editor.

12.48.5.3 **Gtk::Adjustment*** seq64::perfeddit::m_vadjust [private]

12.48.5.4 **Gtk::Adjustment*** seq64::perfeddit::m_hadjust [private]

12.48.5.5 **Gtk::VScrollbar*** seq64::perfeddit::m_vscroll [private]

12.48.5.6 **Gtk::HScrollbar*** seq64::perfeddit::m_hscroll [private]

12.48.5.7 **perfnames*** seq64::perfeddit::m_perfnames [private]

12.48.5.8 **perfroll*** seq64::perfeddit::m_perfroll [private]

12.48.5.9 **perftime*** seq64::perfeddit::m_perftime [private]

12.48.5.10 **Gtk::Menu*** seq64::perfeddit::m_menu_snap [private]

12.48.5.11 **Gtk::Image*** seq64::perfeddit::m_image_play [private]

12.48.5.12 **Gtk::Button*** seq64::perfeddit::m_button_snap [private]

12.48.5.13 **Gtk::Entry*** seq64::perfeddit::m_entry_snap [private]

12.48.5.14 **Gtk::Button*** seq64::perfeddit::m_button_stop [private]

12.48.5.15 **Gtk::Button*** seq64::perfeddit::m_button_play [private]

The Play button object.

12.48.5.16 `Gtk::ToggleButton* seq64::perfedit::m_button_loop` [private]

12.48.5.17 `Gtk::Button* seq64::perfedit::m_button_expand` [private]

12.48.5.18 `Gtk::Button* seq64::perfedit::m_button_collapse` [private]

12.48.5.19 `Gtk::Button* seq64::perfedit::m_button_copy` [private]

12.48.5.20 `Gtk::Button* seq64::perfedit::m_button_grow` [private]

12.48.5.21 `Gtk::Button* seq64::perfedit::m_button_undo` [private]

12.48.5.22 `Gtk::Button* seq64::perfedit::m_button_bpm` [private]

12.48.5.23 `Gtk::Entry* seq64::perfedit::m_entry_bpm` [private]

12.48.5.24 `Gtk::Button* seq64::perfedit::m_button_bw` [private]

12.48.5.25 `Gtk::Entry* seq64::perfedit::m_entry_bw` [private]

12.48.5.26 `Gtk::HBox* seq64::perfedit::m_hbox` [private]

12.48.5.27 `Gtk::HBox* seq64::perfedit::m_hlbox` [private]

12.48.5.28 `Gtk::Tooltips* seq64::perfedit::m_tooltips` [private]

12.48.5.29 `Gtk::Menu* seq64::perfedit::m_menu_bpm` [private]

Drop-down menu for beats/minute.

12.48.5.30 `Gtk::Menu* seq64::perfedit::m_menu_bw` [private]

12.48.5.31 `int seq64::perfedit::m_snap` [private]

12.48.5.32 `int seq64::perfedit::m_bpm` [private]

Do not confuse it with BPM (beats per minute). The numerator of the time signature.

12.48.5.33 `int seq64::perfedit::m_bw` [private]

The denominator of the time signature.

12.48.5.34 int seq64::perfedit::m_ppqn [private]

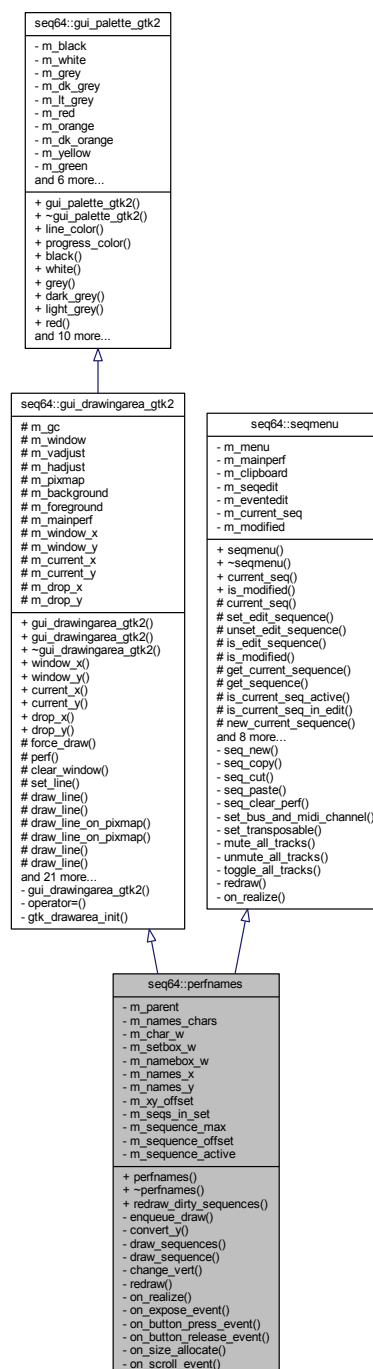
12.48.5.35 bool seq64::perfedit::m_is_running [private]

12.48.5.36 int seq64::perfedit::m_standard_bpm [private]

12.49 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

12.49.1 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).

12.49.2 Constructor & Destructor Documentation

12.49.2.1 `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.

12.49.2.2 `virtual seq64::perfnames::~~perfnames ()` [[inline](#)], [[virtual](#)]

12.49.3 Member Function Documentation

12.49.3.1 `void seq64::perfnames::redraw_dirty_sequences ()`

12.49.3.2 `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.

12.49.3.3 `int seq64::perfnames::convert_y (int y) [private]`

Used in figuring out which sequence to mute/unmute in the performance editor.

Parameters

<code>y</code>	The y value (within the vertical limits of the <code>perfnames</code> column to the left of the performance editor's piano roll.
----------------	--

Returns

Returns the sequence number corresponding to the y value.

12.49.3.4 `void seq64::perfnames::draw_sequences () [private]`

12.49.3.5 `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

<code>seqnum</code>	Index to the sequence information to be drawn.
---------------------	--

12.49.3.6 `void seq64::perfnames::change_vert () [private]`

12.49.3.7 `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](#).

12.49.3.8 `void seq64::perfnames::on_realize () [private]`

It first calls the base-class version of [on_realize\(\)](#). Then it allocates any additional resources needed.

12.49.3.9 `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.

12.49.3.10 `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.

12.49.3.11 `bool seq64::perfnames::on_button_release_event (GdkEventButton * p0)` `[private]`

Parameters

<i>p0</i>	The button event.
-----------	-------------------

Returns

Always returns false.

12.49.3.12 `void seq64::perfnames::on_size_allocate (Gtk::Allocation & a)` `[private]`

It first calls the base-class version of this function.

Parameters

<i>a</i>	The allocation event. It is passed to the base-class on_size_allocate() function, and then <code>m_window_x</code> and <code>m_window_y</code> are set to the width and height, respectively, of the allocation.
----------	--

12.49.3.13 `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

<i>ev</i>	The scrolling event.
-----------	----------------------

Returns

Always returns true.

12.49.4 Friends And Related Function Documentation

12.49.4.1 `friend class perfedit` `[friend]`

12.49.5 Field Documentation

12.49.5.1 `perfedit& seq64::perfnames::m_parent` `[private]`

We want to support two perfedit windows, but the children of perfedit will have to communicate changes requiring a redraw through the parent.

12.49.5.2 `int seq64::perfnames::m_names_chars` `[private]`

Pretty much hardwired to 24 at present.

12.49.5.3 `int seq64::perfnames::m_char_w` `[private]`

This value is obtained from a font-renderer accessor function.

12.49.5.4 `int seq64::perfnames::m_setbox_w` `[private]`

This used to be hardwired to $6 * 2$ (character-width times two).

12.49.5.5 `int seq64::perfnames::m_namebox_w` `[private]`

This used to be a weird calculation based on character width.

12.49.5.6 `int seq64::perfnames::m_names_x` `[private]`

12.49.5.7 `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.

12.49.5.8 `int seq64::perfnames::m_xy_offset` `[private]`

Currently hardwired.

12.49.5.9 `const int seq64::perfnames::m_seqs_in_set` `[private]`

12.49.5.10 `const int seq64::perfnames::m_sequence_max` `[private]`

12.49.5.11 `int seq64::perfnames::m_sequence_offset` `[private]`

12.49.5.12 `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.

12.50 `seq64::perform` Class Reference

This class supports the performance mode.

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 [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](#) (int bpm)
'Setter' function for member m_beats_per_bar
- int [get_beat_width](#) () const
'Getter' function for member m_beat_width
- void [set_beat_width](#) (int bw)
'Setter' function for member m_beat_width
- 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
- bool [is_running](#) () const
'Getter' function for member m_running Could also be called "is_playing()".
- 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_paused](#) () const
'Getter' function for member m_is_paused
- bool [is_pausable](#) () const
'Getter' function for member m_is_paused and ! m_jack_asst.is_running() We might just make this internal.
- void [enregister](#) ([performcallback](#) *pfcfb)

- Adds a pointer to an object to be notified by this perform object.*

 - void `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`
- `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`
- 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`
- 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` ()

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 `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 seq)
Retrieves a reference to a value from `m_midi_cc_toggle[]`.
- `midi_control` & `midi_control_on` (int seq)
Retrieves a reference to a value from `m_midi_cc_on[]`.
- `midi_control` & `midi_control_off` (int seq)
Retrieves a reference to a value from `m_midi_cc_off[]`.
- void `handle_midi_control` (int control, bool state)
Handle the MIDI Control values that provide some automation for the application.
- 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 ¬e)
Copies the given string into `m_screen_set_notepad[]`.
- void `set_screen_set_notepad` (const std::string ¬e)
Sets the notepad text for the current screen-set.
- int `get_screenset` () const
'Getter' function for member `m_screenset`
- void `set_playing_screenset` ()
Sets the screen set that is active, based on the value of `m_screenset`.
- void `set_screenset` (int ss)
Sets the `m_screenset` value (the index or ID of the current screen set).
- int `get_playing_screenset` () const
'Getter' function for member `m_playing_screen`
- bool `any_group_unmutes` () const
'Getter' function for member `m_mute_group[]` Returns true if there are any unmute statuses in the mute-group array.
- 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_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` ()
- 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.
- 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)
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.
- 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_active` (int seq) const
Checks the pattern/sequence for activity.
- `sequence * get_sequence` (int seq)
Retrieves the actual sequence, based on the pattern/sequence number.
- 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` (int bpm)
Sets the value of the BPM into the master MIDI buss, after making sure it is squelched to be between 20 and 500.
- int `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_looping` (bool looping)
'Setter' function for member `m_looping`
- 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.
- 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.*
- 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_sequence_key` Provides access to [keys\(\).show_ui_sequence_key\(\)](#).*
- void [show_ui_sequence_key](#) (bool flag)
 - 'Setter' function for member `m_show_ui_sequence_key`*
- bool [show_ui_sequence_number](#) () const
 - 'Getter' function for member `m_show_ui_sequence_number` Provides access to [keys\(\).show_ui_sequence_number\(\)](#).*
- void [show_ui_sequence_number](#) (bool flag)
 - 'Getter' function for member `m_show_ui_sequence_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](#) ()
 - Pause playback, so that progress bars stay where they are, and playback always resumes where it left off, even in ALSA mode.*
- 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.*
- int [decrement_beats_per_minute](#) ()
 - Encapsulates some calls used in mainwnd.*
- int [increment_beats_per_minute](#) ()
 - Encapsulates some calls used in mainwnd.*
- 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.*
- 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.*
- bool [mainwnd_key_event](#) (const [keystroke](#) &k)
 - Provided for mainwnd :: on_key_press_event() and mainwnd :: on_key_release_event() to call.*
- bool [perffroll_key_event](#) (const [keystroke](#) &k, int drop_sequence)
 - Provided for perffroll :: on_key_press_event() and perffroll :: 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.*

Private Member Functions

- 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 \(\)](#)

Initializes JACK support, if SEQ64_JACK_SUPPORT is defined.
 - [bool deinit_jack \(\)](#)

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 is_midi_control_valid \(int seq\) const](#)

Checks the parameter against `c_midi_controls`.
 - [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 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 \(\)](#)

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_all_key_events \(\)](#)

Pass-along function for [keys\(\).set_all_key_events](#).
 - [void set_all_key_groups \(\)](#)

Pass-along function for [keys\(\).set_all_key_events](#).
 - [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.

Private Attributes

- [gui_assistant](#) & [m_gui_support](#)

Support for a wide range of GUI-related operations.
- [bool m_mute_group \[c_gmute_tracks\]](#)

Mute group support.
- [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.
- [mastermidibus](#) [m_master_bus](#)

Provides our MIDI buss.
- 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](#)
- 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.
- 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.
- `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_is_paused`
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]`
Provides the settings of MIDI Toggle, as read from the "rc" file.
- `midi_control m_midi_cc_on [c_midi_controls]`
Provides the settings of MIDI On, as read from the "rc" file.
- `midi_control m_midi_cc_off [c_midi_controls]`
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_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.*
- `std::vector< performcallback * > m_notify`

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 [midifile](#)
- class [optionsfile](#)
- class [options](#)
- int [jack_sync_callback](#) ([jack_transport_state_t](#) state, [jack_position_t](#) *pos, void *arg)
- *Global functions for JACK support and JACK sessions.*

12.50.1 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.

12.50.2 Constructor & Destructor Documentation

12.50.2.1 `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.

Parameters

<i>mygui</i>	Provides access to the GUI assistant that holds many things, including the containers of keys and the "events" they provide. This is a base-class reference; for a real class, see the gui_assistant_gtk2 class in the seq_gtkmm2 GUI-specific library. Note that we access the <code>m_gui_support</code> member using the gui() accessor function.
<i>ppqn</i>	The default, choosable, or actual PPQN value.

12.50.2.2 seq64::perform::~~perform ()

Finally, any active or inactive (but allocated) patterns/sequences are deleted, and their pointers nullified.

12.50.3 Member Function Documentation

12.50.3.1 bool seq64::perform::is_modified () const [inline]

12.50.3.2 void seq64::perform::modify () [inline]

The setter that will, [is_modified\(\)](#), is private. No one but perform and its friends should falsify this flag.

12.50.3.3 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.

12.50.3.4 int seq64::perform::sequence_max () const [inline]

12.50.3.5 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.

12.50.3.6 void seq64::perform::set_edit_sequence (int *seqnum*) [inline]

Parameters

<i>seqnum</i>	Pass in -1 to disable the edit-sequence number unconditionally. Use unset_edit_sequence() to disable it if it matches the current edit-sequence number.
---------------	---

12.50.3.7 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.
---------------	---

12.50.3.8 bool seq64::perform::is_edit_sequence (int *seqnum*) const [inline]

Parameters

<i>seqnum</i>	Tests the parameter against <code>m_edit_sequence</code> . Returns true if that member is not -1, and the parameter matches it.
---------------	---

12.50.3.9 `int seq64::perform::get_beats_per_bar () const` `[inline]`

12.50.3.10 `void seq64::perform::set_beats_per_bar (int bpm)` `[inline]`

Parameters

<i>bpm</i>	Provides the value for beats/measure. Also used to set the beats/measure in the JACK assistant object.
------------	--

12.50.3.11 `int seq64::perform::get_beat_width () const` `[inline]`

12.50.3.12 `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.
-----------	--

12.50.3.13 `const gui_assistant& seq64::perform::gui () const` `[inline]`

12.50.3.14 `gui_assistant& seq64::perform::gui ()` `[inline]`

12.50.3.15 `const keys_perform& seq64::perform::keys () const` `[inline]`

12.50.3.16 `keys_perform& seq64::perform::keys ()` `[inline]`

12.50.3.17 `mastermidibus& seq64::perform::master_bus ()` `[inline]`

12.50.3.18 `bool seq64::perform::is_running () const` `[inline]`

12.50.3.19 `bool seq64::perform::is_jack_running () const` `[inline]`

12.50.3.20 `bool seq64::perform::is_paused () const` `[inline]`

12.50.3.21 `bool seq64::perform::is_pausable () const` `[inline]`

12.50.3.22 `void seq64::perform::enregister (performcallback * pfc)` `[inline]`

Parameters

<i>pfc</i>	Provides the pointer to the performance callback.
------------	---

12.50.3.23 void 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.

12.50.3.24 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.

Parameters

<i>ppqn</i>	Provides the PPQN value, which is either the default value (192) or is read from the "user" configuration file.
-------------	---

12.50.3.25 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 mainwnd 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.
------------	--

12.50.3.26 void seq64::perform::add_sequence (sequence * seq, int prefnun)

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 prefnun 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 prefnun?

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.

12.50.3.27 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.
------------	---

12.50.3.28 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.

12.50.3.29 void seq64::perform::clear_sequence_triggers (int seq)**Parameters**

<i>seq</i>	Provides the desired sequence. The is_active() function validates this value.
------------	---

12.50.3.30 void seq64::perform::print_triggers () const**12.50.3.31 void seq64::perform::finish () [inline]**

A minor simplification for the `main()` routine, hides the JACK support macro.

12.50.3.32 `midipulse seq64::perform::get_tick () const [inline]`

12.50.3.33 `midipulse seq64::perform::get_jack_tick () const [inline]`

12.50.3.34 `void seq64::perform::set_jack_tick (midipulse tick) [inline]`

Parameters

<i>tick</i>	Provides the current JACK tick (pulse) value to set.
-------------	--

12.50.3.35 `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" ($m_ppqn * 4$) 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.

12.50.3.36 `midipulse seq64::perform::get_left_tick () const [inline]`

12.50.3.37 `void seq64::perform::set_start_tick (midipulse tick) [inline]`

Parameters

<i>tick</i>	Provides the starting JACK tick (pulse) value to set.
-------------	---

12.50.3.38 `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.

12.50.3.39 `midipulse seq64::perform::get_right_tick () const [inline]`

12.50.3.40 `void seq64::perform::move_triggers (bool direction)`

Parameters

<i>direction</i>	Specifies the desired direction; false = left, true = right.
------------------	--

12.50.3.41 `void seq64::perform::copy_triggers ()`

This copies the triggers between the L marker and R marker to the R marker.

12.50.3.42 `void seq64::perform::push_trigger_undo ()`

Too bad we cannot yet keep track of all the undoes for the sake of properly handling the "is modified" flag.

12.50.3.43 `void seq64::perform::pop_trigger_undo ()`

12.50.3.44 `void seq64::perform::split_trigger (int seqnum, midipulse tick)`

Parameters

<i>seqnum</i>	Indicates the sequence that needs to have its trigger split.
<i>tick</i>	The MIDI pulse number at which the trigger should be split.

12.50.3.45 `midipulse seq64::perform::get_max_trigger ()`

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?

12.50.3.46 `void seq64::perform::collapse ()` `[inline]`

12.50.3.47 `void seq64::perform::copy ()` `[inline]`

12.50.3.48 `void seq64::perform::expand ()` `[inline]`

12.50.3.49 `midi_control & seq64::perform::midi_control_toggle (int seq)`

Parameters

<i>seq</i>	Provides the index to pass to <code>is_midi_control_valid()</code> to obtain a control value (such as <code>c_midi_control_bpm_up</code>) to use to retrieve the desired <code>midi_control</code> object. Note that this value is unsigned simply to make the legality check of the parameter easier.
------------	---

Returns

Returns the "toggle" value if the sequence is valid, and a reference to sm_mc_dummy otherwise.

12.50.3.50 midi_control & seq64::perform::midi_control_on (int seq)**Parameters**

<i>seq</i>	Provides the index to pass to is_midi_control_valid() to obtain a control value (such as c_midi_control_bpm_up) to use to retrieve the desired midi_control object.
------------	--

Returns

Returns the "on" value if the sequence is valid, and a reference to sm_mc_dummy otherwise.

12.50.3.51 midi_control & seq64::perform::midi_control_off (int seq)**Parameters**

<i>seq</i>	Provides a control value (such as c_midi_control_bpm_up) to use to retrieve the desired midi_control object.
------------	---

Returns

Returns the "off" value if the sequence is valid, and a reference to sm_mc_dummy otherwise.

12.50.3.52 void seq64::perform::handle_midi_control (int ctrl, bool state)**Parameters**

<i>ctrl</i>	The MIDI control value to use to perform an operation.
<i>state</i>	The state of the control, used with the following values:

```
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
```

12.50.3.53 const std::string & seq64::perform::get_screen_set_notepad (int screenset) const**Parameters**

<i>screenset</i>	The ID number of the string 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.

12.50.3.54 `const std::string& seq64::perform::current_screen_set_notepad () const` `[inline]`

12.50.3.55 `void seq64::perform::set_screen_set_notepad (int screenset, const std::string & notepad)`

Parameters

<i>screenset</i>	The ID number of the string set, an index into the <code>m_screen_set_xxx[]</code> arrays.
<i>notepad</i>	Provides the string data 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.

12.50.3.56 `void seq64::perform::set_screen_set_notepad (const std::string & note)` `[inline]`

Parameters

<i>note</i>	The string value to set into the notepad text.
-------------	--

12.50.3.57 `int seq64::perform::get_screenset () const` `[inline]`

12.50.3.58 `void seq64::perform::set_playing_screenset ()`

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.

12.50.3.59 `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.
-----------	---

12.50.3.60 `int seq64::perform::get_playing_screenset () const [inline]`

12.50.3.61 `bool seq64::perform::any_group_unmutes () const`

If they're all zero, we don't need to save them.

12.50.3.62 `void seq64::perform::mute_group_tracks ()`

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`.

Change Note 2016-05-06 It seems to us that the for (i) clause should have i range from 0 to `m_max_sets`, not `m_seqs_in_set`. So let's do it, pre-emptively.

12.50.3.63 `void seq64::perform::select_and_mute_group (int group)`

Called in `perform` and in `mainwnd`.

Parameters

<code>group</code>	Provides the group number for the group to be muted.
--------------------	--

12.50.3.64 `void seq64::perform::set_mode_group_mute () [inline]`

12.50.3.65 `void seq64::perform::unset_mode_group_mute () [inline]`

12.50.3.66 `void seq64::perform::select_group_mute (int mutegroup)`

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

<code>mutegroup</code>	The number of the desired mute group, clamped to be between 0 and <code>m_seqs_in_set-1</code> . Obviously, it is the set whose state is to be stored, if in group-learn mode.
------------------------	--

12.50.3.67 `void seq64::perform::set_mode_group_learn ()`

This function is called via a MIDI control `c_midi_control_mod_glearn` and via the group-learn keystroke.

12.50.3.68 `void seq64::perform::unset_mode_group_learn ()`

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.

12.50.3.69 `bool seq64::perform::is_group_learning ()` `[inline]`

12.50.3.70 `void seq64::perform::set_and_copy_mute_group (int mutegroup)`

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.
------------------------	------------------------------------

12.50.3.71 `void seq64::perform::start (bool state)`

Parameters

<code>state</code>	What does this state mean?
--------------------	----------------------------

12.50.3.72 `void seq64::perform::stop ()`

The logic seems backward here, in that we call `inner_stop()` if JACK is not running. Or perhaps we misunderstand the meaning of `m_jack_running`?

12.50.3.73 `void seq64::perform::start_jack ()` `[inline]`

12.50.3.74 `void seq64::perform::stop_jack ()` `[inline]`

12.50.3.75 `void seq64::perform::position_jack (bool state)`

12.50.3.76 `void seq64::perform::off_sequences ()`

12.50.3.77 `void seq64::perform::all_notes_off ()`

Then flush the master MIDI buss.

12.50.3.78 void seq64::perform::set_active (int *seq*, bool *active*)

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.

12.50.3.79 void seq64::perform::set_was_active (int *seq*)

Why do we need this routine?

Parameters

<i>seq</i>	The pattern number. It is checked for validity.
------------	---

12.50.3.80 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.

12.50.3.81 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.

12.50.3.82 bool seq64::perform::is_dirty_perf (int *seq*)

Parameters

<code>seq</code>	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.

12.50.3.83 `bool seq64::perform::is_dirty_names (int seq)`

Parameters

<code>seq</code>	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.

12.50.3.84 `bool seq64::perform::is_active (int seq) const` `[inline]`

Todo We should have the sequence object keep track of its own activity and access that via a reference or pointer.

Parameters

<code>seq</code>	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.

12.50.3.85 `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.

12.50.3.86 void seq64::perform::reset_sequences (bool *pause* = false)

Note that these calls are folded into one member function of the sequence class. Finally, flush the master MIDI buss.

12.50.3.87 void seq64::perform::play (midipulse *tick*)

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.

Parameters

<i>tick</i>	Provides the tick at which to start playing.
-------------	--

12.50.3.88 void seq64::perform::set_orig_ticks (midipulse *tick*)

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.
-------------	--

12.50.3.89 void seq64::perform::set_beats_per_minute (int *bpm*)

Replaces perform::set_bpm() from seq24.

The value is set only if neither JACK nor this performance object are running.

It's not clear that we need to set the "is modified" flag just because we changed the beats per minute. This setting does get saved to the MIDI file, with the c_bpmtag.

Parameters

<i>bpm</i>	Provides the beats/minute value to be set. It is clamped, if necessary, between the values SEQ64_MINIMUM_BPM to SEQ64_MAXIMUM_BPM. They provide a wide range of speeds, well beyond what normal music needs.
------------	--

12.50.3.90 int seq64::perform::get_beats_per_minute () [inline]

Returns

Returns the value of beats/minute from the master buss.

12.50.3.91 void seq64::perform::set_looping (bool *looping*) [inline]

Parameters

<i>looping</i>	The boolean value to set for looping, used in the performance editor.
----------------	---

12.50.3.92 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.
---------------	------------------------

12.50.3.93 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.
---------------	------------------------

12.50.3.94 void seq64::perform::sequence_playing_toggle (int *seq*)

If the m_control_status is c_status_queue, then the sequence's toggle_queued() function is called. 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.

Parameters

<i>seq</i>	The sequence number of the sequence to be potentially toggled.
------------	--

12.50.3.95 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.

12.50.3.96 void seq64::perform::sequence_playing_on (int *seq*) `[inline]`

Parameters

<code>seq</code>	The sequence number of the sequence to turn on.
------------------	---

12.50.3.97 `void seq64::perform::sequence_playing_off (int seq) [inline]`

Parameters

<code>seq</code>	The sequence number of the sequence to turn off.
------------------	--

12.50.3.98 `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

<code>flag</code>	If true (the default), the song-mute of the sequence is turned on. Otherwise, it is turned off.
-------------------	---

12.50.3.99 `void seq64::perform::toggle_all_tracks ()`

Covers tracks from 0 to `m_sequence_max`.

Parameters

<code>flag</code>	If true (the default), the song-mute of the sequence is turned on. Otherwise, it is turned off.
-------------------	---

12.50.3.100 `void seq64::perform::mute_screensset (int ss, bool flag = true)`

Parameters

<code>flag</code>	If true (the default), the song-mute of the sequence is turned on. Otherwise, it is turned off.
-------------------	---

12.50.3.101 `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.

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.

12.50.3.102 `void seq64::perform::input_func ()`

12.50.3.103 `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.

12.50.3.104 `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 set_group_mute_state() , this value is offset by adding <code>m_mute_group_selected * m_seqs_in_set</code> .
---------------	--

Returns

Returns the desired `m_mute_group[]` value.

12.50.3.105 `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.
---------------	---------------------

12.50.3.106 `int seq64::perform::get_offset () const [inline]`

12.50.3.107 `void seq64::perform::save_playing_state ()`

Inactive patterns get the value set to false. Used in unsetting the snapshot status (c_status_snapshot).

12.50.3.108 `void seq64::perform::restore_playing_state ()`

Used in unsetting the snapshot status (c_status_snapshot).

12.50.3.109 `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.
----------	--

12.50.3.110 `keys_perform::SlotMap& seq64::perform::get_key_events () [inline]`

12.50.3.111 `keys_perform::SlotMap& seq64::perform::get_key_groups () [inline]`

12.50.3.112 `keys_perform::RevSlotMap& seq64::perform::get_key_events_rev () [inline]`

12.50.3.113 `keys_perform::RevSlotMap& seq64::perform::get_key_groups_rev () [inline]`

12.50.3.114 `bool seq64::perform::show_ui_sequence_key () const [inline]`

Used in mainwid, options, optionsfile, userfile, and perform.

12.50.3.115 `void seq64::perform::show_ui_sequence_key (bool flag) [inline]`

Parameters

<i>flag</i>	Provides the flag to set into keys().show_ui_sequence_key() .
-------------	---

12.50.3.116 `bool seq64::perform::show_ui_sequence_number () const [inline]`

Used in mainwid, optionsfile, and perform.

12.50.3.117 `void seq64::perform::show_ui_sequence_number (bool flag) [inline]`

Parameters

<i>flag</i>	Provides the value to set into keys().show_ui_sequence_number() .
-------------	---

12.50.3.118 `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.

12.50.3.119 `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.

12.50.3.120 `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.

12.50.3.121 `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.

12.50.3.122 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.

Note

It would be nice to know why the following code snippet disables the mute/unmute functionality of the performance/song editor:

```
position_jack(false);
start_jack();
start(false);
```

The `jack_assistant::position()` function doesn't use the boolean parameter at present; that code is effectively disabled. Okay, now it does, if the "relocate" parameter is true. See `perform::position_jack()` and `jack_assistant::position()`. This parameter, when true, allows the "klick" application to get proper position data.

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.

Parameters

<i>songmode</i>	Indicates if the caller wants to start the playback in JACK mode. In the seq42 (yes, "42", not "24") 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. If the previous state was "paused", then we start it in Live mode, which works because Song mode has already turned on the sequences. This method is not quite intuitive, because it is really following Live mode.
-----------------	---

12.50.3.123 void seq64::perform::pause_playing ()

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.

12.50.3.124 `void seq64::perform::stop_playing ()`

Stops playback, turns off the (new) `m_is_paused` 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)..

12.50.3.125 `void seq64::perform::start_key (bool songmode = false)`

Meant to be used by GUIs to unify the treatment of keys versus buttons.

Parameters

<i>songmode</i>	The live/play mode parameter to be passed along to the key processor. Defaults to false (live mode).
-----------------	--

12.50.3.126 `void seq64::perform::pause_key (bool songmode = false)`

Meant to be used by GUIs to unify the treatment of keys versus buttons.

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).
-----------------	--

12.50.3.127 `void seq64::perform::stop_key ()`

Meant to be used by GUIs to unify the treatment of keys versus buttons.

12.50.3.128 `void seq64::perform::learn_toggle ()` `[inline]`

12.50.3.129 `int seq64::perform::decrement_beats_per_minute ()` `[inline]`

Actually does a lot of work in those function calls.

12.50.3.130 `int seq64::perform::increment_beats_per_minute ()` `[inline]`

Actually does a lot of work in those function calls.

12.50.3.131 `int seq64::perform::decrement_screenset ()` `[inline]`

12.50.3.132 `int seq64::perform::increment_screenset ()` `[inline]`

12.50.3.133 `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.
------------	---

12.50.3.134 `bool seq64::perform::is_smf_0 (const sequence & seq) const` `[inline]`

Parameters

<i>seq</i>	Provides a reference to the desired sequence.
------------	---

12.50.3.135 `void seq64::perform::sequence_key (int seq)`

This function is use in mainwnd when toggling the mute/unmute setting using keyboard keys.

Parameters

<i>seq</i>	The sequence's control-key number, which is relative to the current screen-set.
------------	---

12.50.3.136 `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.

12.50.3.137 `void seq64::perform::set_input_bus (int bus, bool active)`

This function is called by `options::input_callback()`.

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.

12.50.3.138 `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.

12.50.3.139 `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 not yet supported).

Returns

Returns true if the key was handled.

12.50.3.140 `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.

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.

12.50.3.141 `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.

12.50.3.142 `void seq64::perform::launch_input_thread () [private]`

This might be a good candidate for a small thread class derived from a small base class.

12.50.3.143 `void seq64::perform::launch_output_thread () [private]`

This might be a good candidate for a small thread class derived from a small base class.

12.50.3.144 `bool seq64::perform::init_jack () [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.

12.50.3.145 `bool seq64::perform::deinit_jack () [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.

12.50.3.146 `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.

12.50.3.147 `void seq64::perform::is_modified (bool flag) [inline], [private]`

Parameters

<i>flag</i>	The value of the modified flag to be set.
-------------	---

12.50.3.148 `bool seq64::perform::is_midi_control_valid (int seq) const [inline], [private]`

Parameters

<i>seq</i>	The value that should be in the <code>c_midi_controls</code> range.
------------	---

Returns

Returns true if the parameter is valid. For this function, no error print-out is generated.

12.50.3.149 `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.

12.50.3.150 void seq64::perform::set_running (bool *running*) [inline], [private]

Parameters

<i>running</i>	The value of the running flag to be set.
----------------	--

12.50.3.151 void seq64::perform::set_playback_mode (bool *playbackmode*) [inline], [private]

Parameters

<i>playbackmode</i>	The value of the playback mode flag to be set.
---------------------	--

12.50.3.152 int seq64::perform::mute_group_offset (int *track*) [inline], [private]

Parameters

<i>track</i>	The number of the desired track.
--------------	----------------------------------

12.50.3.153 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, <code>m_sequence_max</code>).
------------	--

Returns

Returns true if the sequence number is valid.

12.50.3.154 bool seq64::perform::is_mseq_valid (int *seq*) const [private]

It also evaluates the `m_seq[seq]` pointer value.

Note

Since we can have holes in the sequence array, where there are inactive sequences, we check if the sequence is even active before emitting a message about a null pointer for the sequence. We only want to see messages that indicate actual problems.

Parameters

<i>seq</i>	Provides the sequence number to be checked. It is checked for validity. We cannot compare the sequence number versus the sequence_count() , because the current implementation can have inactive holes (with null pointers) interspersed with active pointers.
------------	--

Returns

Returns true if the sequence number is valid as per [is_seq_valid\(\)](#), and the sequence pointer is not null.

12.50.3.155 `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.

12.50.3.156 `void seq64::perform::inner_start (bool state) [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. WE STILL HAVE TO EVALUATE WHAT SIDE-EFFECTS MIGHT
OCCUR. ALSO CONSIDER A RUN-TIME --pause-support option for this
feature.
```

Parameters

<i>state</i>	Sets the playback mode, and, if true, turns off all of the sequences.
--------------	---

12.50.3.157 `void seq64::perform::inner_stop () [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 remove to near the end of the sequence.

12.50.3.158 `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.

12.50.3.159 `void seq64::perform::set_all_key_events () [inline],[private]`

12.50.3.160 `void seq64::perform::set_all_key_groups () [inline],[private]`

12.50.3.161 `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.

12.50.3.162 `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.

12.50.4 Friends And Related Function Documentation

12.50.4.1 friend class `jack_assistant` `[friend]`

12.50.4.2 friend class `keybindentry` `[friend]`

12.50.4.3 friend class `midifile` `[friend]`

12.50.4.4 friend class `optionsfile` `[friend]`

12.50.4.5 friend class `options` `[friend]`

12.50.4.6 `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.

Parameters

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

Returns

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

12.50.5 Field Documentation

12.50.5.1 `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.

12.50.5.2 `gui_assistant& seq64::perform::m_gui_support` `[private]`

12.50.5.3 `bool seq64::perform::m_mute_group[c_gmute_tracks]` `[private]`

This value determines whether a particular track will be muted or unmuted, and it can handle all tracks available in the application (currently 1024). Note that the current state of playing can be "learned", and stored herein as the desired state for the track.

12.50.5.4 `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.

12.50.5.5 `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.

12.50.5.6 `bool seq64::perform::m_mode_group_learn [private]`

12.50.5.7 `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.

12.50.5.8 `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)`.

12.50.5.9 `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.

12.50.5.10 `sequence* seq64::perform::m_seqs[c_max_sequence] [private]`

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

12.50.5.11 `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.

12.50.5.12 `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?

12.50.5.13 `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.

12.50.5.14 `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.

12.50.5.15 `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?

12.50.5.16 `bool seq64::perform::m_sequence_state[c_max_sequence]` `[private]`

12.50.5.17 `mastermidibus seq64::perform::m_master_bus` `[private]`

12.50.5.18 `pthread_t seq64::perform::m_out_thread` `[private]`

Provides a "handle" to the output thread.

12.50.5.19 `pthread_t seq64::perform::m_in_thread` `[private]`

12.50.5.20 `bool seq64::perform::m_out_thread_launched` `[private]`

12.50.5.21 `bool seq64::perform::m_in_thread_launched` `[private]`

12.50.5.22 `bool seq64::perform::m_running` `[private]`

12.50.5.23 `bool seq64::perform::m_inputting` `[private]`

12.50.5.24 `bool seq64::perform::m_outputting` `[private]`

12.50.5.25 `bool seq64::perform::m_looping` `[private]`

If true, the performance will loop between the L and R markers in the performance editor.

12.50.5.26 `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
```

12.50.5.27 `int seq64::perform::m_ppqn` `[private]`

12.50.5.28 `int seq64::perform::m_beats_per_bar` `[private]`

The default value is SEQ64_DEFAULT_BEATS_PER_MEASURE (4).

12.50.5.29 `int seq64::perform::m_beat_width` `[private]`

The default value is SEQ64_DEFAULT_BEAT_WIDTH (4).

12.50.5.30 `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.

12.50.5.31 `midipulse seq64::perform::m_left_tick` `[private]`

Note that "tick" is actually "pulses".

12.50.5.32 `midipulse seq64::perform::m_right_tick` `[private]`

Note that "tick" is actually "pulses".

12.50.5.33 `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".

12.50.5.34 `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.

12.50.5.35 `midipulse seq64::perform::m_jack_tick` [private]

12.50.5.36 `bool seq64::perform::m_usemidiclock` [private]

12.50.5.37 `bool seq64::perform::m_midiclockrunning` [private]

12.50.5.38 `int seq64::perform::m_midiclocktick` [private]

12.50.5.39 `int seq64::perform::m_midiclockpos` [private]

12.50.5.40 `bool seq64::perform::m_is_paused` [private]

12.50.5.41 `std::string seq64::perform::m_screen_set_notepad[c_max_sets]` [private]

12.50.5.42 `midi_control seq64::perform::m_midi_cc_toggle[c_midi_controls]` [private]

12.50.5.43 `midi_control seq64::perform::m_midi_cc_on[c_midi_controls]` [private]

12.50.5.44 `midi_control seq64::perform::m_midi_cc_off[c_midi_controls]` [private]

12.50.5.45 `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.

12.50.5.46 `int seq64::perform::m_control_status` [private]

Need to learn more about this one. It is used in the replace, snapshot, and queue functionality.

12.50.5.47 `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.

12.50.5.48 `int seq64::perform::m_seqs_in_set` [private]

This change will require some arrays to be dynamically allocated (vectors).

12.50.5.49 `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.

12.50.5.50 `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_max_sequence_max`, because there can be inactive sequences amidst the active sequences.

12.50.5.51 `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?

12.50.5.52 `int seq64::perform::m_edit_sequence` `[private]`

Moving this status from seqmenu into perform for better centralized management.

12.50.5.53 `bool seq64::perform::m_is_modified` `[private]`

All the GUIs seem to use a perform object.

12.50.5.54 `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.

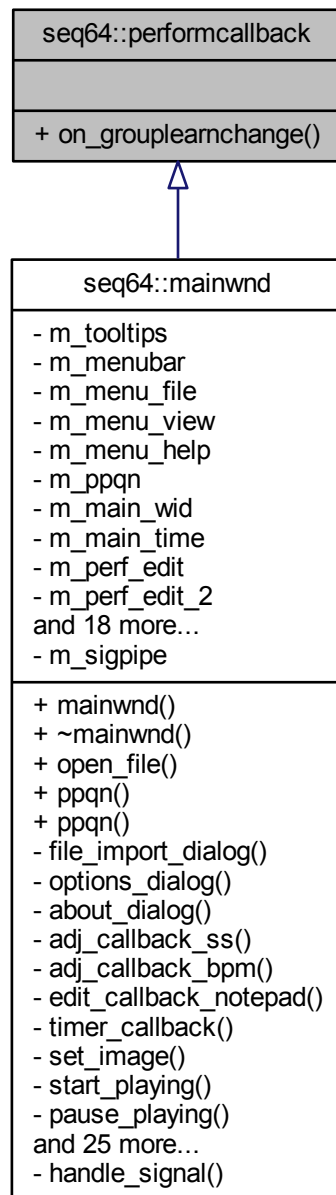
12.50.5.55 `jack_assistant seq64::perform::m_jack_asst` `[private]`

12.50.5.56 `std::vector<performcallback*> seq64::perform::m_notify` `[private]`

12.51 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.

12.51.1 Detailed Description

Provide a response to a group-learn change event.

12.51.2 Member Function Documentation

12.51.2.1 virtual void seq64::performcallback::on_grouplearnchange (bool) [inline],[virtual]

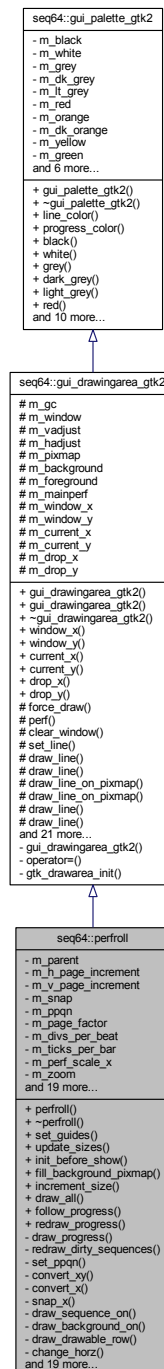
"state" is an Unused parameter.

Reimplemented in [seq64::mainwnd](#).

12.52 seq64::perfroll Class Reference

This class implements the performance roll user interface.

Inheritance diagram for seq64::perffroll:



Public Member Functions

- [perffroll](#) ([perform](#) &[perf](#), [perfedit](#) &parent, Gtk::Adjustment &hadjust, Gtk::Adjustment &vadjust, int ppqn=S↵EQ64_USE_DEFAULT_PPQN)

Principal constructor.

- virtual [~perffroll](#) ()

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 a tick-offset on the x coordinate.
- 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 4-bar 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 `perfedt`, so that it can forward it to any other `perfedt` that exists.
- void [set_zoom](#) (int z)
Implements the horizontal zoom feature.
- 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

- [perfeddit](#) & [m_parent](#)
Provides a link to the `perfeddit` 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.
- int [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 perfrroll names box, in pixes.
- int [m_background_x](#)
The width of the perfrroll background.
- int [m_size_box_w](#)
This is a basically constant value set to `s_perfrroll_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.
- int [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.
- 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::perfrroll_key_event()`.
- class [Seq24PerfInput](#)
- class [perfedit](#)

Additional Inherited Members

12.52.1 Constructor & Destructor Documentation

12.52.1.1 `seq64::perftroll::perftroll (perform & perf, perftedit & parent, Gtk::Adjustment & hadjust, Gtk::Adjustment & vadjust, int ppqn = SEQ64_USE_DEFAULT_PPQN)`

12.52.1.2 `seq64::perftroll::~~perftroll ()` [virtual]

Well, now there are two objects, so no explicit deletion necessary.

12.52.2 Member Function Documentation

12.52.2.1 `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 perftedit::set_guides() . This is actually equal to the measure-pulses divided by the snap value in perftedit; the snap value defaults to 8.
<i>measure</i>	Provides the number of measure-pulses (pulses per measure) as calculated in perftedit::set_guides() .
<i>beat</i>	Provides the number of beat-pulses (pulses per beat) as calculated in perftedit::set_guides() .

12.52.2.2 `void seq64::perftroll::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 perftroll 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.

12.52.2.3 `void seq64::perftroll::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.

12.52.2.4 void seq64::perftroll::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.

12.52.2.5 void seq64::perftroll::increment_size ()

12.52.2.6 void seq64::perftroll::draw_all ()

12.52.2.7 void seq64::perftroll::follow_progress ()

12.52.2.8 void seq64::perftroll::redraw_progress () [inline]

12.52.2.9 void seq64::perftroll::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 `perftroll::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.

12.52.2.10 void seq64::perftroll::redraw_dirty_sequences () [private]

Change Note ca 2016-05-30 Lets try not drawing sequences greater than the maximum, at all.

12.52.2.11 void seq64::perftroll::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_perftroll::scale_x` member starts out at `c_perftroll::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_perftroll::scale_x` constant.

Todo Resolve the issue of `c_perftroll::scale_x` versus `m_perftroll::scale_x` in `perftroll`.

12.52.2.12 void seq64::perftroll::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.

12.52.2.13 `void seq64::perfroll::convert_x (int x, midipulse & tick) [private]`

The result is returned via the tick parameter.

12.52.2.14 `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 `mod = m_snap/m_perf_scale_x` equals the number pixels to snap to.

12.52.2.15 `void seq64::perfroll::draw_sequence_on (int seqnum) [private]`

Statement nesting from hell!

12.52.2.16 `void seq64::perfroll::draw_background_on (int seqnum) [private]`

12.52.2.17 `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 as `y` casted to an `int`, then the drawing of the row is only partial, vertically.

12.52.2.18 `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.

12.52.2.19 `void seq64::perfroll::change_vert () [private]`

12.52.2.20 `void seq64::perfroll::split_trigger (int sequence, midipulse tick) [private]`

12.52.2.21 `void seq64::perfroll::enqueue_draw () [private]`

The parent `perfedit` will call `perfroll::queue_draw()` on behalf of this object, and it will pass a [perfroll::enqueue_draw\(\)](#) to the peer `perfedit`'s `perfroll`, if the peer exists.

12.52.2.22 void seq64::perfrroll::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.

12.52.2.23 void seq64::perfrroll::horizontal_adjust (double *step*) [inline],[private]

A duplicate of the one in seqroll.

Parameters

<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. See gui_drawingarea_gtk2::scroll_hadjust() for more information.
-------------	---

12.52.2.24 `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 gui_drawingarea_gtk2::scroll_vadjust() for more information.
-------------	---

12.52.2.25 `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).
--------------	---

12.52.2.26 `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).
--------------	---

12.52.2.27 `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.

12.52.2.28 `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

<code>ev</code>	Provides the expose event.
-----------------	----------------------------

Returns

Always returns true.

12.52.2.29 `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.

12.52.2.30 `bool seq64::perfroll::on_button_release_event (GdkEventButton * ev) [private]`

This gives us Seq24 versus Fruity behavior.

12.52.2.31 `bool seq64::perfroll::on_motion_notify_event (GdkEventMotion * ev) [private]`

12.52.2.32 `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

<code>ev</code>	Provides the scroll event.
-----------------	----------------------------

Returns

Currently always returns true.

12.52.2.33 `bool seq64::perfroll::on_focus_in_event (GdkEventFocus * ev) [private]`

12.52.2.34 `bool seq64::perfroll::on_focus_out_event (GdkEventFocus * ev) [private]`

12.52.2.35 `void seq64::perfroll::on_size_allocate (Gtk::Allocation & a1) [private]`

```
12.52.2.36 bool seq64::perfroll::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::perfroll_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 `perfroll_key_event()` function.

The `perfroll_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. 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 `perfroll_key_event()` if the Ctrl key was pressed. Be careful.

```
12.52.2.37 void seq64::perfroll::on_size_request ( GtkRequisition * ) [inline],[private]
```

12.52.3 Friends And Related Function Documentation

```
12.52.3.1 friend class FruityPerfInput [friend]
```

The `perfedit` class needs access to the private `enqueue_draw()` function.

```
12.52.3.2 friend class Seq24PerfInput [friend]
```

```
12.52.3.3 friend class perfedit [friend]
```

12.52.4 Field Documentation

```
12.52.4.1 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.

```
12.52.4.2 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.

12.52.4.3 `int seq64::perffroll::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.

12.52.4.4 `int seq64::perffroll::m_snap` `[private]`

12.52.4.5 `int seq64::perffroll::m_ppqn` `[private]`

12.52.4.6 `int seq64::perffroll::m_page_factor` `[private]`

12.52.4.7 `int seq64::perffroll::m_divs_per_beat` `[private]`

12.52.4.8 `int seq64::perffroll::m_ticks_per_bar` `[private]`

12.52.4.9 `int seq64::perffroll::m_perf_scale_x` `[private]`

12.52.4.10 `int seq64::perffroll::m_zoom` `[private]`

It seems to work pretty well now.

12.52.4.11 `int seq64::perffroll::m_names_y` `[private]`

This is currently semantically a constant set to `c_names_y = 24`.

12.52.4.12 `int seq64::perffroll::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?)

12.52.4.13 `int seq64::perffroll::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.

12.52.4.14 `int seq64::perffroll::m_measure_length` `[private]`

12.52.4.15 `int seq64::perffroll::m_beat_length` `[private]`

12.52.4.16 `midipulse seq64::perffroll::m_old_progress_ticks` `[private]`

See the [draw_progress\(\)](#) function. This could almost be static inside that function.

12.52.4.17 `int 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.

12.52.4.18 `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.

12.52.4.19 `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\(\)](#).

12.52.4.20 `midipulse seq64::perfroll::m_drop_tick` `[private]`

Used only by the friend modules `perfroll_input` and `fruityperfroll_input`.

12.52.4.21 `midipulse seq64::perfroll::m_drop_tick_trigger_offset` `[private]`

Used only by the friend modules `perfroll_input` and `fruityperfroll_input`.

12.52.4.22 `int seq64::perfroll::m_drop_sequence` `[private]`

Used for redrawing the sequence.

12.52.4.23 `int seq64::perfroll::m_sequence_max` `[private]`

12.52.4.24 `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.

12.52.4.25 `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`

12.52.4.26 Seq24PerfInput seq64::perfroll::m_seq24_interaction [private]

12.52.4.27 bool seq64::perfroll::m_moving [private]

12.52.4.28 bool seq64::perfroll::m_growing [private]

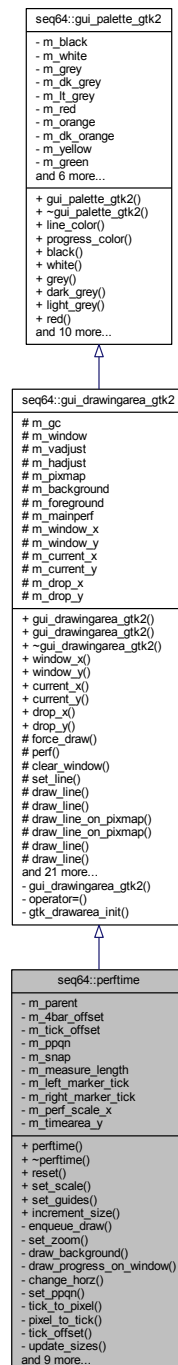
12.52.4.29 bool seq64::perfroll::m_grow_direction [private]

Determines whether the section is growing to the left or to the right.

12.53 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_PPQN)↵

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 `perfdit`, so that it can forward it to any other `perfdit` 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

- [perfeddit](#) & [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

12.53.1 Constructor & Destructor Documentation

12.53.1.1 `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.

12.53.1.2 `virtual seq64::perftime::~~perftime () [inline],[virtual]`

12.53.2 Member Function Documentation

12.53.2.1 `void seq64::perftime::reset ()`

12.53.2.2 `void seq64::perftime::set_scale (int scale)`

12.53.2.3 `void seq64::perftime::set_guides (int snap, int measure)`

This function then fills in the background, and queues up a draw operation.

Parameters

<i>snap</i>	Provides the number of snap-pulses (pulses per snap interval) as calculated in perfedit::set_guides() . This is actually equal to the measure-pulses divided by the snap value in perfedit; the snap value defaults to 8.
<i>measure</i>	Provides the number of measure-pulses (pulses per measure) as calculated in perfedit::set_guides() .

12.53.2.4 `void seq64::perftime::increment_size () [inline]`

Compare it to [perftroll::increment_size\(\)](#).

12.53.2.5 `void seq64::perftime::enqueue_draw () [private]`

The parent perfedit will call `perftime::queue_draw()` on behalf of this object, and it will pass a [perftime::enqueue_↵draw\(\)](#) to the peer perfedit's perftime, if the peer exists.

12.53.2.6 `void seq64::perftime::set_zoom (int z) [private]`

Redraws the background if the new zoom checked out.

Parameters

<i>z</i>	Provides the zoom value, which is checked, and then copied into <code>m_perf_scale_x</code> .
----------	---

12.53.2.7 `void seq64::perftime::draw_background () [private]`

Note that `m_measure_length == 0` will cause integer overflow.

12.53.2.8 `void seq64::perftime::draw_progress_on_window () [private]`

12.53.2.9 `void seq64::perftime::change_horz () [private]`

Again, uses the constant, 16 [now offloaded to the new [tick_offset\(\)](#) function.].

12.53.2.10 `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.
-------------	----------------------------------

12.53.2.11 `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.

12.53.2.12 `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.

12.53.2.13 `int seq64::perftime::tick_offset () [inline],[private]`

Returns

Returns `m_4bar_offset * 16 * m_ppqn`.

12.53.2.14 void seq64::perftime::update_sizes () [inline],[private]

12.53.2.15 int seq64::perftime::idle_progress () [inline],[private]

12.53.2.16 void seq64::perftime::update_pixmap () [inline],[private]

12.53.2.17 void seq64::perftime::draw_pixmap_on_window () [inline],[private]

12.53.2.18 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);
```

12.53.2.19 bool seq64::perftime::on_expose_event (GdkEventExpose * *ev*) [private]

Redraws the background.

Note

The perfedit 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.

12.53.2.20 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).

12.53.2.21 `void seq64::perftime::on_size_allocate (Gtk::Allocation & r) [private]`

12.53.2.22 `bool seq64::perftime::on_button_release_event (GdkEventButton *) [inline],[private]`

"ev", The button event parameter, is not used.

Returns

Always returns false

12.53.2.23 `bool seq64::perftime::key_press_event (GdkEventKey * ev) [private]`

Can't get the keystroke events to be seen by perftroll 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.

12.53.3 Friends And Related Function Documentation

12.53.3.1 `friend class perfedit [friend]`

12.53.4 Field Documentation

12.53.4.1 `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.

12.53.4.2 `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.

12.53.4.3 `int seq64::perftime::m_tick_offset` `[private]`

Why 16?

12.53.4.4 `int seq64::perftime::m_ppqn` `[private]`

12.53.4.5 `int seq64::perftime::m_snap` `[private]`

12.53.4.6 `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.

12.53.4.7 `int seq64::perftime::m_left_marker_tick` `[private]`

Otherwise it is -1. Supports perftime's keystroke processing.

12.53.4.8 `int seq64::perftime::m_right_marker_tick` `[private]`

Otherwise it is -1. Supports perftime's keystroke processing.

12.53.4.9 `int seq64::perftime::m_perf_scale_x` `[private]`

12.53.4.10 `int seq64::perftime::m_timearea_y` `[private]`

12.54 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
Accessor m_auto_option_save
- void [auto_option_save](#) (bool flag)
- bool [legacy_format](#) () const
Accessor m_legacy_format
- void [legacy_format](#) (bool flag)
- bool [lash_support](#) () const
Accessor m_lash_support
- void [lash_support](#) (bool flag)
- bool [allow_mod4_mode](#) () const
Accessor m_allow_mod4_mode
- void [allow_mod4_mode](#) (bool flag)
- bool [show_midi](#) () const
Accessor m_show_midi
- void [show_midi](#) (bool flag)
- bool [priority](#) () const
Accessor m_priority
- void [priority](#) (bool flag)
- bool [stats](#) () const
Accessor m_stats
- void [stats](#) (bool flag)
- bool [pass_sysex](#) () const
Accessor m_pass_sysex
- void [pass_sysex](#) (bool flag)
- bool [with_jack_transport](#) () const
Accessor m_with_jack_transport
- void [with_jack_transport](#) (bool flag)
- bool [with_jack_master](#) () const
Accessor m_with_jack_master
- void [with_jack_master](#) (bool flag)
- bool [with_jack_master_cond](#) () const
Accessor m_with_jack_master_cond
- void [with_jack_master_cond](#) (bool flag)
- bool [with_jack](#) () const
Accessor m_with_jack_transport m_with_jack_master, and m_with_jack_master_cond, to save client code some trouble.
- bool [jack_start_mode](#) () const
Accessor m_jack_start_mode,
- void [jack_start_mode](#) (bool flag)
- bool [manual_alsa_ports](#) () const
Accessor m_manual_alsa_ports
- void [manual_alsa_ports](#) (bool flag)
- bool [reveal_alsa_ports](#) () const
Accessor m_reveal_alsa_ports
- void [reveal_alsa_ports](#) (bool flag)
- bool [is_pattern_playing](#) () const
Accessor m_is_pattern_playing
- void [is_pattern_playing](#) (bool flag)
- bool [print_keys](#) () const
Accessor m_print_keys
- void [print_keys](#) (bool flag)
- bool [device_ignore](#) () const

Accessor *m_device_ignore*

- void [device_ignore](#) (bool flag)
- 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
- 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](#)
- bool [m_legacy_format](#)
- bool [m_lash_support](#)
- bool [m_allow_mod4_mode](#)
- bool [m_show_midi](#)
- bool [m_priority](#)
- bool [m_stats](#)
- bool [m_pass_sysex](#)
- bool [m_with_jack_transport](#)
- bool [m_with_jack_master](#)
- bool [m_with_jack_master_cond](#)
- bool [m_jack_start_mode](#)
- bool [m_manual_alsa_ports](#)
- bool [m_reveal_alsa_ports](#)
- bool [m_is_pattern_playing](#)
- bool [m_print_keys](#)
- bool [m_device_ignore](#)
- int [m_device_ignore_num](#)
- [interaction_method_t](#) [m_interaction_method](#)
- std::string [m_filename](#)
Provides the name of current MIDI file.
- std::string [m_jack_session_uuid](#)
- std::string [m_last_used_dir](#)
- std::string [m_config_directory](#)
- std::string [m_config_filename](#)
- std::string [m_user_filename](#)
- std::string [m_config_filename_alt](#)
- std::string [m_user_filename_alt](#)

12.54.1 Constructor & Destructor Documentation

12.54.1.1 `seq64::rc_settings::rc_settings ()`

12.54.1.2 `seq64::rc_settings::rc_settings (const rc_settings & rhs)`

Parameters

<i>rhs</i>	The source of the data for the copy.
------------	--------------------------------------

12.54.2 Member Function Documentation

12.54.2.1 `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.

12.54.2.2 `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.

12.54.2.3 `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.

12.54.2.4 `void seq64::rc_settings::set_defaults ()`

12.54.2.5 `bool seq64::rc_settings::auto_option_save () const` `[inline]`

12.54.2.6 `void seq64::rc_settings::auto_option_save (bool flag)` `[inline]`

12.54.2.7 `bool seq64::rc_settings::legacy_format () const` `[inline]`

12.54.2.8 `void seq64::rc_settings::legacy_format (bool flag)` `[inline]`

12.54.2.9 `bool seq64::rc_settings::lash_support () const` `[inline]`

12.54.2.10 `void seq64::rc_settings::lash_support (bool flag)` `[inline]`

12.54.2.11 `bool seq64::rc_settings::allow_mod4_mode () const` `[inline]`

12.54.2.12 `void seq64::rc_settings::allow_mod4_mode (bool flag)` `[inline]`

12.54.2.13 `bool seq64::rc_settings::show_midi () const` `[inline]`

12.54.2.14 `void seq64::rc_settings::show_midi (bool flag)` `[inline]`

12.54.2.15 `bool seq64::rc_settings::priority () const` `[inline]`

12.54.2.16 `void seq64::rc_settings::priority (bool flag)` `[inline]`

12.54.2.17 `bool seq64::rc_settings::stats () const` `[inline]`

12.54.2.18 void seq64::rc_settings::stats (bool *flag*) [inline]

12.54.2.19 bool seq64::rc_settings::pass_sysex () const [inline]

12.54.2.20 void seq64::rc_settings::pass_sysex (bool *flag*) [inline]

12.54.2.21 bool seq64::rc_settings::with_jack_transport () const [inline]

12.54.2.22 void seq64::rc_settings::with_jack_transport (bool *flag*) [inline]

12.54.2.23 bool seq64::rc_settings::with_jack_master () const [inline]

12.54.2.24 void seq64::rc_settings::with_jack_master (bool *flag*) [inline]

12.54.2.25 bool seq64::rc_settings::with_jack_master_cond () const [inline]

12.54.2.26 void seq64::rc_settings::with_jack_master_cond (bool *flag*) [inline]

12.54.2.27 bool seq64::rc_settings::with_jack () const [inline]

12.54.2.28 bool seq64::rc_settings::jack_start_mode () const [inline]

12.54.2.29 void seq64::rc_settings::jack_start_mode (bool *flag*) [inline]

12.54.2.30 bool seq64::rc_settings::manual_alsa_ports () const [inline]

12.54.2.31 void seq64::rc_settings::manual_alsa_ports (bool *flag*) [inline]

12.54.2.32 bool seq64::rc_settings::reveal_alsa_ports () const [inline]

12.54.2.33 void seq64::rc_settings::reveal_alsa_ports (bool *flag*) [inline]

12.54.2.34 bool seq64::rc_settings::is_pattern_playing () const [inline]

12.54.2.35 void seq64::rc_settings::is_pattern_playing (bool *flag*) [inline]

12.54.2.36 bool seq64::rc_settings::print_keys () const [inline]

12.54.2.37 void seq64::rc_settings::print_keys (bool *flag*) [inline]

12.54.2.38 bool seq64::rc_settings::device_ignore () const [inline]

12.54.2.39 void seq64::rc_settings::device_ignore (bool *flag*) [inline]

12.54.2.40 int seq64::rc_settings::device_ignore_num () const [inline]

12.54.2.41 `interaction_method_t seq64::rc_settings::interaction_method () const` `[inline]`

12.54.2.42 `const std::string& seq64::rc_settings::filename () const` `[inline]`

12.54.2.43 `const std::string& seq64::rc_settings::jack_session_uuid () const` `[inline]`

12.54.2.44 `const std::string& seq64::rc_settings::last_used_dir () const` `[inline]`

12.54.2.45 `const std::string& seq64::rc_settings::config_directory () const` `[inline]`

12.54.2.46 `const std::string& seq64::rc_settings::config_filename () const` `[inline]`

12.54.2.47 `const std::string& seq64::rc_settings::user_filename () const` `[inline]`

12.54.2.48 `const std::string& seq64::rc_settings::config_filename_alt () const` `[inline]`

12.54.2.49 `const std::string& seq64::rc_settings::user_filename_alt () const` `[inline]`

12.54.2.50 `void seq64::rc_settings::device_ignore_num (int value)`

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

12.54.2.51 `void seq64::rc_settings::interaction_method (interaction_method_t value)`

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

12.54.2.52 `void seq64::rc_settings::filename (const std::string & value)`

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

12.54.2.53 `void seq64::rc_settings::jack_session_uuid (const std::string & value)`

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

12.54.2.54 `void seq64::rc_settings::last_used_dir (const std::string & value)`

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

12.54.2.55 `void seq64::rc_settings::config_directory (const std::string & value)`

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

12.54.2.56 `void seq64::rc_settings::set_config_files (const std::string & value)`

Implements the `--config` option to change both configuration files ("rc" and "usr") with one option.

Parameters

<i>value</i>	The value to use to make the setting, if the string is not empty. If the value has an extension, it is stripped first.
--------------	--

12.54.2.57 `void seq64::rc_settings::config_filename (const std::string & value)`

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.
--------------	---

12.54.2.58 `void seq64::rc_settings::user_filename (const std::string & value)`

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.
--------------	--

12.54.2.59 `void seq64::rc_settings::config_filename_alt (const std::string & value)`

Parameters

<i>value</i>	The value to use to make the setting, if the string is not empty.
--------------	---

12.54.2.60 `void seq64::rc_settings::user_filename_alt (const std::string & value)`

Parameters

<i>value</i>	The value to use to make the setting.
--------------	---------------------------------------

12.54.2.61 `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.

12.54.3 Field Documentation

12.54.3.1 `bool seq64::rc_settings::m_auto_option_save` [private]

12.54.3.2 `bool seq64::rc_settings::m_legacy_format` [private]

12.54.3.3 `bool seq64::rc_settings::m_lash_support` [private]

12.54.3.4 `bool seq64::rc_settings::m_allow_mod4_mode` [private]

12.54.3.5 `bool seq64::rc_settings::m_show_midi` [private]

12.54.3.6 `bool seq64::rc_settings::m_priority` [private]

12.54.3.7 `bool seq64::rc_settings::m_stats` [private]

12.54.3.8 `bool seq64::rc_settings::m_pass_sysex` [private]

12.54.3.9 `bool seq64::rc_settings::m_with_jack_transport` [private]

12.54.3.10 `bool seq64::rc_settings::m_with_jack_master` [private]

12.54.3.11 `bool seq64::rc_settings::m_with_jack_master_cond` [private]

12.54.3.12 `bool seq64::rc_settings::m_jack_start_mode` [private]

12.54.3.13 `bool seq64::rc_settings::m_manual_alsa_ports` [private]

12.54.3.14 `bool seq64::rc_settings::m_reveal_alsa_ports` [private]

12.54.3.15 `bool seq64::rc_settings::m_is_pattern_playing` [private]

12.54.3.16 `bool seq64::rc_settings::m_print_keys` [private]

```

12.54.3.17  bool seq64::rc_settings::m_device_ignore  [private]

12.54.3.18  int seq64::rc_settings::m_device_ignore_num  [private]

12.54.3.19  interaction_method_t seq64::rc_settings::m_interaction_method  [private]

12.54.3.20  std::string seq64::rc_settings::m_filename  [private]

12.54.3.21  std::string seq64::rc_settings::m_jack_session_uuid  [private]

12.54.3.22  std::string seq64::rc_settings::m_last_used_dir  [private]

12.54.3.23  std::string seq64::rc_settings::m_config_directory  [private]

12.54.3.24  std::string seq64::rc_settings::m_config_filename  [private]

12.54.3.25  std::string seq64::rc_settings::m_user_filename  [private]

12.54.3.26  std::string seq64::rc_settings::m_config_filename_alt  [private]

12.54.3.27  std::string seq64::rc_settings::m_user_filename_alt  [private]

```

12.55 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.

12.55.1 Field Documentation

```

12.55.1.1  int seq64::rect::x

12.55.1.2  int seq64::rect::y

12.55.1.3  int seq64::rect::height

12.55.1.4  int seq64::rect::width

```

12.56 seq64::gui_drawingarea_gtk2::rect Struct Reference

A small helper structure representing a rectangle.

Data Fields

- int [x](#)
- int [y](#)
- int [height](#)
- int [width](#)

12.56.1 Field Documentation

12.56.1.1 int seq64::gui_drawingarea_gtk2::rect::x

12.56.1.2 int seq64::gui_drawingarea_gtk2::rect::y

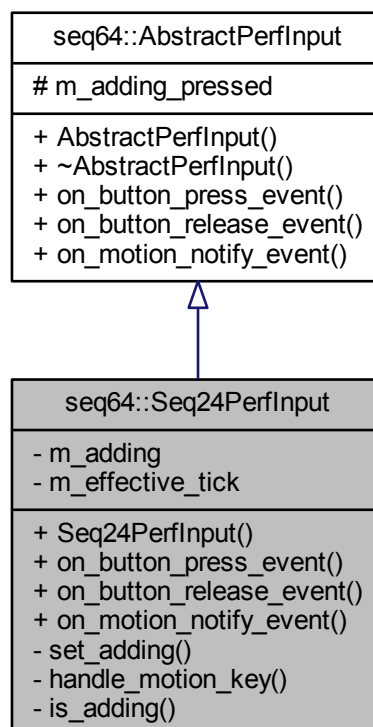
12.56.1.3 int seq64::gui_drawingarea_gtk2::rect::height

12.56.1.4 int seq64::gui_drawingarea_gtk2::rect::width

12.57 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

- void [set_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.
- bool [is_adding](#) () const
'Getter' function for member m_adding

Private Attributes

- bool [m_adding](#)
Indicates we are in the middle of adding a sequence segment to the performance.
- [midipulse](#) [m_effective_tick](#)
The current tick for the current segment?

Friends

- class [perfroll](#)

Additional Inherited Members

12.57.1 Constructor & Destructor Documentation

12.57.1.1 [seq64::Seq24PerfInput::Seq24PerfInput](#) () [\[inline\]](#)

12.57.2 Member Function Documentation

12.57.2.1 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?

Returns

Returns true if a modification occurred.

Implements [seq64::AbstractPerfInput](#).

12.57.2.2 `bool seq64::Seq24Perflnput::on_button_release_event (GdkEventButton * ev, perffroll & roll)` [virtual]

Any use for the middle-button or ctrl-left-click we can add?

Returns

Returns true if any modification occurred.

Implements [seq64::AbstractPerflnput](#).

12.57.2.3 `bool seq64::Seq24Perflnput::on_motion_notify_event (GdkEventMotion * ev, perffroll & roll)` [virtual]

Returns

Returns true if a modification occurs. This function used to always return true.

Implements [seq64::AbstractPerflnput](#).

12.57.2.4 `void seq64::Seq24Perflnput::set_adding (bool adding, perffroll & roll)` [private]

What does it mean?

12.57.2.5 `bool seq64::Seq24Perflnput::handle_motion_key (bool is_left, perffroll & roll)` [private]

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.

12.57.2.6 `bool seq64::Seq24Perflnput::is_adding () const [inline], [private]`

12.57.3 Friends And Related Function Documentation

12.57.3.1 `friend class perflroll [friend]`

12.57.4 Field Documentation

12.57.4.1 `bool seq64::Seq24Perflnput::m_adding [private]`

12.57.4.2 `midipulse seq64::Seq24Perflnput::m_effective_tick [private]`

12.58 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.

12.58.1 Constructor & Destructor Documentation

12.58.1.1 `seq64::Seq24SeqEventInput::Seq24SeqEventInput () [inline]`

12.58.2 Member Function Documentation

12.58.2.1 `void seq64::Seq24SeqEventInput::set_adding (bool adding, sequevent & seqev)`

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.

12.58.2.2 `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(); ???????`

12.58.2.3 `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.

12.58.2.4 `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.

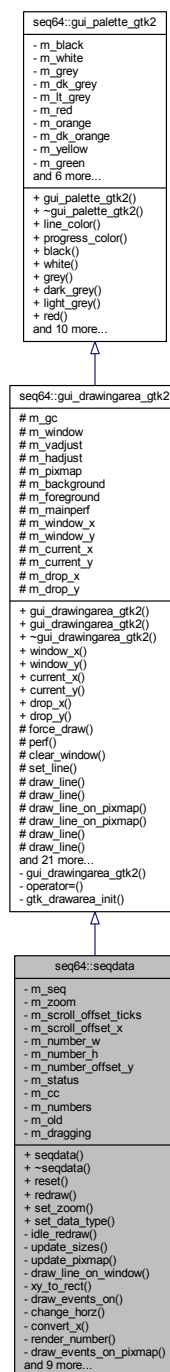
12.58.3 Field Documentation

12.58.3.1 bool seq64::Seq24SeqEventInput::m_adding

12.59 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_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 [seqroll](#)
- class [sequevent](#)

Additional Inherited Members

12.59.1 Constructor & Destructor Documentation

12.59.1.1 `seq64::seqdata::seqdata (sequence & seq, perform & p, int zoom, Gtk::Adjustment & hadjust)`

In the constructor one can only allocate colors, `get_window()` returns 0 because this pane has not yet been realized.

Parameters

<i>seq</i>	The sequence that is being displayed and edited by this data pane.
<i>p</i>	The performance object that oversees all of the sequences. This object is needed here only to access the perform::modify() function.
<i>zoom</i>	The starting zoom of this pane.
<i>hadjust</i>	The horizontal adjustment object provided by the parent class, seqedit, that created this pane.

12.59.1.2 `virtual seq64::seqdata::~~seqdata () [inline],[virtual]`

12.59.2 Member Function Documentation

12.59.2.1 `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.

12.59.2.2 `void seq64::seqdata::redraw () [inline]`

12.59.2.3 `void seq64::seqdata::set_zoom (int z)`

Called by [seqedit::set_zoom\(\)](#), which validates the zoom value.

Parameters

<i>z</i>	The zoom value to be set.
----------	---------------------------

12.59.2.4 `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.

12.59.2.5 `int seq64::seqdata::idle_redraw () [private]`

This drawing is done only if there is no dragging in progress, to guarantee no flicker.

12.59.2.6 `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!

12.59.2.7 `void seq64::seqdata::update_pixmap () [private]`

12.59.2.8 `void seq64::seqdata::draw_line_on_window () [private]`

12.59.2.9 `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

	<code>x1</code>	The input x value for the first data point.
	<code>y1</code>	The input y value for the first data point.
	<code>x2</code>	The input x value for the second data point.
	<code>y2</code>	The input y value for the second data point.
out	<code>rx</code>	The output for the x value of the XWin rectangle.
out	<code>ry</code>	The output for the y value of the XWin rectangle.
out	<code>rw</code>	The output for the width value of the XWin rectangle.
out	<code>rh</code>	The output for the height of the XWin rectangle.

12.59.2.10 `void seq64::seqdata::draw_events_on (Glib::RefPtr< Gdk::Drawable > drawable) [private]`

Very similar to `sequevent::draw_events_on()`. And yet it doesn't handle zooming as well, must fix!

Change Note ca 2016-04-13, 2016-05-24 We now draw the data line for selected event in dark orange, instead of black.

Parameters

<code>drawable</code>	The given drawable object.
-----------------------	----------------------------

12.59.2.11 `void seq64::seqdata::change_horz () [private]`

Basically identical to `sequevent::change_horz()`.

12.59.2.12 `void seq64::seqdata::convert_x (int x, midipulse & tick) [inline], [private]`

12.59.2.13 `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.

12.59.2.14 `void seq64::seqdata::draw_events_on_pixmap () [inline],[private]`

12.59.2.15 `void seq64::seqdata::draw_pixmap_on_window () [inline],[private]`

12.59.2.16 `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).

12.59.2.17 `bool seq64::seqdata::on_expose_event (GdkEventExpose * ev) [private]`

Parameters

<i>ev</i>	Provides the expose-event.
-----------	----------------------------

Returns

Always returns true.

12.59.2.18 `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.

12.59.2.19 `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.

12.59.2.20 `bool seq64::seqdata::on_motion_notify_event (GdkEventMotion * ev) [private]`

It converts the x,y of the mouse to ticks, then sets the events in the event-data-range, updates the pixmap, draws events in the window, and draws a line on the window.

Parameters

<i>ev</i>	The motion event.
-----------	-------------------

Returns

Returns true if a change in event data occurred. If true, then the perform modification flag is set.

12.59.2.21 `bool seq64::seqdata::on_leave_notify_event (GdkEventCrossing * ev) [private]`

Parameter "p0", the crossing point for the event, is unused.

12.59.2.22 `bool seq64::seqdata::on_scroll_event (GdkEventScroll * ev) [private]`

This scroll event only handles basic scrolling, without any modifier keys such as the Ctrl or Shift masks. If there is a note (seqroll pane) or event (sequevent pane) selected, and mouse hovers over the data area (seqdata pane), then this scrolling action will increase or decrease the value of the data item, which lengthens or shortens the line drawn.

Todo DOCUMENT the seqdata scrolling behavior in the documentation projects.

Parameters

<i>ev</i>	Provides the scroll-event.
-----------	----------------------------

Returns

Always returns true.

12.59.2.23 `void seq64::seqdata::on_size_allocate (Gtk::Allocation & r) [private]`

Parameters

<code>r</code>	Provides the allocation event.
----------------	--------------------------------

12.59.3 Friends And Related Function Documentation

12.59.3.1 friend class `seqroll` [`friend`]12.59.3.2 friend class `sequevent` [`friend`]

12.59.4 Field Documentation

12.59.4.1 `sequence& seq64::seqdata::m_seq` [`private`]12.59.4.2 `int seq64::seqdata::m_zoom` [`private`]

the unit is ticks/pixel.

12.59.4.3 `int seq64::seqdata::m_scroll_offset_ticks` [`private`]

Adjusted in the [change_horz\(\)](#) function.

12.59.4.4 `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..

12.59.4.5 `int seq64::seqdata::m_number_w` [`private`]

By "adjusted", well this is just a minor tweak for appearances.

12.59.4.6 `int seq64::seqdata::m_number_h` [`private`]

Basically, the character height times 3. By "adjusted", well this is just a minor tweak for appearances.

12.59.4.7 `int seq64::seqdata::m_number_offset_y` [`private`]

This value was hardwired as 8, for a character height of 10.

12.59.4.8 `midibyte seq64::seqdata::m_status` [`private`]12.59.4.9 `midibyte seq64::seqdata::m_cc` [`private`]12.59.4.10 `Glib::RefPtr<Gdk::Pixmap> seq64::seqdata::m_numbers[c_dataarea_y]` [`private`]

This array is filled only once, in the [on_realize\(\)](#) function.

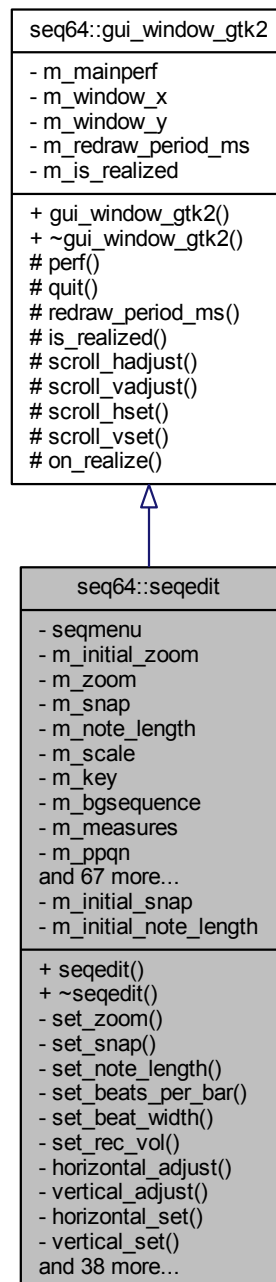
12.59.4.11 GdkRectangle seq64::seqdata::m_old [private]

12.59.4.12 bool seq64::seqdata::m_dragging [private]

12.60 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_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](#) (int 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)
Selects the given MIDI channel parameter in the main sequence object, so that it will use that channel.
- void [set_midi_bus](#) (int midibus)
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_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 [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 [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_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).
- [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::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_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::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::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`

Additional Inherited Members

12.60.1 Detailed Description

- `perform`
- `seqroll`
- `seqkeys`
- `seqdata`
- `seqtime`
- `seqevent`
- `sequence`

This class has a metric ton of user-interface objects and other members.

12.60.2 Constructor & Destructor Documentation

12.60.2.1 `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.

Todo Support the highlight feature in one or both perfedit windows in the same way it is done in the mainwid.

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()`.

12.60.2.2 `seq64::seqedit::~seqedit ()` [virtual]

12.60.3 Member Function Documentation

12.60.3.1 `void seq64::seqedit::set_zoom (int z)` [private]

It is passed to the seqroll, seqtime, seqdata, and sequevent objects, as well. This function doesn't check if the zoom will change, because this function might be used to initialize the zoom of the children.

The notation for zoom in the user-interface is in pixels:ticks, but I would prefer to use pulses/pixel (pulses per pixel). Oh well. Note that this value of zoom is saved to the "user" configuration file when Sequencer64 exit.

Parameters

<code>z</code>	The prospective zoom value to set. It is applied only if between the minimum and maximum allowed zoom values, inclusive. See the usr().min_zoom() and usr().max_zoom() function.
----------------	--

12.60.3.2 `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

<code>s</code>	The prospective snap value to set. It is checked only to make sure it is greater than 0, to avoid a numeric exception.
----------------	--

12.60.3.3 `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

<code>notelength</code>	Provides the note length in units of MIDI pulses.
-------------------------	---

12.60.3.4 `void seq64::seqedit::set_beats_per_bar (int bpm)` [private]

12.60.3.5 void seq64::seqedit::set_beat_width (int *bw*) [private]

12.60.3.6 void seq64::seqedit::set_rec_vol (int *recvol*) [inline],[private]

Parameters

<i>recvol</i>	The setting to be made, obtained from the recording-volume ("Vol") menu.
---------------	--

12.60.3.7 void seq64::seqedit::horizontal_adjust (double *step*) [inline],[private]

A duplicate of the one in seqroll.

Parameters

<i>step</i>	Provides the step value to use for adjusting the horizontal scrollbar. See gui_drawingarea_gtk2::scroll_hadjust() for more information.
-------------	---

12.60.3.8 void seq64::seqedit::vertical_adjust (double *step*) [inline],[private]

A near-duplicate of the one in seqroll.

Parameters

<i>step</i>	Provides the step value to use for adjusting the vertical scrollbar. See gui_drawingarea_gtk2::scroll_vadjust() for more information.
-------------	---

12.60.3.9 void seq64::seqedit::horizontal_set (double *value*) [inline],[private]

Parameters

<i>value</i>	The desired position. Mostly this is either 0.0 or 9999999.0 (an "infinite" value to select the start or end position).
--------------	---

12.60.3.10 void seq64::seqedit::vertical_set (double *value*) [inline],[private]

Parameters

<i>value</i>	The desired position. Mostly this is either 0.0 or 9999999.0 (an "infinite" value to select the start or end position).
--------------	---

12.60.3.11 void seq64::seqedit::set_measures (int *lim*) [private]

Parameters

<i>lim</i>	Provides the sequence length, in measures.
------------	--

12.60.3.12 void seq64::seqedit::apply_length (int *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 sequevent objects are reset().

12.60.3.13 long seq64::seqedit::get_measures () [private]

Todo Create a [sequence::set_units\(\)](#) function or a [sequence::get_measures\(\)](#) function to forward to.

12.60.3.14 void seq64::seqedit::set_midi_channel (int *midichannel*) [private]

Should this change set the is-modified flag? Where should validation occur?

12.60.3.15 void seq64::seqedit::set_midi_bus (int *bus*) [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.

12.60.3.16 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\(\)](#).

12.60.3.17 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\(\)](#).

12.60.3.18 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\(\)](#).

12.60.3.19 `void seq64::seqedit::name_change_callback () [private]`

That name is the name the user has given to the sequence being edited.

12.60.3.20 `void seq64::seqedit::play_change_callback () [private]`

12.60.3.21 `void seq64::seqedit::record_change_callback () [private]`

12.60.3.22 `void seq64::seqedit::q_rec_change_callback () [private]`

12.60.3.23 `void seq64::seqedit::thru_change_callback () [private]`

12.60.3.24 `void seq64::seqedit::undo_callback () [private]`

12.60.3.25 `void seq64::seqedit::redo_callback () [private]`

12.60.3.26 `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!

12.60.3.27 `void seq64::seqedit::update_all_windows () [private]`

12.60.3.28 `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.

12.60.3.29 `void seq64::seqedit::create_menus () [private]`

The first menu is the Zoom menu, represented in the pattern/sequence editor by a button with a magnifying glass. The values are "pixels to ticks", where "ticks" are actually the "pulses" of "pulses per quarter note". We would prefer the notation "n" instead of "1:n", as in "n pulses per pixel".

Note that many of the setups here could be loops through data structures. The Snap menu is actually the Grid Snap button, which shows two arrows pointing to a central bar. This menu somewhat duplicates the same menu in `perfedit`.

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").

12.60.3.30 `void seq64::seqedit::popup_menu (Gtk::Menu * menu) [private]`

12.60.3.31 `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.

12.60.3.32 `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.

12.60.3.33 `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.

12.60.3.34 `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.

12.60.3.35 `void seq64::seqedit::popup_midich_menu () [private]`

12.60.3.36 `Gtk::Image * seq64::seqedit::create_menu_image (bool state = false) [private]`

12.60.3.37 `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.

12.60.3.38 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).

12.60.3.39 void seq64::seqedit::mouse_action (mouse_action_e *action*) [private]

12.60.3.40 void seq64::seqedit::change_focus (bool *set_it* = true) [private]

Similar to the same function in eventedit.

Parameters

<i>set_it</i>	If true (the default value), indicates we want focus, otherwise we want to give up focus.
---------------	---

12.60.3.41 void seq64::seqedit::handle_close () [private]

12.60.3.42 void seq64::seqedit::on_realize () [private]

12.60.3.43 void seq64::seqedit::on_set_focus (Widget * *focus*) [private]

Only works in certain circumstances.

12.60.3.44 bool seq64::seqedit::on_focus_in_event (GdkEventFocus *) [private]

12.60.3.45 bool seq64::seqedit::on_focus_out_event (GdkEventFocus *) [private]

12.60.3.46 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.

12.60.3.47 `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.

12.60.3.48 `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.

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)`.

12.60.4 Field Documentation

12.60.4.1 `friend seq64::seqedit::seqmenu [private]`

12.60.4.2 `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.

12.60.4.3 `int seq64::seqedit::m_initial_note_length` [static], [private]

12.60.4.4 `const int seq64::seqedit::m_initial_zoom` [private]

12.60.4.5 `int seq64::seqedit::m_zoom` [private]

The value of zoom is the same as the number of pixels per tick on the piano roll.

12.60.4.6 `int seq64::seqedit::m_snap` [private]

12.60.4.7 `int seq64::seqedit::m_note_length` [private]

12.60.4.8 `int seq64::seqedit::m_scale` [private]

12.60.4.9 `int seq64::seqedit::m_key` [private]

12.60.4.10 `int seq64::seqedit::m_bgsequence` [private]

12.60.4.11 `long seq64::seqedit::m_measures` [private]

12.60.4.12 `int seq64::seqedit::m_ppqn` [private]

12.60.4.13 `sequence& seq64::seqedit::m_seq` [private]

12.60.4.14 `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.

- 12.60.4.15 `Gtk::Menu*` `seq64::seqedit::m_menu_tools` `[private]`
- 12.60.4.16 `Gtk::Menu*` `seq64::seqedit::m_menu_zoom` `[private]`
- 12.60.4.17 `Gtk::Menu*` `seq64::seqedit::m_menu_snap` `[private]`
- 12.60.4.18 `Gtk::Menu*` `seq64::seqedit::m_menu_note_length` `[private]`
- 12.60.4.19 `Gtk::Menu*` `seq64::seqedit::m_menu_length` `[private]`
- 12.60.4.20 `Gtk::Menu*` `seq64::seqedit::m_menu_midich` `[private]`
- 12.60.4.21 `Gtk::Menu*` `seq64::seqedit::m_menu_midibus` `[private]`
- 12.60.4.22 `Gtk::Menu*` `seq64::seqedit::m_menu_data` `[private]`
- 12.60.4.23 `Gtk::Menu*` `seq64::seqedit::m_menu_key` `[private]`
- 12.60.4.24 `Gtk::Menu*` `seq64::seqedit::m_menu_scale` `[private]`
- 12.60.4.25 `Gtk::Menu*` `seq64::seqedit::m_menu_sequences` `[private]`
- 12.60.4.26 `Gtk::Menu*` `seq64::seqedit::m_menu_bpm` `[private]`
- 12.60.4.27 `Gtk::Menu*` `seq64::seqedit::m_menu_bw` `[private]`
- 12.60.4.28 `Gtk::Menu*` `seq64::seqedit::m_menu_rec_vol` `[private]`
- 12.60.4.29 `Gtk::Adjustment*` `seq64::seqedit::m_vadjust` `[private]`

Vertical position descriptor.

- 12.60.4.30 `Gtk::Adjustment*` `seq64::seqedit::m_hadjust` `[private]`
- 12.60.4.31 `Gtk::VScrollbar*` `seq64::seqedit::m_vscroll_new` `[private]`
- 12.60.4.32 `Gtk::HScrollbar*` `seq64::seqedit::m_hscroll_new` `[private]`
- 12.60.4.33 `seqkeys*` `seq64::seqedit::m_seqkeys_wid` `[private]`

This item draws the piano-keys at the left of the seqedit window.

- 12.60.4.34 `seqtime*` `seq64::seqedit::m_seqtime_wid` `[private]`

This is the location where the measure numbers and the END marker are shown.

12.60.4.35 **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.

12.60.4.36 **sequevent*** seq64::seqedit::m_sequevent_wid [private]

12.60.4.37 **seqroll*** seq64::seqedit::m_seqroll_wid [private]

12.60.4.38 **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.

12.60.4.39 **Gtk::VBox*** seq64::seqedit::m_vbox [private]

12.60.4.40 **Gtk::HBox*** seq64::seqedit::m_hbox [private]

12.60.4.41 **Gtk::HBox*** seq64::seqedit::m_hbox2 [private]

12.60.4.42 **Gtk::Button*** seq64::seqedit::m_button_undo [private]

12.60.4.43 **Gtk::Button*** seq64::seqedit::m_button_redo [private]

12.60.4.44 **Gtk::Button*** seq64::seqedit::m_button_quantize [private]

12.60.4.45 **Gtk::Button*** seq64::seqedit::m_button_tools [private]

12.60.4.46 **Gtk::Button*** seq64::seqedit::m_button_sequence [private]

12.60.4.47 **Gtk::Entry*** seq64::seqedit::m_entry_sequence [private]

12.60.4.48 **Gtk::Button*** seq64::seqedit::m_button_bus [private]

12.60.4.49 **Gtk::Entry*** seq64::seqedit::m_entry_bus [private]

12.60.4.50 **Gtk::Button*** seq64::seqedit::m_button_channel [private]

12.60.4.51 **Gtk::Entry*** seq64::seqedit::m_entry_channel [private]

12.60.4.52 **Gtk::Button*** seq64::seqedit::m_button_snap [private]

12.60.4.53 **Gtk::Entry*** seq64::seqedit::m_entry_snap [private]

12.60.4.54 **Gtk::Button*** seq64::seqedit::m_button_note_length [private]

- 12.60.4.55 `Gtk::Entry*` `seq64::seqedit::m_entry_note_length` `[private]`
- 12.60.4.56 `Gtk::Button*` `seq64::seqedit::m_button_zoom` `[private]`
- 12.60.4.57 `Gtk::Entry*` `seq64::seqedit::m_entry_zoom` `[private]`
- 12.60.4.58 `Gtk::Button*` `seq64::seqedit::m_button_length` `[private]`
- 12.60.4.59 `Gtk::Entry*` `seq64::seqedit::m_entry_length` `[private]`
- 12.60.4.60 `Gtk::Button*` `seq64::seqedit::m_button_key` `[private]`
- 12.60.4.61 `Gtk::Entry*` `seq64::seqedit::m_entry_key` `[private]`
- 12.60.4.62 `Gtk::Button*` `seq64::seqedit::m_button_scale` `[private]`
- 12.60.4.63 `Gtk::Entry*` `seq64::seqedit::m_entry_scale` `[private]`
- 12.60.4.64 `Gtk::Tooltips*` `seq64::seqedit::m_tooltips` `[private]`
- 12.60.4.65 `Gtk::Button*` `seq64::seqedit::m_button_data` `[private]`
- 12.60.4.66 `Gtk::Entry*` `seq64::seqedit::m_entry_data` `[private]`
- 12.60.4.67 `Gtk::Button*` `seq64::seqedit::m_button_bpm` `[private]`
- 12.60.4.68 `Gtk::Entry*` `seq64::seqedit::m_entry_bpm` `[private]`
- 12.60.4.69 `Gtk::Button*` `seq64::seqedit::m_button_bw` `[private]`
- 12.60.4.70 `Gtk::Entry*` `seq64::seqedit::m_entry_bw` `[private]`
- 12.60.4.71 `Gtk::Button*` `seq64::seqedit::m_button_rec_vol` `[private]`
- 12.60.4.72 `Gtk::ToggleButton*` `seq64::seqedit::m_toggle_play` `[private]`
- 12.60.4.73 `Gtk::ToggleButton*` `seq64::seqedit::m_toggle_record` `[private]`
- 12.60.4.74 `Gtk::ToggleButton*` `seq64::seqedit::m_toggle_q_rec` `[private]`
- 12.60.4.75 `Gtk::ToggleButton*` `seq64::seqedit::m_toggle_thru` `[private]`
- 12.60.4.76 `Gtk::Entry*` `seq64::seqedit::m_entry_name` `[private]`
- 12.60.4.77 `midibyte` `seq64::seqedit::m_editing_status` `[private]`

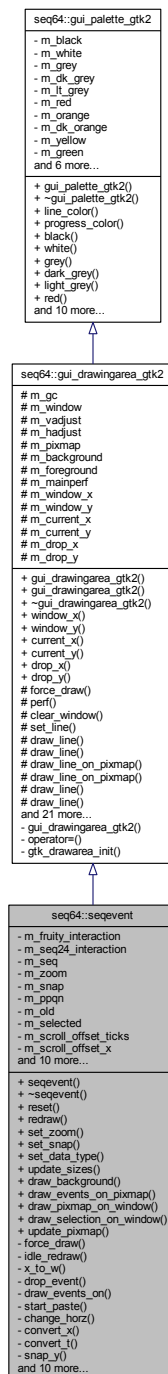
12.60.4.78 midibyte seq64::seqedit::m_editing_cc [private]

12.60.4.79 bool seq64::seqedit::m_have_focus [private]

12.61 seq64::sequevent Class Reference

Implements the piano event drawing area.

Inheritance diagram for seq64::sequevent:



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 [~sequest](#) ()
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 coordinate, 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

12.61.1 Constructor & Destructor Documentation

- 12.61.1.1 `seq64::sequevent::sequevent (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.

- 12.61.1.2 `virtual seq64::sequevent::~~sequevent ()` `[inline]`, `[virtual]`

12.61.2 Member Function Documentation

- 12.61.2.1 `void seq64::sequevent::reset ()`

Basically identical to [seqtime::reset\(\)](#).

12.61.2.2 `void seq64::segevent::redraw ()`

Somewhat similar to [seqroll::redraw\(\)](#).

12.61.2.3 `void seq64::segevent::set_zoom (int z)`

Parameters

<i>z</i>	The desired zoom value, presumably already validated by the caller.
----------	---

12.61.2.4 `void seq64::segevent::set_snap (int snap)` `[inline]`

The parameter is not validated.

12.61.2.5 `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.

12.61.2.6 `void seq64::segevent::update_sizes ()`

This ends up filling the background with dotted lines, etc.

12.61.2.7 `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.

12.61.2.8 `void seq64::segevent::draw_events_on_pixmap ()`

12.61.2.9 `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.
```

12.61.2.10 void seq64::sequest::draw_selection_on_window ()

12.61.2.11 void seq64::sequest::update_pixmap ()

12.61.2.12 void seq64::sequest::force_draw () [private],[virtual]

Reimplemented from [seq64::gui_drawingarea_gtk2](#).

12.61.2.13 int seq64::sequest::idle_redraw () [private]

Who calls this routine? Probably the default timer routine, but not sure.

Returns

Always returns true.

12.61.2.14 void seq64::sequest::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.

12.61.2.15 void seq64::sequest::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.
-------------	--

12.61.2.16 void seq64::sequest::draw_events_on (Glib::RefPtr< Gdk::Drawable > *drawable*) [private]

Very similar to [seqdata::draw_events_on\(\)](#).

Parameters

<i>drawable</i>	The given drawable object.
-----------------	----------------------------

12.61.2.17 `void seq64::seqevent::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.

12.61.2.18 `void seq64::seqevent::change_horz() [private]`

Very similar to [seqroll::change_horz\(\)](#). Basically identical to [seqdata::change_horz\(\)](#).

12.61.2.19 `void seq64::seqevent::convert_x(int x, midipulse & tick) [inline],[private]`

Why not just return it normally?

Parameters

	<i>x</i>	The x (pixel) value to convert.
out	<i>tick</i>	The destination for the converted x value.

12.61.2.20 `void seq64::seqevent::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.

12.61.2.21 `void seq64::seqevent::snap_y(int & y) [inline],[private]`

Parameters

out	<i>y</i>	The return parameter for the conversion. Why not just return the value?
-----	----------	---

12.61.2.22 `void seq64::seqevent::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.
-----	----------	---

12.61.2.23 void seq64::sequevent::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.

12.61.2.24 bool seq64::sequevent::on_expose_event (GdkEventExpose * *ev*) [private]

Parameters

<i>ev</i>	The expose event.
-----------	-------------------

12.61.2.25 bool seq64::sequevent::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.

12.61.2.26 bool seq64::sequevent::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.

12.61.2.27 `bool seq64::seqevent::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.

12.61.2.28 `bool seq64::seqevent::on_focus_in_event (GdkEventFocus *) [private]`

Parameter "ev" is the focus event, unused.

Returns

Always returns false.

12.61.2.29 `bool seq64::seqevent::on_focus_out_event (GdkEventFocus *) [private]`

Parameter "ev" is the focus event, unused.

Returns

Always returns false.

12.61.2.30 `bool seq64::seqevent::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.

12.61.2.31 void seq64::sequest::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

a	The allocation to be processed.
---	---------------------------------

12.61.3 Friends And Related Function Documentation

12.61.3.1 friend struct FruitySeqEventInput [friend]

12.61.3.2 friend struct Seq24SeqEventInput [friend]

12.61.4 Field Documentation

12.61.4.1 FruitySeqEventInput seq64::sequest::m_fruity_interaction [private]

Why should we need both at the same time? Just load the one that is specified in the configuration.

12.61.4.2 Seq24SeqEventInput seq64::sequest::m_seq24_interaction [private]

12.61.4.3 sequence& seq64::sequest::m_seq [private]

12.61.4.4 int seq64::sequest::m_zoom [private]

12.61.4.5 int seq64::sequest::m_snap [private]

Same meaning as for the piano roll. This value is the denominator of the note size used for the snap.

12.61.4.6 int seq64::sequest::m_ppqn [private]

Used in snap and zoom scaling.

12.61.4.7 GdkRectangle seq64::sequest::m_old [private]

12.61.4.8 GdkRectangle seq64::sequest::m_selected [private]

12.61.4.9 int seq64::sequest::m_scroll_offset_ticks [private]

12.61.4.10 int seq64::sequest::m_scroll_offset_x [private]

Set to m_scroll_offset_ticks divided by m_zoom.

12.61.4.11 `seqdata& seq64::seqevent::m_seqdata_wid` [private]

12.61.4.12 `bool seq64::seqevent::m_selecting` [private]

12.61.4.13 `bool seq64::seqevent::m_moving_init` [private]

12.61.4.14 `bool seq64::seqevent::m_moving` [private]

WARNING: This operation seems to have a bug. It makes the events very very long. This bug exists in Seq24.

12.61.4.15 `bool seq64::seqevent::m_growing` [private]

Does growing work in this view? Need to do some better testing.

12.61.4.16 `bool seq64::seqevent::m_painting` [private]

12.61.4.17 `bool seq64::seqevent::m_paste` [private]

12.61.4.18 `int seq64::seqevent::m_move_snap_offset_x` [private]

12.61.4.19 `midibyte seq64::seqevent::m_status` [private]

The current status/event byte.

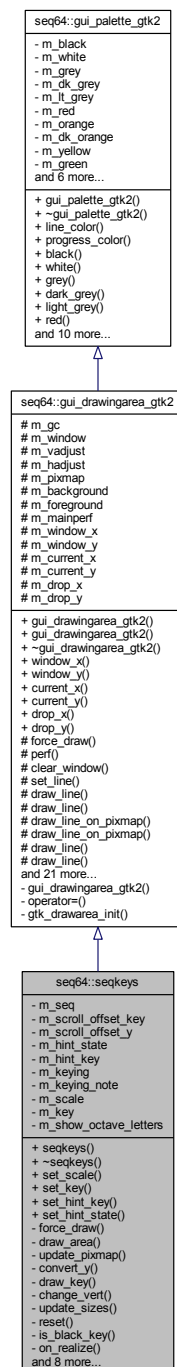
12.61.4.20 `midibyte seq64::seqevent::m_cc` [private]

The current MIDI CC value.

12.62 `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 [draw_area](#) ()

Draws the updated pixmap on the drawable area of the window where the keys' location is hardwired.

- void [update_pixmap](#) ()

Updates the pixmaps to prepare it for the next draw operation.

- void [convert_y](#) (int y, int ¬e)

Takes the screen y coordinate, and returns the note value in the second parameter.

- void [draw_key](#) (int key, bool state)

Draws the given key according to the given state.

- void [change_vert](#) ()

Changes the y offset of the scrolling, and the forces a draw.

- void [update_sizes](#) ()

- void [reset](#) ()

Resetting the keys view updates the pixmap and queues up a draw operation.

- bool [is_black_key](#) (int key) const

Detects a black key.

- void [on_realize](#) ()

Implements the on-realize event.

- bool [on_expose_event](#) (GdkEventExpose *ev)

Implements the on-expose event, by drawing on the window.

- bool [on_button_press_event](#) (GdkEventButton *ev)

Implements the on-button-press event callback.

- bool [on_button_release_event](#) (GdkEventButton *ev)

Implements the on-button-release event callback.

- bool [on_motion_notify_event](#) (GdkEventMotion *p0)

Implements the on-motion-notify event handler.

- bool [on_enter_notify_event](#) (GdkEventCrossing *p0)

Implements the on-enter notification event handler.

- bool [on_leave_notify_event](#) (GdkEventCrossing *p0)

Implements the on-leave notification event handler.

- bool [on_scroll_event](#) (GdkEventScroll *ev)

Implements the on-scroll-event notification event handler.

- void [on_size_allocate](#) (Gtk::Allocation &)

Implements the on-size-allocation notification event handler.

Private Attributes

- [sequence](#) & [m_seq](#)
The sequence object that the keys pane will be using.
- int [m_scroll_offset_key](#)
Provides the value of the current top key in the keys pane.
- int [m_scroll_offset_y](#)
Provides the value of the current top key in the keys pane in units of relative pixels.
- bool [m_hint_state](#)
Indicates if a piano key is set to indicate where on the pitch scale the mouse cursor is sitting.
- int [m_hint_key](#)
Indicates the current y-value of the mouse pointer in units of key value.
- bool [m_keying](#)
Set to true while the left mouse button is being pressed.
- int [m_keying_note](#)
The note to be played when selected in the seqkeys pane.
- int [m_scale](#)
This member holds the scale value for the musical scale for the current edit of the sequence.
- int [m_key](#)
This member holds the key value for the musical key for the current edit of the sequence.
- bool [m_show_octave_letters](#)
The default value is to show the octave letters on the vertical virtual keyboard.

Additional Inherited Members

12.62.1 Constructor & Destructor Documentation

12.62.1.1 `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.

12.62.1.2 `virtual seq64::seqkeys::~seqkeys () [inline],[virtual]`

12.62.2 Member Function Documentation

12.62.2.1 `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.
--------------	------------------------------------

12.62.2.2 `void seq64::seqkeys::set_key (int key)`

Parameters

<i>key</i>	The musical key value to be set.
------------	----------------------------------

12.62.2.3 `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.
------------	---------------------------------------

12.62.2.4 `void seq64::seqkeys::set_hint_state (bool state)`

Parameters

<i>state</i>	Provides the value for hinting, where true == on, false == off.
--------------	---

12.62.2.5 `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](#).

12.62.2.6 `void seq64::seqkeys::draw_area ()` [*private*]

12.62.2.7 `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.

12.62.2.8 `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.

12.62.2.9 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 grayed when the mouse cursor is at the same vertical location on the piano as the key.

12.62.2.10 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();
```

12.62.2.11 void seq64::seqkeys::update_sizes () [private]

12.62.2.12 void seq64::seqkeys::reset () [private]

12.62.2.13 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).

12.62.2.14 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.

12.62.2.15 bool seq64::seqkeys::on_expose_event (GdkEventExpose * *ev*) [private]

Parameters

<i>ev</i>	The expose-event object.
-----------	--------------------------

12.62.2.16 `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.

12.62.2.17 `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.

12.62.2.18 `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.

12.62.2.19 `bool seq64::seqkeys::on_enter_notify_event (GdkEventCrossing * p0)` `[private]`

This greys the current key.

12.62.2.20 `bool seq64::seqkeys::on_leave_notify_event (GdkEventCrossing * p0)` `[private]`

This un-greys the current key and stops playing the note.

12.62.2.21 `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

<code><i>ev</i></code>	Provides the direction of the scroll event.
------------------------	---

Returns

Always returns true.

12.62.2.22 `void seq64::seqkeys::on_size_allocate (Gtk::Allocation & all)` `[private]`

Parameters

<code><i>all</i></code>	Provies the allocation and its width and height.
-------------------------	--

12.62.3 Field Documentation

12.62.3.1 `sequence& seq64::seqkeys::m_seq` `[private]`

12.62.3.2 `int seq64::seqkeys::m_scroll_offset_key` `[private]`

Modified in [change_vert\(\)](#).

12.62.3.3 `int seq64::seqkeys::m_scroll_offset_y` `[private]`

Modified in [change_vert\(\)](#).

12.62.3.4 `bool seq64::seqkeys::m_hint_state` `[private]`

12.62.3.5 `int seq64::seqkeys::m_hint_key` `[private]`

12.62.3.6 `bool seq64::seqkeys::m_keying` `[private]`

Used in playing the sound for each note as it is clicked in the seqkeys pane.

12.62.3.7 `int seq64::seqkeys::m_keying_note` `[private]`

12.62.3.8 `int seq64::seqkeys::m_scale` `[private]`

12.62.3.9 `int seq64::seqkeys::m_key` `[private]`

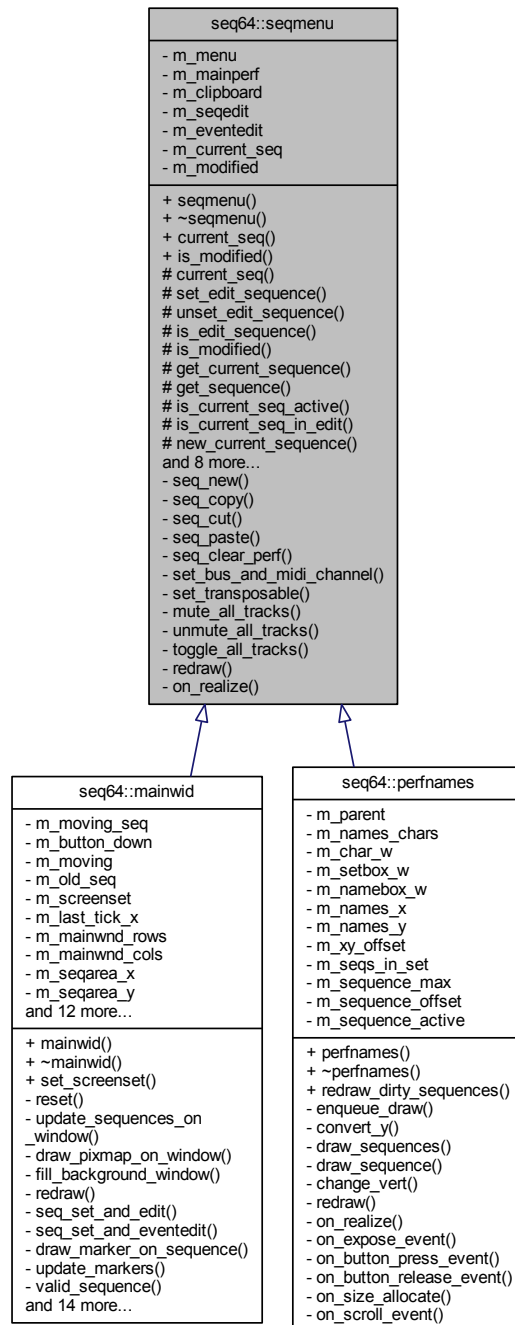
12.62.3.10 `bool seq64::seqkeys::m_show_octave_letters` `[private]`

If false, then the MIDI key numbers are shown instead. This is a new feature of Sequencer64.

12.63 `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.

- 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".

Private Member Functions

- 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.
- virtual void [redraw](#) (int a_sequence)=0
- 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.

12.63.1 Detailed Description

It is an abstract base class.

12.63.2 Constructor & Destructor Documentation

12.63.2.1 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.
----------	---

12.63.2.2 `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.

12.63.3 Member Function Documentation

12.63.3.1 `int seq64::seqmenu::current_seq () const` `[inline]`

12.63.3.2 `bool seq64::seqmenu::is_modified () const` `[inline]`

12.63.3.3 `void seq64::seqmenu::current_seq (int seq)` `[inline]`, `[protected]`

12.63.3.4 `void seq64::seqmenu::set_edit_sequence (int seqnum)` `[inline]`, `[protected]`

Now a pass-along to the perform object.

12.63.3.5 `void seq64::seqmenu::unset_edit_sequence (int seqnum)` `[inline]`, `[protected]`

12.63.3.6 `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.

12.63.3.7 `void seq64::seqmenu::is_modified (bool flag)` `[inline]`, `[protected]`

12.63.3.8 `sequence* seq64::seqmenu::get_current_sequence () const` `[inline]`, `[protected]`

12.63.3.9 `sequence* seq64::seqmenu::get_sequence (int seqnum) const` `[inline]`, `[protected]`

12.63.3.10 `bool seq64::seqmenu::is_current_seq_active () const` `[inline]`, `[protected]`

12.63.3.11 `bool seq64::seqmenu::is_current_seq_in_edit () const` `[inline]`, `[protected]`

12.63.3.12 `void seq64::seqmenu::new_current_sequence ()` `[inline]`, `[protected]`

12.63.3.13 `void seq64::seqmenu::new_sequence (int seqnum)` `[inline]`, `[protected]`

12.63.3.14 `void seq64::seqmenu::delete_current_sequence ()` `[inline]`, `[protected]`

12.63.3.15 `void seq64::seqmenu::toggle_current_sequence ()` `[inline]`, `[protected]`

12.63.3.16 `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.

12.63.3.17 `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.

12.63.3.18 `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.

12.63.3.19 `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.

Parameters

<i>seqnum</i>	The number of the sequence to edit.
---------------	-------------------------------------

Reimplemented in [seq64::mainwid](#).

12.63.3.20 `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](#).

12.63.3.21 `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.

12.63.3.22 `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.

12.63.3.23 `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.

12.63.3.24 `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.

12.63.3.25 `void seq64::seqmenu::seq_clear_perf () [private]`

Todo All of [seq_paste\(\)](#) can be offloaded to a (new) perform member function.

12.63.3.26 `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.

12.63.3.27 `void seq64::seqmenu::set_transposable (bool flag) [private]`

Parameters

<i>flag</i>	The value to use to set the flag.
-------------	-----------------------------------

12.63.3.28 `void seq64::seqmenu::mute_all_tracks () [private]`

12.63.3.29 `void seq64::seqmenu::unmute_all_tracks () [private]`

12.63.3.30 void seq64::seqmenu::toggle_all_tracks () [private]

12.63.3.31 virtual void seq64::seqmenu::redraw (int *a_sequence*) [private],[pure virtual]

Implemented in [seq64::mainwid](#), and [seq64::perfnames](#).

12.63.3.32 void seq64::seqmenu::on_realize () [private]

12.63.4 Field Documentation

12.63.4.1 Gtk::Menu* seq64::seqmenu::m_menu [private]

12.63.4.2 perform& seq64::seqmenu::m_mainperf [private]

12.63.4.3 sequence seq64::seqmenu::m_clipboard [private]

12.63.4.4 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.

12.63.4.5 eventedit* seq64::seqmenu::m_eventedit [private]

12.63.4.6 int seq64::seqmenu::m_current_seq [private]

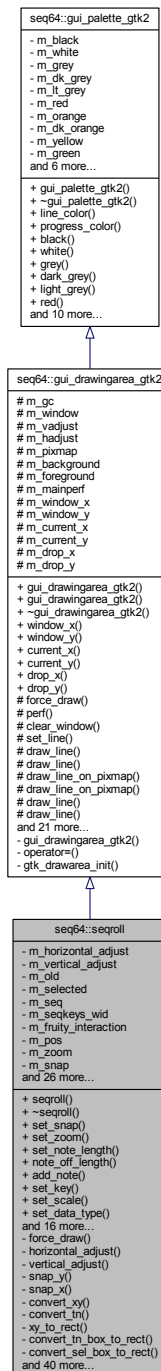
12.63.4.7 bool seq64::seqmenu::m_modified [private]

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

12.64 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 [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_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 ¬e)
- 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 ¬e_h, `midipulse` &tick_f, int ¬e_l)
A convenience function wrapping a common call to `m_seq.get_selected_box()` and `convert_tn_box_to_rect()`.
- void `draw_events_on` (Glib::RefPtr< Gdk::Drawable > draw)
Draws events on the given drawable area.
- int `idle_redraw` ()
Draw the events on the main window and on the pixmap.
- int `idle_progress` ()
- void `change_horz` ()
Change the horizontal scrolling offset and redraw.
- void `change_vert` ()
Change the vertical scrolling offset and redraw.
- void `move_selection_box` (int dx, int dy)
Function to allow motion of the selection box via the arrow keys.
- void `move_selected_notes` (int dx, int dy)
Proposed new function to encapsulate the movement of selections even more fully.
- void `grow_selected_notes` (int dx)
Proposed new function to encapsulate the movement of selections even more fully.
- void `set_adding` (bool adding)
Changes the mouse cursor pixmap according to whether a note is being added or not.
- void `update_mouse_pointer` (bool adding=false)
Updates the mouse pointer, implementing a context-sensitive mouse.
- bool `button_press_initial` (GdkEventButton *ev, int &norm_x, int &snapped_x, int &snapped_y)
- void `align_selection` (`midipulse` &tick_s, int ¬e_h, `midipulse` &tick_f, int ¬e_l, int snapped_x)
Get the box that selected elements are in.
- bool `button_press` (GdkEventButton *ev)
Implements the on-button-press event handling for the Seq24 style of mouse interaction.
- bool `button_release` (GdkEventButton *ev)
Implements the on-button-release event handling for the Seq24 style of mouse interaction.
- bool `motion_notify` (GdkEventMotion *ev)

- Seq24-style on-motion mouse interaction.*

 - void [clear_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.*
 - 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`
We need direct access to the horizontal scrollbar if we want to be able to make it follow the progress bar.
- `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_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.
- 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

12.64.1 Constructor & Destructor Documentation

- 12.64.1.1 `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 seqroll::m_pos . This is actually the sequence number, and is currently unused. However, we're sure we can find a use for it sometime.
<i>hadjust</i>	Represents the horizontal scrollbar of this window. It is actually created by the "parent" seqedit object.
<i>vadjust</i>	Represents the vertical scrollbar of this window. It is actually created by the "parent" seqedit object.
<i>ppqn</i>	The initial value of the PPQN for this sequence. Useful in scale calculations.

12.64.1.2 `seq64::seqroll::~~seqroll ()` `[virtual]`

The only thing to delete here is the clipboard. Except it is never used, so is commented out.

12.64.2 Member Function Documentation

12.64.2.1 `void seq64::seqroll::set_snap (int snap)` `[inline]`

Parameters

<i>snap</i>	Provides the sname value to set.
-------------	----------------------------------

12.64.2.2 `void seq64::seqroll::set_zoom (int zoom)`

Parameters

<i>zoom</i>	The desired zoom value.
-------------	-------------------------

12.64.2.3 `void seq64::seqroll::set_note_length (int note_length)` `[inline]`

12.64.2.4 `int seq64::seqroll::note_off_length () const` `[inline]`

12.64.2.5 `bool seq64::seqroll::add_note (midipulse tick, int note, bool paint = true)`

The length parameters is obtained from the [note_off_length\(\)](#) function. This sets the note length at a little less than the snap value.

Parameters

<i>tick</i>	The time destination of the new note, in pulses.
<i>note</i>	The pitch destination of the new note.
<i>paint</i>	If true, repaint to be left with just the inserted event. The default is true. The value of false is useful in inserting a number of events and saving the repainting until last. It is a bit tricky, as the default paint value for sequence::add_note() is false.

12.64.2.6 `void seq64::seqroll::set_key (int key)`

Parameters

<i>key</i>	The desired key value.
------------	------------------------

12.64.2.7 `void seq64::seqroll::set_scale (int scale)`

Parameters

<i>scale</i>	The desired scale value.
--------------	--------------------------

12.64.2.8 `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.

12.64.2.9 `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.

12.64.2.10 `void seq64::seqroll::update_pixmap ()`

12.64.2.11 `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.

12.64.2.12 `void seq64::seqroll::update_background ()`

The first thing done is to clear the background, painting it white.

12.64.2.13 `void seq64::seqroll::draw_background_on_pixmap ()`

12.64.2.14 `void seq64::seqroll::draw_events_on_pixmap ()`

Just calls [draw_events_on\(\)](#).

12.64.2.15 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.

12.64.2.16 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.

12.64.2.17 void seq64::seqroll::reset ()

It's almost identical to the [change_horz\(\)](#) function, just calling [update_sizes\(\)](#) before [update_and_draw\(\)](#).

12.64.2.18 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.
--------------	---

12.64.2.19 void seq64::seqroll::redraw ()

Somewhat similar to [sequevent::redraw\(\)](#). Actually, we don't seem to need to ignore redraw when making settings in the seqedit constructor, so this member no longer exists.

12.64.2.20 void seq64::seqroll::redraw_events ()

Actually, that member is not needed and no longer exists.

12.64.2.21 void seq64::seqroll::start_paste ()

12.64.2.22 void seq64::seqroll::complete_paste () [inline]

12.64.2.23 void seq64::seqroll::complete_paste (int *x*, int *y*)

12.64.2.24 void seq64::seqroll::follow_progress () [inline]

12.64.2.25 void seq64::seqroll::force_draw () [private],[virtual]

Reimplemented from [seq64::gui_drawingarea_gtk2](#).

12.64.2.26 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 gui_drawingarea_gtk2::scroll_hadjust() for more information.
-------------	---

12.64.2.27 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 gui_drawingarea_gtk2::scroll_vadjust() for more information.
-------------	---

12.64.2.28 void seq64::seqroll::snap_y (int & *y*) [inline],[private]

Parameters

out	<i>y</i>	The y-value to be snapped.
-----	----------	----------------------------

12.64.2.29 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

out	<i>x</i>	Provides the x-value to be snapped and returned. A return value would be better.
-----	----------	--

12.64.2.30 `void seq64::seqroll::convert_xy (int x, int y, midipulse & ticks, int & note)` [private]

12.64.2.31 `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.

12.64.2.32 `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.
	<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.

12.64.2.33 `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</i> ↔ <i>_h</i>	The high note of the rectangle.

Parameters

	<i>note</i> ↔ <i>_l</i>	The low note of the rectangle.
out	<i>x</i>	The destination for the x value in pixels.
out	<i>y</i>	The destination for the y value in pixels.
out	<i>w</i>	The destination for the rectangle width in pixels.
out	<i>h</i>	The destination for the rectangle height value in pixels.

12.64.2.34 `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</i> ↔ <i>_h</i>	The high note of the rectangle.
<i>note</i> ↔ <i>_l</i>	The low note of the rectangle.

12.64.2.35 `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</i> ↔ <i>_h</i>	The high note of the rectangle.
out	<i>note</i> ↔ <i>_l</i>	The low note of the rectangle.

12.64.2.36 `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.
-------------	---------------------------------

12.64.2.37 `int seq64::seqroll::idle_redraw ()` [private]

12.64.2.38 `int seq64::seqroll::idle_progress ()` [private]

12.64.2.39 `void seq64::seqroll::change_horz ()` [private]

Roughly similar to [sequevent::change_horz\(\)](#).

12.64.2.40 `void seq64::seqroll::change_vert () [private]`

12.64.2.41 `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.

12.64.2.42 `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.

12.64.2.43 `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.
-----------	---

12.64.2.44 `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.
---------------	------------------------

12.64.2.45 `void seq64::seqroll::update_mouse_pointer (bool adding = false) [private]`

Moved here from the "fruity" seqroll class.

12.64.2.46 `bool seq64::seqroll::button_press_initial (GdkEventButton * ev, int & norm_x, int & snapped_x, int & snapped_y) [private]`

12.64.2.47 `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().

12.64.2.48 `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

<i>ev</i>	Provides the button-press event to process.
-----------	---

Returns

Returns the value of needs_update. It used to return only true.

12.64.2.49 `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

<i>ev</i>	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 `perffrollinput.cpp`.

12.64.2.50 `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

<code>ev</code>	Provides the button-release event to process.
-----------------	---

Returns

Returns true if the event was processed.

12.64.2.51 `void seq64::seqroll::clear_selected () [inline],[private]`

12.64.2.52 `void seq64::seqroll::clear_old () [inline],[private]`

12.64.2.53 `void seq64::seqroll::clear_flags () [inline],[private]`

12.64.2.54 `int seq64::seqroll::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.
----------------	------------------------

12.64.2.55 `int seq64::seqroll::scroll_offset_y (int y) const [inline],[private]`

Offsets the y value by the y origin of the current page.

Parameters

<i>y</i>	The y value to offset.
----------	------------------------

12.64.2.56 `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

<i>x</i>	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

<i>y</i>	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

<i>x</i>	The y value to offset.
<i>y</i>	The y value to offset.

12.64.2.57 `bool seq64::seqroll::adding () const` `[inline], [private]`

12.64.2.58 `bool seq64::seqroll::selecting () const` `[inline], [private]`

12.64.2.59 `bool seq64::seqroll::growing () const` `[inline], [private]`

12.64.2.60 `bool seq64::seqroll::normal_action () const` `[inline], [private]`

Returns

Returns true if one of those five flags are set.

12.64.2.61 `bool seq64::seqroll::select_action () const` `[inline], [private]`

Returns

Returns true if one of those four flags are set.

12.64.2.62 `bool seq64::seqroll::drop_action () const [inline], [private]`

Returns

Returns true if one of those two flags are set.

12.64.2.63 `void seq64::seqroll::on_realize () [private]`

12.64.2.64 `bool seq64::seqroll::on_expose_event (GdkEventExpose * ev) [private]`

Parameters

<i>ev</i>	The expose event to process.
-----------	------------------------------

Returns

Always returns true.

12.64.2.65 `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().

12.64.2.66 `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

<i>ev</i>	The button release event to process.
-----------	--------------------------------------

Returns

Returns the return value of Seq24SeqRollInput::on_button_release_event() or FruitySeqRollInput::on_↔button_release_event().

12.64.2.67 `bool seq64::seqroll::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\(\)](#).

12.64.2.68 `bool seq64::seqroll::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.

12.64.2.69 `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.

12.64.2.70 `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

<code>ev</code>	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)

12.64.2.71 `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

<code>ev</code>	The scroll event to process.
-----------------	------------------------------

Returns

Returns true if the scroll event was handled.

12.64.2.72 `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

<code>a</code>	The GDK allocation event object.
----------------	----------------------------------

12.64.2.73 `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.

12.64.2.74 `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.

12.64.3 Friends And Related Function Documentation

12.64.3.1 `friend class FruitySeqRollInput` `[friend]`

We've absorbed the Seq24SeqRollInput class functionality back into seqroll, to save code.

12.64.4 Field Documentation

12.64.4.1 `Gtk::Adjustment& seq64::seqroll::m_horizontal_adjust` [private]

12.64.4.2 `Gtk::Adjustment& seq64::seqroll::m_vertical_adjust` [private]

12.64.4.3 `rect seq64::seqroll::m_old` [private]

12.64.4.4 `rect seq64::seqroll::m_selected` [private]

12.64.4.5 `sequence& seq64::seqroll::m_seq` [private]

12.64.4.6 `seqkeys& seq64::seqroll::m_seqkeys_wid` [private]

12.64.4.7 `FruitySeqRollInput seq64::seqroll::m_fruity_interaction` [private]

12.64.4.8 `int seq64::seqroll::m_pos` [private]

Need to clarify what exactly this member is used for.

12.64.4.9 `int seq64::seqroll::m_zoom` [private]

12.64.4.10 `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.

12.64.4.11 `int seq64::seqroll::m_ppqn` [private]

Supports values other than the default of 192.

12.64.4.12 `int seq64::seqroll::m_note_length` [private]

Used in the seq24seqroll module only.

12.64.4.13 `int seq64::seqroll::m_scale` [private]

12.64.4.14 `int seq64::seqroll::m_key` [private]

12.64.4.15 `bool seq64::seqroll::m_adding` [private]

This flag was moved from both the fruity and the seq24 seqroll classes.

12.64.4.16 `bool seq64::seqroll::m_selecting` [private]

12.64.4.17 `bool seq64::seqroll::m_moving` [private]

12.64.4.18 `bool seq64::seqroll::m_moving_init` [private]

Used in the fruity and seq24 mouse-handling modules.

12.64.4.19 `bool seq64::seqroll::m_growing` [private]

12.64.4.20 `bool seq64::seqroll::m_painting` [private]

Used in the fruity and seq24 mouse-handling modules.

12.64.4.21 `bool seq64::seqroll::m_paste` [private]

12.64.4.22 `bool seq64::seqroll::m_is_drag_pasting` [private]

Used in the fruity mouse-handling module.

12.64.4.23 `bool seq64::seqroll::m_is_drag_pasting_start` [private]

Used in the fruity mouse-handling module.

12.64.4.24 `bool seq64::seqroll::m_justselected_one` [private]

Used in the fruity mouse-handling module.

12.64.4.25 `int seq64::seqroll::m_move_delta_x` [private]

12.64.4.26 `int seq64::seqroll::m_move_delta_y` [private]

12.64.4.27 `int seq64::seqroll::m_move_snap_offset_x` [private]

12.64.4.28 `int seq64::seqroll::m_progress_x` [private]

12.64.4.29 `int seq64::seqroll::m_scroll_offset_ticks` [private]

12.64.4.30 `int seq64::seqroll::m_scroll_offset_key` [private]

12.64.4.31 `int seq64::seqroll::m_scroll_offset_x` [private]

12.64.4.32 `int seq64::seqroll::m_scroll_offset_y` [private]

12.64.4.33 `int seq64::seqroll::m_background_sequence` [private]

12.64.4.34 `bool seq64::seqroll::m_drawing_background_seq` [private]

12.64.4.35 `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.

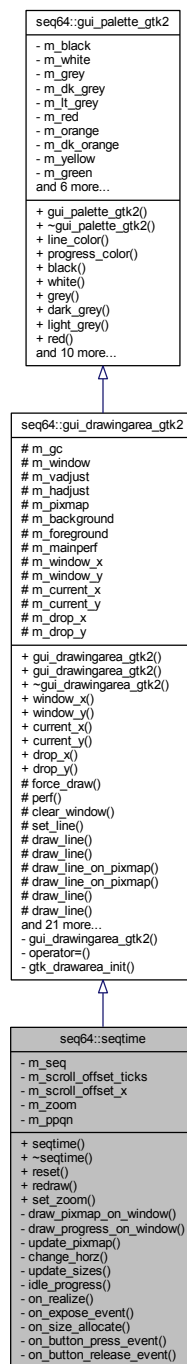
12.64.4.36 midibyte seq64::seqroll::m_cc [private]

Not used in seqroll at present.

12.65 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

12.65.1 Constructor & Destructor Documentation

12.65.1.1 `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>

12.65.1.2 `virtual seq64::seqtime::~~seqtime () [inline],[virtual]`

12.65.2 Member Function Documentation

12.65.2.1 `void seq64::seqtime::reset ()`

Basically identical to [seqevent::reset\(\)](#).

12.65.2.2 `void seq64::seqtime::redraw ()`

12.65.2.3 `void seq64::seqtime::set_zoom (int zoom)`

12.65.2.4 `void seq64::seqtime::draw_pixmap_on_window () [private]`

12.65.2.5 `void seq64::seqtime::draw_progress_on_window () [private]`

12.65.2.6 `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.

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.

12.65.2.7 void seq64::seqtime::change_horz () [private]

12.65.2.8 void seq64::seqtime::update_sizes () [private]

12.65.2.9 bool seq64::seqtime::idle_progress () [inline],[private]

12.65.2.10 void seq64::seqtime::on_realize () [private]

Call the base-class version of this function first. Then addition resources are allocated.

12.65.2.11 bool seq64::seqtime::on_expose_event (GdkEventExpose * *a_ev*) [private]

12.65.2.12 void seq64::seqtime::on_size_allocate (Gtk::Allocation & *a*) [private]

12.65.2.13 bool seq64::seqtime::on_button_press_event (GdkEventButton *) [inline],[private]

Simply returns false.

12.65.2.14 bool seq64::seqtime::on_button_release_event (GdkEventButton *) [inline],[private]

Simply returns false.

12.65.3 Field Documentation

12.65.3.1 sequence& seq64::seqtime::m_seq [private]

12.65.3.2 int seq64::seqtime::m_scroll_offset_ticks [private]

12.65.3.3 int seq64::seqtime::m_scroll_offset_x [private]

12.65.3.4 int seq64::seqtime::m_zoom [private]

12.65.3.5 int seq64::seqtime::m_ppqn [private]

12.66 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

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
- const [event_list](#) & [events](#) () const
'Getter' function for member m_events
- bool [any_selected_notes](#) () const
'Getter' function for member m_events.any_selected_notes()
- [triggers::List](#) & [triggerlist](#) ()
'Getter' function for member m_triggers
- int [number](#) () const
'Getter' function for member m_seq_number
- void [number](#) (int seqnum)
'Setter' function for member m_seq_number This setter will set the sequence number only if it has not already been set.
- int [event_count](#) () const
Returns the number of events stored in m_events.
- void [push_undo](#) ()
Pushes the event-list into the undo-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 [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
- 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](#) (int upqn)
'Setter' function for member m_us_per_quarter_note
- int [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
- void [set_song_mute](#) (bool mute)
'Setter' function for member m_song_mute This function also calls [set_dirty_mp\(\)](#) to make sure that the perfnames panel is updated to show the new mute status of the sequence.
- void [toggle_song_mute](#) ()
'Setter' function for member m_song_mute This function toggles the song muting status.
- bool [get_song_mute](#) () const
'Getter' function for member m_song_mute
- 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, bool adjust_triggers=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)
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.
- bool `add_event` (const `event` &er)
Adds an event to the internal event list in a sorted manner.
 - 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.
 - 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 ¬e)
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` ()
Copies the first selected trigger that is found.
 - void `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`, int which=2)
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)
Sets the midibus number to dump 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?*
- int `select_note_events` (`midipulse` tick_s, int note_h, `midipulse` tick_f, int note_l, `select_action_e` action)
 - This function selects events in range of 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.*
- void `select_all_notes` (bool inverse=false)
- int `get_num_selected_notes` () const
 - Counts the selected notes in the event list.*
- int `get_num_selected_events` (`midibyte` status, `midibyte` cc) const
 - Counts the selected events, with the given status, in the event list.*
- void `select_all` ()
 - Selects all events, unconditionally.*
- void `copy_selected` ()
 - Copies the selected events.*
- void `cut_selected` (bool copyevents=true)
 - Cuts the selected events.*
- void `paste_selected` (`midipulse` tick, int note)
 - Pastes the selected notes (and only note events) at the given tick and the given note value.*
- void `get_selected_box` (`midipulse` &tick_s, int ¬e_h, `midipulse` &tick_f, int ¬e_l)
 - Returns the 'box' of the selected items.*
- void `get_clipboard_box` (`midipulse` &tick_s, int ¬e_h, `midipulse` &tick_f, int ¬e_l)
 - Returns the 'box' of the clipboard items.*
- `midipulse` `adjust_timestamp` (`midipulse` t, bool isnoteoff=false)
 - 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.*
- void `add_note` (`midipulse` tick, `midipulse` len, int note, bool paint=false)
 - Adds a note 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.*
- void `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 `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.*

 - void `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 `pause` ()
- A pause version of `reset()`.*

 - void `reset` (bool live_mode)
- Provides a helper function simplify and speed up `perform::reset_sequences()`.*

 - 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` `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)
- 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 `fill_container` (midi_container &c, int tracknumber)
- This function fills the given MIDI container with MIDI data from the current sequence, preparatory to writing it to a file.*

 - 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.*

 - 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)
- 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 `split_trigger` (trigger &trig, midipulse splittick)
Splits the trigger given by the parameter into two triggers.
- 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.

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.
- [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.
- [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.
- [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 recoring in quantized mode.
- [bool m_thru](#)
True if recoring 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.
- `int 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.
- `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`

12.66.1 Detailed Description

More members than you can shake a stick at.

12.66.2 Member Typedef Documentation

12.66.2.1 `typedef std::stack<event_list> seq64::sequence::EventStack` `[private]`

12.66.3 Member Enumeration Documentation

12.66.3.1 `enum seq64::sequence::select_action_e`

See the [select_note_events\(\)](#) and [select_events\(\)](#) functions.

Enumerator

- `e_select`** To select an event.
- `e_select_one`** To select a single event.
- `e_is_selected`** The events are selected.
- `e_would_select`** The events would be selected.
- `e_deselect`** To deselect the event under the cursor.
- `e_toggle_selection`** To toggle the selection of the event under the cursor.
- `e_remove_one`** To remove one note under the cursor.

12.66.4 Constructor & Destructor Documentation

12.66.4.1 `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.
-------------	---

12.66.4.2 `seq64::sequence::~~sequence ()`

12.66.5 Member Function Documentation

12.66.5.1 `sequence& seq64::sequence::operator= (const sequence & rhs)` `[private]`

12.66.5.2 `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.
------------	---

12.66.5.3 `event_list& seq64::sequence::events ()` `[inline]`

12.66.5.4 `const event_list& seq64::sequence::events () const` `[inline]`

12.66.5.5 `bool seq64::sequence::any_selected_notes () const` `[inline]`

12.66.5.6 `triggers::List& seq64::sequence::triggerlist ()` `[inline]`

12.66.5.7 `int seq64::sequence::number () const` `[inline]`

12.66.5.8 `void seq64::sequence::number (int seqnum)` `[inline]`

12.66.5.9 `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()`.

12.66.5.10 `void seq64::sequence::push_undo ()`

Threadsafe

12.66.5.11 `void seq64::sequence::pop_undo ()`

Threadsafe

12.66.5.12 `void seq64::sequence::pop_redo ()`

Threadsafe

12.66.5.13 `void seq64::sequence::push_trigger_undo ()`

Threadsafe

12.66.5.14 void seq64::sequence::pop_trigger_undo ()

12.66.5.15 void seq64::sequence::set_name (const std::string & *name*)

12.66.5.16 void seq64::sequence::set_name (char * *name*)

12.66.5.17 void seq64::sequence::set_measures (int *lengthmeasures*)

12.66.5.18 int seq64::sequence::get_measures ()

12.66.5.19 int seq64::sequence::get_ppqn () const [inline]

12.66.5.20 void seq64::sequence::set_beats_per_bar (int *beatspermeasure*)

Threadsafe

Parameters

<i>beatspermeasure</i>	The new setting of the beats-per-bar value.
------------------------	---

12.66.5.21 int seq64::sequence::get_beats_per_bar () const [inline]

12.66.5.22 void seq64::sequence::set_beat_width (int *beatwidth*)

Threadsafe

Parameters

<i>beatwidth</i>	The new setting of the beat width value.
------------------	--

12.66.5.23 int seq64::sequence::get_beat_width () const [inline]

Threadsafe

12.66.5.24 void seq64::sequence::clocks_per_metronome (int *cpm*) [inline]

12.66.5.25 int seq64::sequence::clocks_per_metronome () const [inline]

12.66.5.26 void seq64::sequence::set_32nds_per_quarter (int *tpq*) [inline]

12.66.5.27 int seq64::sequence::get_32nds_per_quarter () const [inline]

12.66.5.28 void seq64::sequence::us_per_quarter_note (int *upqn*) [inline]

12.66.5.29 `int seq64::sequence::us_per_quarter_note () const` `[inline]`

12.66.5.30 `void seq64::sequence::set_rec_vol (int recvol)`

Threadsafe

Parameters

<i>recvol</i>	The new setting of the recording volume setting.
---------------	--

12.66.5.31 `void seq64::sequence::set_song_mute (bool mute)` `[inline]`

12.66.5.32 `void seq64::sequence::toggle_song_mute ()` `[inline]`

12.66.5.33 `bool seq64::sequence::get_song_mute () const` `[inline]`

12.66.5.34 `const char* seq64::sequence::get_name () const` `[inline]`

Deprecated

12.66.5.35 `const std::string& seq64::sequence::name () const` `[inline]`

12.66.5.36 `void seq64::sequence::set_editing (bool edit)` `[inline]`

12.66.5.37 `bool seq64::sequence::get_editing () const` `[inline]`

12.66.5.38 `void seq64::sequence::set_raise (bool edit)` `[inline]`

12.66.5.39 `bool seq64::sequence::get_raise (void) const` `[inline]`

12.66.5.40 `void seq64::sequence::set_length (midipulse len, bool adjust_triggers = true)`

This function is called in [seqedit::apply_length\(\)](#), when the user selects a sequence length in measures. 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

12.66.5.41 `midipulse seq64::sequence::get_length () const [inline]`

12.66.5.42 `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 `perfed` do?

12.66.5.43 `void seq64::sequence::set_last_tick (midipulse tick)`

Threadsafe

12.66.5.44 `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.

12.66.5.45 `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.
----------	--

12.66.5.46 `bool seq64::sequence::get_playing () const [inline]`

12.66.5.47 `void seq64::sequence::toggle_playing () [inline]`

How exactly does this differ from toggling the mute status?

12.66.5.48 `void seq64::sequence::toggle_queued ()`

Also calculates the queued tick based on `m_last_tick`.

Threadsafe

12.66.5.49 `void seq64::sequence::off_queued ()`

Do we need to set `m_queued_tick` as in [toggle_queued\(\)](#)? Currently not used.

Threadsafe

12.66.5.50 void seq64::sequence::on_queued ()

Do we need to set m_queued_tick as in [toggle_queued\(\)](#)? Currently not used.

Threadsafe

12.66.5.51 bool seq64::sequence::get_queued () const [inline]

12.66.5.52 midipulse seq64::sequence::get_queued_tick () const [inline]

12.66.5.53 bool seq64::sequence::check_queued_tick (midipulse tick) const [inline]

12.66.5.54 void seq64::sequence::set_recording (bool r)

Threadsafe

12.66.5.55 bool seq64::sequence::get_recording () const [inline]

12.66.5.56 void seq64::sequence::set_snap_tick (int st)

Threadsafe

12.66.5.57 void seq64::sequence::set_quantized_rec (bool qr)

Threadsafe

12.66.5.58 bool seq64::sequence::get_quantized_rec () const [inline]

12.66.5.59 void seq64::sequence::set_thru (bool r)

Threadsafe

12.66.5.60 bool seq64::sequence::get_thru () const [inline]

12.66.5.61 bool seq64::sequence::is_dirty_main ()

resets it). This flag signals that a redraw is needed from recording.

Threadsafe

Returns

Returns the dirty status.

12.66.5.62 `bool seq64::sequence::is_dirty_edit ()`

The `m_dirty_edit` flag is set by the function [set_dirty\(\)](#).

Threadsafe

Returns

Returns the dirty status.

12.66.5.63 `bool seq64::sequence::is_dirty_perf ()`

Threadsafe

Returns

Returns the dirty status.

12.66.5.64 `bool seq64::sequence::is_dirty_names ()`

Not sure that we need to lock a boolean on modern processors.

Threadsafe

Returns

Returns the dirty status.

12.66.5.65 `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

12.66.5.66 `void seq64::sequence::set_dirty ()`

Threadsafe

12.66.5.67 `midibyte seq64::sequence::get_midi_channel () const` `[inline]`

12.66.5.68 `bool seq64::sequence::is_smf_0 () const` `[inline]`

12.66.5.69 `void seq64::sequence::set_midi_channel (midibyte ch)`

Threadsafe

12.66.5.70 void seq64::sequence::print () const

Not threadsafe

12.66.5.71 void seq64::sequence::print_triggers () const

Not threadsafe

12.66.5.72 void seq64::sequence::play (midipulse 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.

Parameters

<i>tick</i>	Provides the current end-tick value.
<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 windows, in live mode.

Threadsafe

12.66.5.73 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.

12.66.5.74 `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.

12.66.5.75 `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.
------------------	---------------------------------

12.66.5.76 `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 <i>tickto</i> for the check.

Threadsafe

12.66.5.77 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.
-------------	--

12.66.5.78 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.

12.66.5.79 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().

12.66.5.80 bool seq64::sequence::unselect_triggers ()

Returns

Returns the m_triggers.unselect() return value.

12.66.5.81 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.

12.66.5.82 `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.

12.66.5.83 `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.

12.66.5.84 void seq64::sequence::del_selected_trigger ()

12.66.5.85 void seq64::sequence::cut_selected_trigger ()

12.66.5.86 void seq64::sequence::copy_selected_trigger ()

12.66.5.87 void seq64::sequence::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.

12.66.5.88 bool seq64::sequence::move_selected_triggers_to (midipulse tick, bool adjustoffset, int which = 2)

```
min_tick][0          1][max_tick
          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.
```

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 [Seq24PerfInput::handle_motion_key\(\)](#).

12.66.5.89 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.

12.66.5.90 **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.

12.66.5.91 **midipulse** seq64::sequence::get_max_trigger ()

Threadsafe

Returns

Returns the maximum trigger value.

12.66.5.92 **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.

12.66.5.93 **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.

12.66.5.94 **void** seq64::sequence::clear_triggers ()

Threadsafe

12.66.5.95 **midipulse** seq64::sequence::get_trigger_offset () **const** `[inline]`

12.66.5.96 **void** seq64::sequence::set_midi_bus (**char** mb)

Threadsafe

12.66.5.97 `char seq64::sequence::get_midi_bus () const` `[inline]`

12.66.5.98 `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.
------------	---

12.66.5.99 `int seq64::sequence::select_note_events (midipulse tick_s, int note_h, midipulse tick_f, int note_l, select_action_e action)`

Returns the number selected.

Threadsafe

Parameters

<i>tick_s</i>	The start time of the selection.
<i>note↔_h</i>	The high note of the selection.
<i>tick_f</i>	The finish time of the selection.
<i>note↔_l</i>	The low note of the selection.
<i>action</i>	The action to perform, one of e_select, e_select_one, e_is_selected, e_would_select, e_deselect, e_toggle_selection, and e_remove_one.

Returns

Returns the number of events acted on, or 0 if no desired event was found.

12.66.5.100 `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↔_s</i>	The start time of the selection.
<i>tick↔_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.

```
12.66.5.101 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.

```
12.66.5.102 void seq64::sequence::select_all_notes ( bool inverse = false ) [inline]
```

```
12.66.5.103 int seq64::sequence::get_num_selected_notes ( ) const
```

Threadsafe

Returns

Returns m_events.count_selected_notes().

```
12.66.5.104 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()`.

12.66.5.105 `void seq64::sequence::select_all ()`

Threadsafe

12.66.5.106 `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.
-# 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

12.66.5.107 `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.
-------------------	---

12.66.5.108 `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 bugs 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.

12.66.5.109 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↔ _h</i>	Side-effect return reference for the high note.
out	<i>tick_f</i>	Side-effect return reference for the finish time.
out	<i>note↔ _l</i>	Side-effect return reference for the low note.

12.66.5.110 void seq64::sequence::get_clipboard_box (midipulse & tick_s, int & note_h, midipulse & tick_f, int & note_l)

Note the common-code between this function and [get_selected_box\(\)](#). Also note we could return a boolean indicating if the return values were filled in.

Threadsafe

Parameters

out	<i>tick_s</i>	Side-effect return reference for the start time.
out	<i>note↔ _h</i>	Side-effect return reference for the high note.
out	<i>tick_f</i>	Side-effect return reference for the finish time.
out	<i>note↔ _l</i>	Side-effect return reference for the low note.

12.66.5.111 **midipulse** seq64::sequence::adjust_timestamp (**midipulse** *t*, **bool** *isnoteoff* = *false*)

If the timestamp is greater than `m_length`, we do round robin magic. Taken from similar code in [move_selected_↔notes\(\)](#) and [grow_selected\(\)](#). Be careful using this function.

Parameters

<i>t</i>	Provides the timestamp to be adjusted based on <code>m_length</code> .
<i>isnoteoff</i>	Used for "expanding" the timestamp from 0 to just less than <code>m_length</code> , if necessary. Should be set to true only for Note Off events; it defaults to false, which means to wrap the events around the end of the sequence if necessary, and is used only in movement, not in growth.

Returns

Returns the adjusted timestamp.

12.66.5.112 **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.

12.66.5.113 **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.

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.

12.66.5.114 `void seq64::sequence::add_note (midipulse tick, midipulse len, int note, bool paint = false)`

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 through 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.

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.

12.66.5.115 `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 through 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.

12.66.5.116 void 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.

Todo Consider adding a feature where event's are rejected if their channel doesn't match that of the sequence. This has been a complaint of some people. Would modify the [add_event\(\)](#) and [add_note\(\)](#) functions.

Threadsafe

Parameters

<i>ev</i>	Provides the event to stream.
-----------	-------------------------------

12.66.5.117 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.

12.66.5.118 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.

12.66.5.119 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.

12.66.5.120 void seq64::sequence::grow_selected (midipulse *delta_tick*)

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_tick</i>	An offset for each linked event's timestamp.
-------------------	--

12.66.5.121 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.
-------------------	--

12.66.5.122 `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.

12.66.5.123 `void seq64::sequence::mark_selected ()`

Threadsafe

12.66.5.124 `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.

12.66.5.125 `void seq64::sequence::unpaint_all ()`

Threadsafe

12.66.5.126 `void seq64::sequence::unselect ()`

Threadsafe

12.66.5.127 `void seq64::sequence::verify_and_link ()`

Threadsafe

12.66.5.128 `void seq64::sequence::link_new ()`

Threadsafe

12.66.5.129 `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. However, commenting out this setting doesn't have any effect that we can see with a quick look at the user-interface.

12.66.5.130 `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.
-------------	-------------------

12.66.5.131 void seq64::sequence::play_note_off (int *note*)

Threadsafe

Parameters

<i>note</i>	The note to turn off.
-------------	-----------------------

12.66.5.132 void seq64::sequence::off_playing_notes ()

This function does not bother checking if m_masterbus is a null pointer.

Threadsafe

12.66.5.133 void seq64::sequence::pause ()

The [reset\(\)](#) function is currently not called when pausing, but we still need the note-shutoff capability to prevent notes from lingering. Not that we do not call set_playing(false)... it disarms the sequence, which we do not want upon pausing.

12.66.5.134 void seq64::sequence::reset (bool *live_mode*)

Note that, in live mode, the user controls playback, while otherwise JACK or the performance/song editor controls playback. (We're still a bit confounded about these modes, alas.)

Parameters

<i>live_mode</i>	True if live mode is on. This means that JACK transport is not in control of playback.
------------------	--

12.66.5.135 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

12.66.5.136 void seq64::sequence::reset_draw_trigger_marker ()

Threadsafe

12.66.5.137 **draw_type** 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.

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.

12.66.5.138 **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.

12.66.5.139 **bool** seq64::sequence::get_next_event (midibyte *status*, midibyte *cc*, midipulse * *tick*, midibyte * *d0*, midibyte * *d1*, bool * *selected*)

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.

12.66.5.140 `bool seq64::sequence::get_next_event (midibyte * status, midibyte * cc)`

Then set the status and control character parameters using that event.

12.66.5.141 `bool seq64::sequence::get_next_trigger (midipulse * tick_on, midipulse * tick_off, bool * selected, midipulse * tick_offset)`

12.66.5.142 `void seq64::sequence::fill_container (midi_container & c, int tracknumber)`

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).

Parameters

<i>c</i>	Provides the <code>std::list</code> object to push events to the front, which thus inserts them in backwards order. (These events are then popped back, which restores the order, with some exceptions).
<i>tracknumber</i>	Provides the track number. This number is masked into the track information.

12.66.5.143 `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 <code>snap_tick</code> 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 seem to be either 1 or 2.
<i>linked</i>	False by default, this parameter indicates if marked events are to be relinked, as far as we can tell.

12.66.5.144 `void seq64::sequence::push_quantize (midibyte status, midibyte cc, midipulse snap_tick, int divide, bool linked = false)`

12.66.5.145 `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.

12.66.5.146 **midibyte** seq64::sequence::musical_key () const [inline]

12.66.5.147 **void** seq64::sequence::musical_key (int *key*) [inline]

12.66.5.148 **midibyte** seq64::sequence::musical_scale () const [inline]

12.66.5.149 **void** seq64::sequence::musical_scale (int *scale*) [inline]

12.66.5.150 **int** seq64::sequence::background_sequence () const [inline]

12.66.5.151 **void** seq64::sequence::background_sequence (int *bs*) [inline]

Disabling the sequence number (setting it to SEQ64_SEQUENCE_LIMIT) is valid.

12.66.5.152 **void** seq64::sequence::show_events () const

12.66.5.153 **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.
------------------	--

12.66.5.154 **midipulse** seq64::sequence::note_off_margin () const [inline]

12.66.5.155 **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.
----------	---

12.66.5.156 `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

12.66.5.157 `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.
-----------------------	---------------------------------

12.66.5.158 `void seq64::sequence::split_trigger (trigger & trig, midipulse splittick) [private]`

This is the private overload of `split_trigger`.

Threadsafe

Parameters

<i>trig</i>	Provides the original trigger, and also holds the changes made to that trigger as it is shortened.
<i>splittick</i>	The position just after where the original trigger will be truncated, and the new trigger begins.

12.66.5.159 `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.
------------------	---

12.66.5.160 **midipulse seq64::sequence::adjust_offset (midipulse offset)** [private]

12.66.5.161 **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.
----------	---

12.66.5.162 **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.
----------	--

12.66.5.163 **void seq64::sequence::remove_all ()** [private]

Unsets the modified flag. (Why?) Also see the new [copy_events\(\)](#) function.

12.66.6 Friends And Related Function Documentation

12.66.6.1 **friend class perform** [friend]

12.66.6.2 **friend class triggers** [friend]

12.66.7 Field Documentation

12.66.7.1 **event_list seq64::sequence::m_events_clipboard** [static], [private]

Being static allows for copy/paste between patterns.

12.66.7.2 **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 midifile 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.

12.66.7.3 `event_list seq64::sequence::m_events` [private]

12.66.7.4 `triggers seq64::sequence::m_triggers` [private]

12.66.7.5 `EventStack seq64::sequence::m_events_undo` [private]

12.66.7.6 `EventStack seq64::sequence::m_events_redo` [private]

12.66.7.7 `event_list::iterator seq64::sequence::m_iterator_draw` [private]

12.66.7.8 `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.

12.66.7.9 `midibyte seq64::sequence::m_bus` [private]

12.66.7.10 `bool seq64::sequence::m_song_mute` [private]

12.66.7.11 `int seq64::sequence::m_notes_on` [private]

12.66.7.12 `mastermidibus* seq64::sequence::m_masterbus` [private]

12.66.7.13 `int seq64::sequence::m_playing_notes[SEQ64_MIDI_NOTES_MAX]` [private]

It is used when muting, to shut off the notes that are playing.

12.66.7.14 `bool seq64::sequence::m_was_playing` [private]

12.66.7.15 `bool seq64::sequence::m_playing` [private]

12.66.7.16 `bool seq64::sequence::m_recording` [private]

12.66.7.17 `bool seq64::sequence::m_quantized_rec` [private]

12.66.7.18 `bool seq64::sequence::m_thru` [private]

12.66.7.19 `bool seq64::sequence::m_queued` [private]

12.66.7.20 `bool seq64::sequence::m_dirty_main` [private]

Provides the main dirtiness flag.

12.66.7.21 `bool seq64::sequence::m_dirty_edit` [private]

12.66.7.22 `bool seq64::sequence::m_dirty_perf` [private]

12.66.7.23 `bool seq64::sequence::m_dirty_names` [private]

12.66.7.24 `bool seq64::sequence::m_editing` [private]

12.66.7.25 `bool seq64::sequence::m_raise` [private]

It allows a sequence editor window to pop up if not already raised, in [seqedit::timeout\(\)](#).

12.66.7.26 `std::string seq64::sequence::m_name` [private]

12.66.7.27 `midipulse seq64::sequence::m_last_tick` [private]

Provides the last tick played.

12.66.7.28 `midipulse seq64::sequence::m_queued_tick` [private]

12.66.7.29 `midipulse seq64::sequence::m_trigger_offset` [private]

12.66.7.30 `const int seq64::sequence::m_maxbeats` [private]

Hardwired to `c_maxbeats` at present.

12.66.7.31 `int seq64::sequence::m_ppqn` [private]

12.66.7.32 `int seq64::sequence::m_seq_number` [private]

This number is set in the [perform::install_sequence\(\)](#) function.

12.66.7.33 `midipulse seq64::sequence::m_length` [private]

This value should be a power of two when used as a bar unit.

12.66.7.34 `midipulse seq64::sequence::m_snap_tick` [private]

It starts out as the value `m_ppqn / 4`.

12.66.7.35 `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.

12.66.7.36 `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.

12.66.7.37 `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.

12.66.7.38 `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.

12.66.7.39 `int 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.

12.66.7.40 `int seq64::sequence::m_rec_vol` `[private]`

12.66.7.41 `midibyte seq64::sequence::m_musical_key` `[private]`

If the value is `SEQ64_KEY_OF_C`, then there is no musical key to be set.

12.66.7.42 `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.

12.66.7.43 `int seq64::sequence::m_background_sequence` `[private]`

If the value is greater than `max_sequence()`, then there is no background sequence to be set.

12.66.7.44 `mutex seq64::sequence::m_mutex` `[mutable], [private]`

Made mutable for use in certain locked getter functions.

12.66.7.45 `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.

12.67 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 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.

12.67.1 Detailed Description

This class is used in playback, and is contained in the triggers class.

12.67.2 Constructor & Destructor Documentation

12.67.2.1 `seq64::trigger::trigger ()` `[inline]`

12.67.3 Member Function Documentation

12.67.3.1 `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.

12.67.3.2 `midipulse seq64::trigger::tick_start () const` `[inline]`

12.67.3.3 `void seq64::trigger::tick_start (midipulse s)` `[inline]`

12.67.3.4 `void seq64::trigger::increment_tick_start (midipulse s)` `[inline]`

12.67.3.5 `void seq64::trigger::decrement_tick_start (midipulse s)` `[inline]`

12.67.3.6 `midipulse seq64::trigger::tick_end () const` `[inline]`

12.67.3.7 `void seq64::trigger::tick_end (midipulse e)` `[inline]`

12.67.3.8 `void seq64::trigger::increment_tick_end (midipulse s)` `[inline]`

12.67.3.9 `void seq64::trigger::decrement_tick_end (midipulse s)` `[inline]`

12.67.3.10 `midipulse seq64::trigger::offset () const` `[inline]`

12.67.3.11 `void seq64::trigger::offset (midipulse o)` `[inline]`

12.67.3.12 `void seq64::trigger::increment_offset (midipulse s)` `[inline]`

12.67.3.13 `void seq64::trigger::decrement_offset (midipulse s)` `[inline]`

12.67.3.14 `bool seq64::trigger::selected () const` `[inline]`

12.67.3.15 `void seq64::trigger::selected (bool s)` `[inline]`

12.67.4 Field Documentation

12.67.4.1 `midipulse seq64::trigger::m_tick_start` `[private]`

12.67.4.2 `midipulse seq64::trigger::m_tick_end` `[private]`

12.67.4.3 `midipulse seq64::trigger::m_offset` `[private]`

12.67.4.4 `bool seq64::trigger::m_selected` `[private]`

12.68 seq64::triggers Class Reference

The triggers class is a receptable the triggers that can be used with a sequence object.

Public Types

- `typedef std::list< trigger > List`
Exposes the triggers type, currently needed for [midi_container](#) only.

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.
- `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 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 `split` (`trigger` &trig, `midipulse` splittick)
Splits the trigger given by the parameter into two triggers.
- 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` ()
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, int which=2)
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.

Private Types

- typedef std::stack< `List` > `Stack`

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.

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.

- 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.

12.68.1 Member Typedef Documentation

12.68.1.1 `typedef std::list<trigger> seq64::triggers::List`

12.68.1.2 `typedef std::stack<List> seq64::triggers::Stack` `[private]`

12.68.2 Constructor & Destructor Documentation

12.68.2.1 `seq64::triggers::triggers (sequence & parent)`

Parameters

<code>parent</code>	The triggers object often needs to tell its parent sequence object what to do (such as stop playing).
---------------------	---

12.68.2.2 `seq64::triggers::~~triggers ()`

12.68.3 Member Function Documentation

12.68.3.1 triggers & seq64::triggers::operator= (const triggers & rhs)

Follows the stock rules for such an operator, but does a little more then just assign member values.

FIXED, BEWARE: Currently, it does not assign them all, so we should create a `partial_copy()` function to do this work, and use it where it is needed.

Parameters

<i>rhs</i>	Provides the "right-hand side" of the assignment operation.
------------	---

Returns

Returns a reference to self, for use in concatenated assignment operations.

12.68.3.2 void seq64::triggers::set_ppqn (int ppqn) [inline]**12.68.3.3 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.

12.68.3.4 List& seq64::triggers::triggerlist () [inline]**12.68.3.5 void seq64::triggers::push_undo ()****12.68.3.6 void seq64::triggers::pop_undo ()****12.68.3.7 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.
----------------	---

12.68.3.8 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 are trigger start/end values before the *end_tick*. If so, then the trigger state is set to true (start only within the tick range) or false (end is within the tick range), and the trigger tick is set to start or end. The first start or end trigger that is past the end tick cause the search to end.

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.

12.68.3.9 `void seq64::triggers::add (midipulse tick, midipulse len, midipulse offset = 0, bool fixoffset = true)`

What is this?

```

is      ie
<      ><      ><      >
es      ee
<      >
XX
es ee
<  >
<>
es      ee
<      >
<      >
es      ee
<      >
<      >

```

Parameters

<i>tick</i>	Provides the tick (pulse) time at which the trigger goes on.
<i>len</i>	Provides the length of the trigger. This value is actually calculated from the "on" value minus the "off" value read from the MIDI file.
<i>offset</i>	This value specifies the offset of the trigger. It is a feature of the <code>c_triggers_new</code> that <code>c_triggers</code> doesn't have. It is the third value in the trigger specification of the Sequencer64 MIDI file.
<i>fixoffset</i>	If true, the offset parameter is modified by adjust_offset() first. We think that basically makes sure it is positive.

12.68.3.10 `void seq64::triggers::adjust_offsets_to_length (midipulse newlength)`

Parameters

<i>newlength</i>	Provides the length to which to adjust the offsets.
------------------	---

COMMON CODE?

COMMON CODE?

12.68.3.11 `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.
------------------	--

12.68.3.12 void seq64::triggers::split (trigger & *trig*, midipulse *splittick*)

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.

12.68.3.13 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.

12.68.3.14 void seq64::triggers::remove (midipulse *tick*)

Parameters

<i>tick</i>	Provides the tick to be examined.
-------------	-----------------------------------

12.68.3.15 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.

12.68.3.16 `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.

12.68.3.17 `bool seq64::triggers::unselect ()`

Returns

Always returns false.

12.68.3.18 `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 (<code>m_tick_start</code>) of the matching trigger.
<i>ender</i>	The destination for the ending tick (<code>m_tick_end</code>) 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.

12.68.3.19 `void seq64::triggers::remove_selected ()`

12.68.3.20 `void seq64::triggers::copy_selected ()`

12.68.3.21 `void seq64::triggers::paste ()`

It pastes at the copy end.

12.68.3.22 `bool seq64::triggers::move_selected (midipulse tick, bool fixoffset, int which = 2)`

```

    mintick][0          1][maxtick
                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.

```

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.

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.

12.68.3.23 `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.

12.68.3.24 `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.

12.68.3.25 `midipulse seq64::triggers::get_maximum ()`**Returns**

Returns the tick-end for the last trigger, if available. Otherwise, 0 is returned.

12.68.3.26 `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.

12.68.3.27 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
    [      ][      ][ ]

```

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.

12.68.3.28 void seq64::triggers::clear () [inline]

12.68.3.29 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.

12.68.3.30 `trigger seq64::triggers::next_trigger ()`

Returns

Returns the next trigger. If there is none, a default trigger object is returned.

12.68.3.31 `void seq64::triggers::reset_draw_trigger_marker ()` `[inline]`

12.68.3.32 `midipulse seq64::triggers::adjust_offset (midipulse offset)` `[private]`

Parameters

<i>offset</i>	Provides the offset, mod'ed against <code>m_length</code> , 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.

12.68.4 Field Documentation

12.68.4.1 `sequence& seq64::triggers::m_parent` `[private]`

12.68.4.2 `List seq64::triggers::m_triggers` `[private]`

12.68.4.3 `trigger seq64::triggers::m_clipboard` `[private]`

12.68.4.4 `Stack seq64::triggers::m_undo_stack` `[private]`

12.68.4.5 `Stack seq64::triggers::m_redo_stack` `[private]`

12.68.4.6 `List::iterator seq64::triggers::m_iterator_play_trigger` `[private]`

12.68.4.7 `List::iterator seq64::triggers::m_iterator_draw_trigger` `[private]`

12.68.4.8 `bool seq64::triggers::m_trigger_copied` `[private]`

12.68.4.9 `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.

12.68.4.10 `int seq64::triggers::m_length` `[private]`

This might change, we're not yet sure.

12.69 `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.

12.69.1 Detailed Description

Will later make the size adjustable, if it makes sense to do so.

12.69.2 Constructor & Destructor Documentation

12.69.2.1 `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.
-------------	--

12.69.2.2 `seq64::user_instrument::user_instrument (const user_instrument & rhs)`

Parameters

<i>rhs</i>	The sources of the data for the copy.
------------	---------------------------------------

12.69.3 Member Function Documentation

12.69.3.1 `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.

12.69.3.2 `bool seq64::user_instrument::is_valid () const` `[inline]`

12.69.3.3 `void seq64::user_instrument::set_defaults ()`

Also invalidates the object.

12.69.3.4 `const std::string& seq64::user_instrument::name () const` `[inline]`

12.69.3.5 `int seq64::user_instrument::controller_count () const` `[inline]`

12.69.3.6 `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).

12.69.3.7 `const std::string & seq64::user_instrument::controller_name (int c) const`

Parameters

<code>c</code>	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.

12.69.3.8 `bool seq64::user_instrument::controller_active (int c) const`

Parameters

<code>c</code>	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.

12.69.3.9 `void seq64::user_instrument::set_controller (int c, const std::string & cname, bool isactive)`

Parameters

<code>c</code>	The index of the desired controller.
<code>cname</code>	The name of the controller to be set as the controller name.
<code>isactive</code>	A flag that indicates if the desired controller is active.

12.69.3.10 `void seq64::user_instrument::set_name (const std::string & instname)` `[private]`

Too tricky?

Parameters

<code>instname</code>	The name of the instrument, valid only if it is not empty.
-----------------------	--

12.69.3.11 `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.
------------	---

12.69.4 Field Documentation

12.69.4.1 `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.

12.69.4.2 `int seq64::user_instrument::m_controller_count [private]`

Often, the "user" configuration file has only a few out of the 128 assigned explicitly.

12.69.4.3 `user_instrument_t seq64::user_instrument::m_instrument_def [private]`

12.70 seq64::user_instrument_t Struct Reference

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

12.70.1 Field Documentation

12.70.1.1 `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).

12.70.1.2 `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.

12.70.1.3 `bool seq64::user_instrument_t::controllers_active[SEQ64_MIDI_CONTROLLER_MAX]`

If false, it might be an unsupported controller or a General MIDI device.

12.71 `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.

12.71.1 Detailed Description

Will later make the size adjustable, if it makes sense to do so.

12.71.2 Constructor & Destructor Documentation

12.71.2.1 `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.
-------------	--

12.71.2.2 `seq64::user_midi_bus::user_midi_bus (const user_midi_bus & rhs)`

Parameters

<i>rhs</i>	The sources of the data for the copy.
------------	---------------------------------------

12.71.3 Member Function Documentation

12.71.3.1 `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.

12.71.3.2 `bool seq64::user_midi_bus::is_valid () const` `[inline]`

12.71.3.3 `void seq64::user_midi_bus::set_defaults ()`

Also invalidates the object. All 16 of the channels are set to SEQ64_GM_INSTRUMENT_FLAG (-1).

12.71.3.4 `const std::string& seq64::user_midi_bus::name () const [inline]`

12.71.3.5 `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.

12.71.3.6 `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).

12.71.3.7 `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.

12.71.3.8 `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.

12.71.3.9 `void seq64::user_midi_bus::set_name (const std::string & name) [inline], [private]`

12.71.3.10 `void seq64::user_midi_bus::copy_definitions (const user_midi_bus & rhs) [private]`

Does not include the validity flag.

12.71.4 Field Documentation

12.71.4.1 `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.

12.71.4.2 `int seq64::user_midi_bus::m_channel_count` `[private]`

Often, the "user" configuration file has only a few out of the 16 assigned explicitly.

12.71.4.3 `user_midi_bus_t seq64::user_midi_bus::m_midi_bus_def` `[private]`

12.72 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.

12.72.1 Field Documentation

12.72.1.1 `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.

12.72.1.2 `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.

12.73 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*
- bool [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*
- int [window_redraw_rate](#) () const
 - *'Getter' function for member m_window_redraw_rate_ms*
- 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*
- int [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 [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](#) (bool flag)
'Setter' function for member m_progress_bar_colored
 - void [progress_bar_thick](#) (bool flag)
'Setter' function for member m_progress_bar_thick
 - void [window_redraw_rate](#) (int ms)
'Setter' function for member m_window_redraw_rate_ms
 - 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.

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](#) (int 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

- 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.
- bool [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...
- int [m_window_redraw_rate_ms](#)
Provides the global setting for redraw rate of windows.
- 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 testing.

- int [m_total_seqs](#)
- 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](#)

12.73.1 Detailed Description

These settings will eventually be made part of the "user" settings file.

12.73.2 Member Typedef Documentation

12.73.2.1 `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.

12.73.2.2 `typedef std::vector<user_midi_bus>::iterator seq64::user_settings::BussIterator` [private]

12.73.2.3 `typedef std::vector<user_midi_bus>::const_iterator seq64::user_settings::BussConstIterator`
[private]

12.73.2.4 `typedef std::vector<user_instrument> seq64::user_settings::Instruments` [private]

Internal type for the container of [user_instrument](#) objects.

12.73.2.5 `typedef std::vector<user_instrument>::iterator seq64::user_settings::InstrumentIterator`
[private]

12.73.2.6 `typedef std::vector<user_instrument>::const_iterator seq64::user_settings::InstrumentConstIterator`
[private]

12.73.3 Member Enumeration Documentation

12.73.3.1 `enum seq64::user_settings::mainwid_grid_style_t` [private]

Enumerator

grid_style_normal Provides a setting to control the overall style of grid-drawing for the pattern slots in mainwid. These values can be specified in the [user-interface-settings] section of the "user" configuration file.

The grid background color is the normal background color for the current GTK theme. The box is drawn with brackets on either side.

grid_style_white 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.

12.73.4 Constructor & Destructor Documentation

12.73.4.1 `seq64::user_settings::user_settings ()`

12.73.4.2 `seq64::user_settings::user_settings (const user_settings & rhs)`

12.73.5 Member Function Documentation

12.73.5.1 `user_settings & seq64::user_settings::operator= (const user_settings & rhs)`

12.73.5.2 `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!

12.73.5.3 void seq64::user_settings::normalize ()

12.73.5.4 bool seq64::user_settings::add_bus (const std::string & *alias*)

12.73.5.5 bool seq64::user_settings::add_instrument (const std::string & *instname*)

12.73.5.6 const user_midi_bus& seq64::user_settings::bus (int *index*) [inline]

Cannot append the const specifier.

12.73.5.7 const user_instrument& seq64::user_settings::instrument (int *index*) [inline]

Cannot append the const specifier.

12.73.5.8 int seq64::user_settings::bus_count () const [inline]

12.73.5.9 void seq64::user_settings::set_bus_instrument (int *index*, int *channel*, int *instrum*)

12.73.5.10 int seq64::user_settings::bus_instrument (int *buss*, int *channel*) [inline]

12.73.5.11 const std::string& seq64::user_settings::bus_name (int *buss*) [inline]

12.73.5.12 int seq64::user_settings::instrument_count () const [inline]

12.73.5.13 void seq64::user_settings::set_instrument_controllers (int *index*, int *cc*, const std::string & *ccname*, bool *isactive*)

12.73.5.14 const std::string& seq64::user_settings::instrument_name (int *instrum*) [inline]

12.73.5.15 const std::string& seq64::user_settings::instrument_name (int *buss*, int *channel*) [inline]

12.73.5.16 bool seq64::user_settings::instrument_controller_active (int *instrum*, int *cc*) [inline]

12.73.5.17 bool seq64::user_settings::controller_active (int *buss*, int *channel*, int *cc*) [inline]

It also has a shorter name.

12.73.5.18 const std::string& seq64::user_settings::instrument_controller_name (int *instrum*, int *cc*) [inline]

12.73.5.19 const std::string& seq64::user_settings::controller_name (int *buss*, int *channel*, int *cc*) [inline]

It also has a shorter name.

- 12.73.5.20 int seq64::user_settings::grid_style () const [inline]
- 12.73.5.21 bool seq64::user_settings::grid_is_normal () const [inline]
- 12.73.5.22 bool seq64::user_settings::grid_is_white () const [inline]
- 12.73.5.23 bool seq64::user_settings::grid_is_black () const [inline]
- 12.73.5.24 int seq64::user_settings::grid_brackets () const [inline]
- 12.73.5.25 int seq64::user_settings::mainwnd_rows () const [inline]
- 12.73.5.26 int seq64::user_settings::mainwnd_cols () const [inline]
- 12.73.5.27 int seq64::user_settings::seqs_in_set () const [inline]
- 12.73.5.28 int seq64::user_settings::gmute_tracks () const [inline]
- 12.73.5.29 int seq64::user_settings::max_sets () const [inline]
- 12.73.5.30 int seq64::user_settings::max_sequence () const [inline]
- 12.73.5.31 int seq64::user_settings::text_x () const [inline]
- 12.73.5.32 int seq64::user_settings::text_y () const [inline]
- 12.73.5.33 int seq64::user_settings::seqchars_x () const [inline]
- 12.73.5.34 int seq64::user_settings::seqchars_y () const [inline]
- 12.73.5.35 int seq64::user_settings::seqarea_x () const [inline]
- 12.73.5.36 int seq64::user_settings::seqarea_y () const [inline]
- 12.73.5.37 int seq64::user_settings::seqarea_seq_x () const [inline]
- 12.73.5.38 int seq64::user_settings::seqarea_seq_y () const [inline]
- 12.73.5.39 int seq64::user_settings::mainwid_border () const [inline]
- 12.73.5.40 int seq64::user_settings::mainwid_spacing () const [inline]
- 12.73.5.41 int seq64::user_settings::mainwid_x () const [inline]
- 12.73.5.42 int seq64::user_settings::mainwid_y () const [inline]
- 12.73.5.43 int seq64::user_settings::control_height () const [inline]
- 12.73.5.44 int seq64::user_settings::zoom () const [inline]
- 12.73.5.45 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.

```

12.73.5.46  bool seq64::user_settings::global_seq_feature ( ) const  [inline]

12.73.5.47  void seq64::user_settings::global_seq_feature ( bool flag )  [inline]

12.73.5.48  int seq64::user_settings::seqedit_scale ( ) const  [inline]

12.73.5.49  void seq64::user_settings::seqedit_scale ( int scale )  [inline]

12.73.5.50  int seq64::user_settings::seqedit_key ( ) const  [inline]

12.73.5.51  void seq64::user_settings::seqedit_key ( int key )  [inline]

12.73.5.52  int seq64::user_settings::seqedit_bgsequence ( ) const  [inline]

12.73.5.53  void seq64::user_settings::seqedit_bgsequence ( int seqnum )  [inline]

12.73.5.54  bool seq64::user_settings::use_new_font ( ) const  [inline]

12.73.5.55  bool seq64::user_settings::allow_two_perfedits ( ) const  [inline]

12.73.5.56  int seq64::user_settings::perf_h_page_increment ( ) const  [inline]

12.73.5.57  int seq64::user_settings::perf_v_page_increment ( ) const  [inline]

12.73.5.58  bool seq64::user_settings::progress_bar_colored ( ) const  [inline]

12.73.5.59  bool seq64::user_settings::progress_bar_thick ( ) const  [inline]

12.73.5.60  int seq64::user_settings::window_redraw_rate ( ) const  [inline]

12.73.5.61  bool seq64::user_settings::save_user_config ( ) const  [inline]

12.73.5.62  void seq64::user_settings::save_user_config ( bool flag )  [inline]

12.73.5.63  void seq64::user_settings::grid_brackets ( int thickness )  [inline],[protected]

12.73.5.64  void seq64::user_settings::grid_style ( int gridstyle )  [protected]

12.73.5.65  void seq64::user_settings::mainwnd_rows ( int value )  [protected]

```

The default value is 4. Dependent values are recalculated after the assignment.

```

12.73.5.66  void seq64::user_settings::mainwnd_cols ( int value )  [protected]

```

The default value is 8. Dependent values are recalculated after the assignment.

12.73.5.67 void seq64::user_settings::max_sets (int *value*) [protected]

The default value is 32. Dependent values are recalculated after the assignment.

12.73.5.68 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.

12.73.5.69 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.

12.73.5.70 void seq64::user_settings::seqchars_x (int *value*) [protected]

12.73.5.71 void seq64::user_settings::seqchars_y (int *value*) [protected]

12.73.5.72 void seq64::user_settings::seqarea_x (int *value*) [protected]

12.73.5.73 void seq64::user_settings::seqarea_y (int *value*) [protected]

12.73.5.74 void seq64::user_settings::seqarea_seq_x (int *value*) [protected]

12.73.5.75 void seq64::user_settings::seqarea_seq_y (int *value*) [protected]

12.73.5.76 void seq64::user_settings::mainwid_border (int *value*) [protected]

The default value is 0. Dependent values are recalculated after the assignment.

12.73.5.77 void seq64::user_settings::mainwid_spacing (int *value*) [protected]

The default value is 2. Dependent values are recalculated after the assignment.

12.73.5.78 void seq64::user_settings::control_height (int *value*) [protected]

The default value is 0. Dependent values are recalculated after the assignment.

12.73.5.79 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.

12.73.5.80 `int seq64::user_settings::midi_ppqn () const [inline]`

12.73.5.81 `int seq64::user_settings::midi_beats_per_bar () const [inline]`

12.73.5.82 `int seq64::user_settings::midi_beats_per_minute () const [inline]`

12.73.5.83 `int seq64::user_settings::midi_beat_width () const [inline]`

12.73.5.84 `char seq64::user_settings::midi_buss_override () const [inline]`

12.73.5.85 `int seq64::user_settings::min_zoom () const [inline]`

12.73.5.86 `int seq64::user_settings::max_zoom () const [inline]`

12.73.5.87 `int seq64::user_settings::baseline_ppqn () const [inline]`

12.73.5.88 `void seq64::user_settings::use_new_font (bool flag) [inline]`

12.73.5.89 `void seq64::user_settings::allow_two_perfedits (bool flag) [inline]`

12.73.5.90 `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)

12.73.5.91 `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)

12.73.5.92 `void seq64::user_settings::progress_bar_colored (bool flag) [inline]`

12.73.5.93 `void seq64::user_settings::progress_bar_thick (bool flag) [inline]`

12.73.5.94 `void seq64::user_settings::window_redraw_rate (int ms) [inline]`

12.73.5.95 `void seq64::user_settings::midi_ppqn (int value)`

The default value is 192.

12.73.5.96 `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".)

12.73.5.97 void seq64::user_settings::midi_beats_per_bar (int *value*) [protected]

The default value is 4.

12.73.5.98 void seq64::user_settings::midi_beats_per_minute (int *value*) [protected]

The default value is 120.

12.73.5.99 void seq64::user_settings::midi_beat_width (int *bw*) [protected]

The default value is 4.

12.73.5.100 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.

12.73.5.101 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.

12.73.6 Friends And Related Function Documentation

12.73.6.1 friend class userfile [friend]

12.73.7 Field Documentation

12.73.7.1 Busses seq64::user_settings::m_midi_buses [private]

Since this object is a vector, its size is adjustable.

12.73.7.2 Instruments seq64::user_settings::m_instruments [private]

The size is adjustable, and grows as objects are added.

12.73.7.3 `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.

12.73.7.4 `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.

12.73.7.5 `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.

12.73.7.6 `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".

12.73.7.7 `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.

12.73.7.8 `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.

12.73.7.9 int seq64::user_settings::m_mainwid_spacing [private]

12.73.7.10 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.

12.73.7.11 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.

12.73.7.12 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.

12.73.7.13 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.

12.73.7.14 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.

12.73.7.15 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.

12.73.7.16 `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.

12.73.7.17 `bool seq64::user_settings::m_allow_two_perfedits` `[private]`

Defaults to true.

12.73.7.18 `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.

12.73.7.19 `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.

12.73.7.20 `bool seq64::user_settings::m_progress_bar_colored` `[private]`

This value is hardwired in the `gui_palette_gtk2` module, to red. Really, that is the only color that stands out as well as black.

12.73.7.21 `bool seq64::user_settings::m_progress_bar_thick` `[private]`

2 pixels. It isn't useful to support anything thicker.

12.73.7.22 `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.

12.73.7.23 `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.

12.73.7.24 `int seq64::user_settings::m_text_y [private]`

12.73.7.25 `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.

12.73.7.26 `int seq64::user_settings::m_seqchars_y [private]`

12.73.7.27 `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`.

12.73.7.28 `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.

12.73.7.29 `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.

12.73.7.30 `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.

12.73.7.31 `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`.

12.73.7.32 `int seq64::user_settings::m_total_seqs` [private]

12.73.7.33 `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.

12.73.7.34 `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.

12.73.7.35 `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;
```

12.73.7.36 `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.

12.73.7.37 `int seq64::user_settings::m_seqarea_y` [private]

12.73.7.38 `int seq64::user_settings::m_seqarea_seq_x` [private]

These are used only in the `mainwid` module.

12.73.7.39 `int seq64::user_settings::m_seqarea_seq_y` [private]

12.73.7.40 `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
);
```


12.73.7.41 `int seq64::user_settings::m_mainwid_y` [private]

12.73.7.42 `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.

12.73.7.43 `const int seq64::user_settings::mc_min_zoom` [private]

It's value is 1.

12.73.7.44 `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.

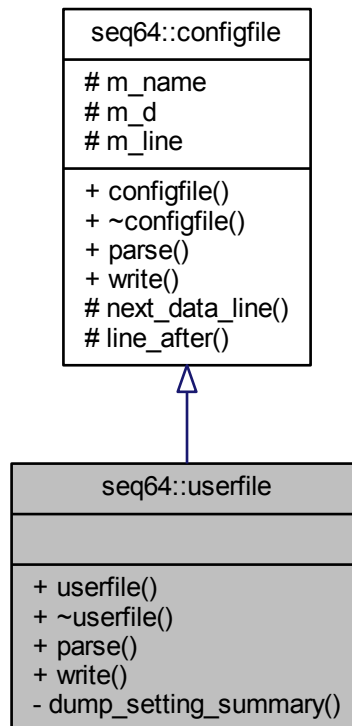
12.73.7.45 `const int seq64::user_settings::mc_baseline_ppqn` [private]

This value is necessary in order to keep user-interface elements stable when different PPQNs are used. It is set to `DEFAULT_PPQN`.

12.74 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

12.74.1 Constructor & Destructor Documentation

12.74.1.1 seq64::userfile::userfile (const std::string & name)

Parameters

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

12.74.1.2 seq64::userfile::~userfile ()

12.74.2 Member Function Documentation

12.74.2.1 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](#).

12.74.2.2 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](#).

12.74.2.3 void seq64::userfile::dump_setting_summary () [private]

Does work only if PLATFORM_DEBUG is defined; see the [user_settings](#) class.

Index

- ~AbstractPerfInput
 - seq64::AbstractPerfInput, [84](#)
- ~automutex
 - seq64::automutex, [85](#)
- ~configfile
 - seq64::configfile, [92](#)
- ~editable_event
 - seq64::editable_event, [98](#)
- ~editable_events
 - seq64::editable_events, [106](#)
- ~event
 - seq64::event, [113](#)
- ~event_list
 - seq64::event_list, [126](#)
- ~eventedit
 - seq64::eventedit, [135](#)
- ~eventslots
 - seq64::eventslots, [146](#)
- ~gui_assistant
 - seq64::gui_assistant, [168](#)
- ~gui_assistant_gtk2
 - seq64::gui_assistant_gtk2, [170](#)
- ~gui_drawingarea_gtk2
 - seq64::gui_drawingarea_gtk2, [174](#)
- ~gui_palette_gtk2
 - seq64::gui_palette_gtk2, [185](#)
- ~gui_window_gtk2
 - seq64::gui_window_gtk2, [189](#)
- ~jack_assistant
 - seq64::jack_assistant, [194](#)
- ~keys_perform
 - seq64::keys_perform, [214](#)
- ~keys_perform_gtk2
 - seq64::keys_perform_gtk2, [226](#)
- ~maintime
 - seq64::maintime, [236](#)
- ~mainwid
 - seq64::mainwid, [241](#)
- ~mainwnd
 - seq64::mainwnd, [253](#)
- ~mastermidibus
 - seq64::mastermidibus, [262](#)
- ~midi_container
 - seq64::midi_container, [271](#)
- ~midi_list
 - seq64::midi_list, [277](#)
- ~midi_splitter
 - seq64::midi_splitter, [281](#)
- ~midi_vector
 - seq64::midi_vector, [287](#)
- ~midibus
 - seq64::midibus, [291](#)
- ~midifile
 - seq64::midifile, [297](#)
- ~optionsfile
 - seq64::optionsfile, [316](#)
- ~perfedit
 - seq64::perfedit, [323](#)
- ~perfnames
 - seq64::perfnames, [331](#)
- ~perform
 - seq64::perform, [346](#)
- ~perfroll
 - seq64::perfroll, [388](#)
- ~perftime
 - seq64::perftime, [401](#)
- ~seqdata
 - seq64::seqdata, [423](#)
- ~seqedit
 - seq64::seqedit, [434](#)
- ~seqevent
 - seq64::seqevent, [449](#)
- ~seqkeys
 - seq64::seqkeys, [459](#)
- ~seqmenu
 - seq64::seqmenu, [468](#)
- ~seqroll
 - seq64::seqroll, [478](#)
- ~seqtime
 - seq64::seqtime, [495](#)
- ~sequence
 - seq64::sequence, [506](#)
- ~triggers
 - seq64::triggers, [542](#)
- ~userfile
 - seq64::userfile, [579](#)
- about_dialog
 - seq64::mainwnd, [254](#)
- AbstractPerfInput
 - seq64::AbstractPerfInput, [84](#)
- active
 - seq64::midi_control, [274](#)
- add
 - seq64::editable_events, [107](#)
 - seq64::event_list, [127](#)
 - seq64::triggers, [544](#)
- add_bus
 - seq64::user_settings, [566](#)

- add_event
 - seq64::sequence, [513](#), [524](#)
- add_instrument
 - seq64::user_settings, [566](#)
- add_jack_sync_page
 - seq64::options, [314](#)
- add_keyboard_page
 - seq64::options, [314](#)
- add_long
 - seq64::midi_container, [272](#)
- add_midi_clock_page
 - seq64::options, [314](#)
- add_midi_input_page
 - seq64::options, [314](#)
- add_mouse_page
 - seq64::options, [314](#)
- add_note
 - seq64::seqroll, [478](#)
 - seq64::sequence, [524](#)
- add_sequence
 - seq64::perform, [349](#)
- add_trigger
 - seq64::midifile, [301](#)
 - seq64::sequence, [514](#)
- add_variable
 - seq64::midi_container, [272](#)
- adding
 - seq64::seqroll, [487](#)
- adj_callback_bpm
 - seq64::mainwnd, [254](#)
- adj_callback_ss
 - seq64::mainwnd, [254](#)
- adjust_offset
 - seq64::sequence, [534](#)
 - seq64::triggers, [549](#)
- adjust_offsets_to_length
 - seq64::triggers, [544](#)
- adjust_timestamp
 - seq64::sequence, [523](#)
- adjust_trigger_offsets_to_length
 - seq64::sequence, [533](#)
- adjustment_dummy
 - seq64, [72](#)
- alias
 - seq64::user_midi_bus_t, [557](#)
- align_selection
 - seq64::seqroll, [485](#)
- all_notes_off
 - seq64::perform, [356](#)
- allow_mod4_mode
 - seq64::rc_settings, [409](#)
- allow_two_perfedits
 - seq64::user_settings, [568](#), [570](#)
- analyze
 - seq64::editable_event, [101](#)
- any_group_unmutes
 - seq64::perform, [355](#)
- any_selected_notes
 - seq64::event_list, [130](#)
 - seq64::sequence, [507](#)
- append_sysex
 - seq64::event, [118](#)
- apply_length
 - seq64::seqedit, [437](#)
- at_bpm_dn
 - seq64::keys_perform, [220](#)
- at_bpm_up
 - seq64::keys_perform, [220](#)
- at_event_edit
 - seq64::keys_perform, [222](#)
- at_group_learn
 - seq64::keys_perform, [221](#)
- at_group_off
 - seq64::keys_perform, [221](#)
- at_group_on
 - seq64::keys_perform, [221](#)
- at_keep_queue
 - seq64::keys_perform, [221](#)
- at_pattern_edit
 - seq64::keys_perform, [222](#)
- at_pause
 - seq64::keys_perform, [222](#)
- at_queue
 - seq64::keys_perform, [221](#)
- at_replace
 - seq64::keys_perform, [220](#)
- at_screenset_dn
 - seq64::keys_perform, [221](#)
- at_screenset_up
 - seq64::keys_perform, [221](#)
- at_set_playing_screenset
 - seq64::keys_perform, [221](#)
- at_show_ui_sequence_key
 - seq64::keys_perform, [222](#)
- at_show_ui_sequence_number
 - seq64::keys_perform, [222](#)
- at_snapshot_1
 - seq64::keys_perform, [221](#)
- at_snapshot_2
 - seq64::keys_perform, [221](#)
- at_start
 - seq64::keys_perform, [222](#)
- at_stop
 - seq64::keys_perform, [222](#)
- auto_option_save
 - seq64::rc_settings, [409](#)
- automutex
 - seq64::automutex, [85](#)
- BLACK_ON_CYAN
 - seq64::font, [156](#)
- BLACK_ON_YELLOW
 - seq64::font, [156](#)
- BLACK
 - seq64::font, [156](#)
- background_sequence
 - seq64::sequence, [532](#)

- baseline_ppqn
 - seq64::user_settings, 570
- beat_width
 - seq64::midi_timing, 285
- beats
 - seq64::midi_measures, 279
- beats_per_measure
 - seq64::midi_timing, 284
- beats_per_minute
 - seq64::midi_timing, 284
- beats_per_minute_from_tempo
 - seq64, 60
- begin
 - seq64::editable_events, 107
 - seq64::event_list, 126
- bg_color
 - seq64::gui_palette_gtk2, 186
- black
 - seq64::gui_palette_gtk2, 185
- blue
 - seq64::gui_palette_gtk2, 186
- bpm_dn
 - seq64::keys_perform, 215
- bpm_up
 - seq64::keys_perform, 215
- build_details
 - seq64, 65
- bus
 - seq64::user_settings, 566
- bus_count
 - seq64::user_settings, 566
- bus_instrument
 - seq64::user_settings, 566
- bus_name
 - seq64::user_settings, 566
- BussConstIterator
 - seq64::user_settings, 565
- BussIterator
 - seq64::user_settings, 564
- bussbyte
 - seq64, 51
- Busses
 - seq64::user_settings, 564
- button
 - seq64::click, 88
 - seq64::options, 313
- button_press
 - seq64::seqroll, 485
- button_press_initial
 - seq64::seqroll, 485
- button_release
 - seq64::seqroll, 485
- c_backsequence
 - seq64, 76
- c_bpmtag
 - seq64, 76
- c_chord_text
 - seq64, 78
- c_controller_names
 - seq64, 73
- c_interval_text
 - seq64, 78
- c_key_text
 - seq64, 78
- c_mainwid_x
 - seq64, 80
- c_mainwid_y
 - seq64, 81
- c_max_busses
 - seq64, 78
- c_max_instruments
 - seq64, 78
- c_midi_control_bpm_dn
 - seq64, 76
- c_midi_control_bpm_up
 - seq64, 76
- c_midi_control_mod_glearn
 - seq64, 77
- c_midi_control_mod_gmute
 - seq64, 76
- c_midi_control_mod_queue
 - seq64, 76
- c_midi_control_mod_replace
 - seq64, 76
- c_midi_control_mod_snapshot
 - seq64, 76
- c_midi_control_play_ss
 - seq64, 77
- c_midi_control_ss_dn
 - seq64, 76
- c_midi_control_ss_up
 - seq64, 76
- c_midi_controls
 - seq64, 77
- c_midi_track_ctrl
 - seq64, 76
- c_midibus
 - seq64, 75
- c_midibus_input_size
 - seq64, 75
- c_midibus_output_size
 - seq64, 75
- c_midibus_sysex_chunk
 - seq64, 75
- c_midich
 - seq64, 75
- c_midiCLOCKS
 - seq64, 76
- c_midiCtrl
 - seq64, 76
- c_music_scales
 - seq64, 53
- c_musickey
 - seq64, 76
- c_musicscale
 - seq64, 76

- c_mutegroups
 - seq64, [76](#)
- c_notes
 - seq64, [76](#)
- c_quantize_events
 - seq64, [81](#)
- c_quantize_notes
 - seq64, [81](#)
- c_reserved
 - seq64, [81](#)
- c_scale_blues
 - seq64, [53](#)
- c_scale_c_whole_tone
 - seq64, [53](#)
- c_scale_harmonic_minor
 - seq64, [53](#)
- c_scale_major
 - seq64, [53](#)
- c_scale_major_pentatonic
 - seq64, [53](#)
- c_scale_melodic_minor
 - seq64, [53](#)
- c_scale_minor
 - seq64, [53](#)
- c_scale_minor_pentatonic
 - seq64, [53](#)
- c_scale_off
 - seq64, [53](#)
- c_scale_size
 - seq64, [53](#)
- c_scales_policy
 - seq64, [77](#)
- c_scales_text
 - seq64, [78](#)
- c_scales_transpose_dn
 - seq64, [77](#)
- c_scales_transpose_up
 - seq64, [77](#)
- c_select_all_events
 - seq64, [81](#)
- c_select_all_notes
 - seq64, [81](#)
- c_select_inverse_events
 - seq64, [81](#)
- c_select_inverse_notes
 - seq64, [81](#)
- c_status_queue
 - seq64, [80](#)
- c_status_replace
 - seq64, [80](#)
- c_status_snapshot
 - seq64, [80](#)
- c_swing_notes
 - seq64, [81](#)
- c_tighten_events
 - seq64, [81](#)
- c_tighten_notes
 - seq64, [81](#)
- c_timesig
 - seq64, [76](#)
- c_transpose_h
 - seq64, [81](#)
- c_transpose_notes
 - seq64, [81](#)
- c_triggers
 - seq64, [76](#)
- c_triggers_new
 - seq64, [76](#)
- CYAN_ON_BLACK
 - seq64::font, [156](#)
- calculate_base_sizes
 - seq64::mainwid, [244](#)
- category
 - seq64::editable_event, [99](#), [100](#)
- category_channel_message
 - seq64::editable_event, [97](#)
- category_meta_event
 - seq64::editable_event, [97](#)
- category_name
 - seq64::editable_event, [97](#)
- category_prop_event
 - seq64::editable_event, [97](#)
- category_string
 - seq64::editable_event, [99](#)
- category_system_message
 - seq64::editable_event, [97](#)
- category_t
 - seq64::editable_event, [97](#)
- change_event_data_range
 - seq64::sequence, [525](#)
- change_focus
 - seq64::eventedit, [137](#)
 - seq64::seqedit, [440](#)
- change_horz
 - seq64::perfroll, [390](#)
 - seq64::perftime, [401](#)
 - seq64::seqdata, [424](#)
 - seq64::seqevent, [452](#)
 - seq64::seqroll, [483](#)
 - seq64::seqtime, [495](#)
- change_vert
 - seq64::eventslots, [151](#)
 - seq64::perfnames, [332](#)
 - seq64::perfroll, [390](#)
 - seq64::seqkeys, [461](#)
 - seq64::seqroll, [483](#)
- channel_count
 - seq64::user_midi_bus, [556](#)
- channel_max
 - seq64::user_midi_bus, [556](#)
- channel_string
 - seq64::editable_event, [101](#)
- char_height
 - seq64::font, [157](#)
- char_width
 - seq64::font, [157](#)

- CharList
 - seq64::midi_list, [277](#)
- CharVector
 - seq64::midi_vector, [287](#)
- check_channel
 - seq64::event, [114](#)
- check_queued_tick
 - seq64::sequence, [511](#)
- checklen
 - seq64::midifile, [301](#)
- choose_file
 - seq64::mainwnd, [257](#)
- choose_ppqn
 - seq64, [71](#)
- clamp
 - seq64, [73](#)
- clamp_track
 - seq64::perform, [375](#)
- clear
 - seq64::editable_events, [108](#)
 - seq64::event_list, [128](#)
 - seq64::triggers, [548](#)
- clear_all
 - seq64::perform, [348](#)
- clear_flags
 - seq64::seqroll, [486](#)
- clear_link
 - seq64::event, [119](#)
- clear_links
 - seq64::event_list, [129](#)
- clear_old
 - seq64::seqroll, [486](#)
- clear_selected
 - seq64::seqroll, [486](#)
- clear_sequence_triggers
 - seq64::perform, [350](#)
- clear_triggers
 - seq64::sequence, [518](#)
- clear_window
 - seq64::gui_drawingarea_gtk2, [175](#)
- click
 - seq64::click, [87](#)
- client
 - seq64::jack_assistant, [198](#)
- client_name
 - seq64::jack_assistant, [198](#)
- client_open
 - seq64::jack_assistant, [199](#)
- client_uuid
 - seq64::jack_assistant, [198](#)
- clip_timestamp
 - seq64::sequence, [523](#)
- clock
 - seq64::mastermidibus, [264](#)
 - seq64::midibus, [292](#)
- clock_callback_mod
 - seq64::options, [313](#)
- clock_callback_off
 - seq64::options, [313](#)
- clock_callback_on
 - seq64::options, [313](#)
- clock_e
 - seq64, [52](#)
- clock_mod_callback
 - seq64::options, [313](#)
- clock_tick_duration_bogus
 - seq64, [62](#)
- clock_ticks_from_ppqn
 - seq64, [63](#)
- clocks_per_metronome
 - seq64::sequence, [508](#)
- close_out
 - seq64::eventedit, [138](#)
- collapse
 - seq64::perfedit, [325](#)
 - seq64::perform, [352](#)
- Color
 - seq64::font, [156](#)
 - seq64::gui_palette_gtk2, [184](#)
- complete_paste
 - seq64::seqroll, [481](#)
- condition_var
 - seq64::condition_var, [90](#)
- config_directory
 - seq64::rc_settings, [411](#), [412](#)
- config_filename
 - seq64::rc_settings, [411](#), [412](#)
- config_filename_alt
 - seq64::rc_settings, [411](#), [412](#)
- config_filespec
 - seq64::rc_settings, [409](#)
- configfile
 - seq64::configfile, [92](#)
- const_iterator
 - seq64::editable_events, [105](#)
 - seq64::event_list, [126](#)
- continue_from
 - seq64::mastermidibus, [265](#)
 - seq64::midibus, [292](#)
- control_height
 - seq64::user_settings, [567](#), [569](#)
- controller_active
 - seq64::user_instrument, [552](#)
 - seq64::user_settings, [566](#)
- controller_count
 - seq64::user_instrument, [552](#)
- controller_max
 - seq64::user_instrument, [552](#)
- controller_name
 - seq64::user_instrument, [552](#)
 - seq64::user_settings, [566](#)
- controllers
 - seq64::user_instrument_t, [553](#)
- controllers_active
 - seq64::user_instrument_t, [554](#)
- convert_sel_box_to_rect

- seq64::seqroll, [483](#)
- convert_t
 - seq64::seqevent, [452](#)
- convert_tn
 - seq64::seqroll, [482](#)
- convert_tn_box_to_rect
 - seq64::seqroll, [482](#)
- convert_x
 - seq64::perfroll, [390](#)
 - seq64::seqdata, [424](#)
 - seq64::seqevent, [452](#)
- convert_xy
 - seq64::perfroll, [389](#)
 - seq64::seqroll, [482](#)
- convert_y
 - seq64::eventslots, [150](#)
 - seq64::perfnames, [332](#)
 - seq64::seqkeys, [460](#)
- copy
 - seq64::perfedit, [325](#)
 - seq64::perform, [352](#)
 - seq64::triggers, [548](#)
- copy_definitions
 - seq64::user_instrument, [553](#)
 - seq64::user_midi_bus, [556](#)
- copy_events
 - seq64::sequence, [532](#)
- copy_selected
 - seq64::sequence, [521](#)
 - seq64::triggers, [546](#)
- copy_selected_trigger
 - seq64::sequence, [517](#)
- copy_triggers
 - seq64::perform, [352](#)
 - seq64::sequence, [518](#)
- count
 - seq64::editable_events, [107](#)
 - seq64::event_list, [127](#)
 - seq64::midi_splitter, [282](#)
- count_selected_events
 - seq64::event_list, [130](#)
- count_selected_notes
 - seq64::event_list, [130](#)
- create_lash_driver
 - seq64, [70](#)
- create_menu_image
 - seq64::seqedit, [439](#)
- create_menus
 - seq64::seqedit, [438](#)
- current_event
 - seq64::editable_events, [108](#)
- current_index
 - seq64::eventslots, [146](#)
- current_screen_set_notepad
 - seq64::perform, [354](#)
- current_seq
 - seq64::seqmenu, [468](#)
- current_x
 - seq64::gui_drawingarea_gtk2, [174](#)
- current_y
 - seq64::gui_drawingarea_gtk2, [174](#)
- cut_selected
 - seq64::sequence, [521](#)
- cut_selected_trigger
 - seq64::sequence, [517](#)
- DRAW_FIN
 - seq64, [54](#)
- DRAW_NORMAL_LINKED
 - seq64, [54](#)
- DRAW_NOTE_OFF
 - seq64, [54](#)
- DRAW_NOTE_ON
 - seq64, [54](#)
- dark_cyan
 - seq64::gui_palette_gtk2, [186](#)
- dark_grey
 - seq64::gui_palette_gtk2, [186](#)
- dark_orange
 - seq64::gui_palette_gtk2, [186](#)
- data
 - seq64::event, [119](#)
 - seq64::midi_control, [274](#)
- data_string
 - seq64::editable_event, [101](#)
- decrement_beats_per_minute
 - seq64::perform, [368](#)
- decrement_bottom
 - seq64::eventslots, [152](#)
- decrement_current
 - seq64::eventslots, [152](#)
- decrement_data1
 - seq64::event, [118](#)
- decrement_data2
 - seq64::event, [118](#)
- decrement_offset
 - seq64::trigger, [539](#)
- decrement_screenset
 - seq64::perform, [368](#)
- decrement_selected
 - seq64::sequence, [526](#)
- decrement_tick_end
 - seq64::trigger, [539](#)
- decrement_tick_start
 - seq64::trigger, [539](#)
- decrement_top
 - seq64::eventslots, [151](#)
- deinit
 - seq64::jack_assistant, [195](#)
- deinit_in
 - seq64::midibus, [291](#)
- deinit_jack
 - seq64::perform, [371](#)
- del_selected_trigger
 - seq64::sequence, [517](#)
- del_trigger
 - seq64::sequence, [514](#)

- delete_current_event
 - seq64::eventslots, [147](#)
- delete_current_sequence
 - seq64::seqmenu, [468](#)
- delete_lash_driver
 - seq64, [70](#)
- delete_sequence
 - seq64::perform, [350](#)
- delta_time_us_to_ticks
 - seq64, [61](#)
- device_ignore
 - seq64::rc_settings, [410](#)
- device_ignore_num
 - seq64::rc_settings, [410](#), [411](#)
- divisions
 - seq64::midi_measures, [279](#)
- do_action
 - seq64::seqedit, [439](#)
- done
 - seq64::midi_container, [271](#)
 - seq64::midi_list, [277](#)
 - seq64::midi_vector, [287](#)
- double_ticks_from_ppqn
 - seq64, [63](#)
- draw_all
 - seq64::perfroll, [389](#)
- draw_area
 - seq64::seqkeys, [460](#)
- draw_background
 - seq64::perftime, [401](#)
 - seq64::seqevent, [450](#)
- draw_background_on
 - seq64::perfroll, [390](#)
- draw_background_on_pixmap
 - seq64::seqroll, [479](#)
- draw_drawable
 - seq64::gui_drawingarea_gtk2, [180](#)
- draw_drawable_row
 - seq64::perfroll, [390](#)
- draw_event
 - seq64::eventslots, [150](#)
- draw_events
 - seq64::eventslots, [150](#)
- draw_events_on
 - seq64::seqdata, [424](#)
 - seq64::seqevent, [451](#)
 - seq64::seqroll, [483](#)
- draw_events_on_pixmap
 - seq64::seqdata, [425](#)
 - seq64::seqevent, [450](#)
 - seq64::seqroll, [479](#)
- draw_key
 - seq64::seqkeys, [461](#)
- draw_line
 - seq64::gui_drawingarea_gtk2, [175](#), [176](#)
- draw_line_on_pixmap
 - seq64::gui_drawingarea_gtk2, [175](#)
- draw_line_on_window
 - seq64::seqdata, [424](#)
- draw_marker_on_sequence
 - seq64::mainwid, [242](#)
- draw_normal_rectangle_on_pixmap
 - seq64::gui_drawingarea_gtk2, [180](#)
- draw_pixmap_on_window
 - seq64::mainwid, [242](#)
 - seq64::perftime, [403](#)
 - seq64::seqdata, [425](#)
 - seq64::seqevent, [450](#)
 - seq64::seqtime, [495](#)
- draw_progress
 - seq64::perfroll, [389](#)
- draw_progress_on_window
 - seq64::perftime, [401](#)
 - seq64::seqroll, [480](#)
 - seq64::seqtime, [495](#)
- draw_rectangle
 - seq64::gui_drawingarea_gtk2, [177–179](#)
- draw_rectangle_on_pixmap
 - seq64::gui_drawingarea_gtk2, [179](#)
- draw_selection_on_window
 - seq64::seqevent, [450](#)
 - seq64::seqroll, [479](#)
- draw_sequence
 - seq64::perfnames, [332](#)
- draw_sequence_on
 - seq64::perfroll, [390](#)
- draw_sequence_on_pixmap
 - seq64::mainwid, [243](#)
- draw_sequence_pixmap_on_window
 - seq64::mainwid, [243](#)
- draw_sequences
 - seq64::perfedit, [325](#)
 - seq64::perfnames, [332](#)
- draw_sequences_on_pixmap
 - seq64::mainwid, [243](#)
- draw_type
 - seq64, [53](#)
- dref
 - seq64::event_list, [128](#)
- drop_action
 - seq64::seqroll, [487](#)
- drop_event
 - seq64::seqevent, [451](#)
- drop_x
 - seq64::gui_drawingarea_gtk2, [174](#)
- drop_y
 - seq64::gui_drawingarea_gtk2, [174](#)
- dump_setting_summary
 - seq64::userfile, [579](#)
- dump_summary
 - seq64::user_settings, [569](#)
- e_action_draw
 - seq64, [54](#)
- e_action_grow
 - seq64, [54](#)
- e_action_select

- seq64, [54](#)
- e_clock_mod
 - seq64, [53](#)
- e_clock_off
 - seq64, [53](#)
- e_clock_pos
 - seq64, [53](#)
- e_deselect
 - seq64::sequence, [506](#)
- e_fruity_interaction
 - seq64, [53](#)
- e_is_selected
 - seq64::sequence, [506](#)
- e_jack_connect
 - seq64::options, [313](#)
- e_jack_disconnect
 - seq64::options, [313](#)
- e_jack_master
 - seq64::options, [313](#)
- e_jack_master_cond
 - seq64::options, [313](#)
- e_jack_start_mode_live
 - seq64::options, [313](#)
- e_jack_start_mode_song
 - seq64::options, [313](#)
- e_jack_transport
 - seq64::options, [313](#)
- e_number_of_interactions
 - seq64, [53](#)
- e_remove_one
 - seq64::sequence, [506](#)
- e_select
 - seq64::sequence, [506](#)
- e_select_one
 - seq64::sequence, [506](#)
- e_seq24_interaction
 - seq64, [53](#)
- e_toggle_selection
 - seq64::sequence, [506](#)
- e_would_select
 - seq64::sequence, [506](#)
- EVENT_AFTERTOUCH
 - seq64, [73](#)
- EVENT_ANY
 - seq64, [73](#)
- EVENT_CHANNEL_PRESSURE
 - seq64, [73](#)
- EVENT_CLEAR_CHAN_MASK
 - seq64, [75](#)
- EVENT_CONTROL_CHANGE
 - seq64, [73](#)
- EVENT_GET_CHAN_MASK
 - seq64, [74](#)
- EVENT_MIDI_ACTIVE_SENS
 - seq64, [74](#)
- EVENT_MIDI_CLOCK
 - seq64, [74](#)
- EVENT_MIDI_CONTINUE
 - seq64, [74](#)
- EVENT_MIDI_META
 - seq64, [74](#)
- EVENT_MIDI_QUARTER_FRAME
 - seq64, [73](#)
- EVENT_MIDI_RESET
 - seq64, [74](#)
- EVENT_MIDI_SONG_F4
 - seq64, [74](#)
- EVENT_MIDI_SONG_F5
 - seq64, [74](#)
- EVENT_MIDI_SONG_F9
 - seq64, [74](#)
- EVENT_MIDI_SONG_FD
 - seq64, [74](#)
- EVENT_MIDI_SONG_POS
 - seq64, [74](#)
- EVENT_MIDI_SONG_SELECT
 - seq64, [74](#)
- EVENT_MIDI_START
 - seq64, [74](#)
- EVENT_MIDI_STOP
 - seq64, [74](#)
- EVENT_MIDI_SYSEX_END
 - seq64, [74](#)
- EVENT_MIDI_SYSEX
 - seq64, [73](#)
- EVENT_MIDI_TUNE_SELECT
 - seq64, [74](#)
- EVENT_NOTE_OFF
 - seq64, [73](#)
- EVENT_NOTE_ON
 - seq64, [73](#)
- EVENT_NULL_CHANNEL
 - seq64, [74](#)
- EVENT_PITCH_WHEEL
 - seq64, [73](#)
- EVENT_PROGRAM_CHANGE
 - seq64, [73](#)
- EVENT_STATUS_BIT
 - seq64, [73](#)
- EVENT_SYSEX_CONTINUE
 - seq64, [74](#)
- EVENT_SYSEX_END
 - seq64, [74](#)
- EVENT_SYSEX
 - seq64, [74](#)
- edit_callback_notepad
 - seq64::mainwnd, [254](#)
- editable_event
 - seq64::editable_event, [98](#)
- editable_events
 - seq64::editable_events, [105](#), [106](#)
 - seq64::event_list, [130](#)
- empty
 - seq64::event_list, [127](#)
- end
 - seq64::editable_events, [107](#)

- seq64::event_list, 126
- enqueue_draw
 - seq64::eventedit, 136
 - seq64::eventslots, 150
 - seq64::perfedit, 323
 - seq64::perfnames, 332
 - seq64::perfroll, 390
 - seq64::perftime, 401
- enregister
 - seq64::perform, 348
- enregister_peer
 - seq64::perfedit, 324
- enregister_perfedit
 - seq64::mainwnd, 256
- errdump
 - seq64::midifile, 307
- error_is_fatal
 - seq64::midifile, 299
- error_message
 - seq64::jack_assistant, 199
 - seq64::midifile, 299
 - seq64::optionsfile, 318
- event
 - seq64::event, 113
- event_count
 - seq64::eventslots, 146
 - seq64::sequence, 507
- event_edit
 - seq64::keys_perform, 218
- event_key
 - seq64::event_list::event_key, 122, 123
- event_list
 - seq64::event_list, 126
- event_name
 - seq64::editable_event::name_value_t, 311
- event_value
 - seq64::editable_event::name_value_t, 311
- EventStack
 - seq64::sequence, 506
- eventedit
 - seq64::eventedit, 134
 - seq64::eventslots, 153
- Events
 - seq64::editable_events, 105
 - seq64::event_list, 126
- events
 - seq64::editable_events, 107
 - seq64::event_list, 130
 - seq64::keybindentry, 207
 - seq64::sequence, 507
- EventsPair
 - seq64::editable_events, 105
 - seq64::event_list, 126
- eventslots
 - seq64::editable_events, 108
 - seq64::eventedit, 140
 - seq64::eventslots, 146
- expand
 - seq64::perfedit, 325
 - seq64::perform, 352
- extract_timing_numbers
 - seq64, 54
- fg_color
 - seq64::gui_palette_gtk2, 186
- file_access
 - seq64, 66
- file_accessible
 - seq64, 66
- file_executable
 - seq64, 67
- file_exists
 - seq64, 66
- file_exit
 - seq64::mainwnd, 256
- file_import_dialog
 - seq64::mainwnd, 254
- file_is_directory
 - seq64, 67
- file_new
 - seq64::mainwnd, 256
- file_open
 - seq64::mainwnd, 256
- file_readable
 - seq64, 66
- file_save
 - seq64::mainwnd, 256
- file_save_as
 - seq64::mainwnd, 256
- file_writable
 - seq64, 66
- filename
 - seq64::rc_settings, 411
- fill
 - seq64::midi_container, 271
- fill_background_pixmap
 - seq64::perfroll, 388
- fill_background_window
 - seq64::mainwid, 242
- fill_container
 - seq64::sequence, 531
- fill_top_bar
 - seq64::seqedit, 438
- finish
 - seq64::perform, 350
- flush
 - seq64::mastermidibus, 264
 - seq64::midibus, 293
- follow_progress
 - seq64::perfroll, 389
 - seq64::seqroll, 481
- font
 - seq64::font, 156
- font_render
 - seq64, 72
- force_draw
 - seq64::gui_drawingarea_gtk2, 174

- seq64::seqevent, 451
- seq64::seqkeys, 460
- seq64::seqroll, 481
- format_timestamp
 - seq64::editable_event, 101
- FruityPerfInput
 - seq64::FruityPerfInput, 160
 - seq64::perroll, 394
- FruitySeqEventInput
 - seq64::FruitySeqEventInput, 163
 - seq64::seqevent, 455
- FruitySeqRollInput
 - seq64::FruitySeqRollInput, 165
 - seq64::seqroll, 490
- g_rc_settings
 - seq64, 80
- g_user_settings
 - seq64, 80
- get
 - seq64::midi_container, 272
 - seq64::midi_list, 277
 - seq64::midi_vector, 288
- get_32nds_per_quarter
 - seq64::sequence, 508
- get_alsa_seq
 - seq64::mastermidibus, 263
- get_beat_width
 - seq64::jack_assistant, 194
 - seq64::perform, 348
 - seq64::sequence, 508
- get_beats_per_bar
 - seq64::perform, 348
 - seq64::sequence, 508
- get_beats_per_measure
 - seq64::jack_assistant, 195
- get_beats_per_minute
 - seq64::jack_assistant, 195
 - seq64::mastermidibus, 263
 - seq64::perform, 359
- get_channel
 - seq64::event, 114
- get_client
 - seq64::midibus, 293
- get_clipboard_box
 - seq64::sequence, 522
- get_clock
 - seq64::mastermidibus, 267
 - seq64::midibus, 293
- get_clock_mod
 - seq64::midibus, 293
- get_current_sequence
 - seq64::seqmenu, 468
- get_data
 - seq64::event, 118
- get_editing
 - seq64::sequence, 509
- get_group_mute_state
 - seq64::perform, 363
- get_id
 - seq64::midibus, 292
- get_input
 - seq64::mastermidibus, 267
 - seq64::midibus, 293
- get_jack_client_info
 - seq64::jack_assistant, 200
- get_jack_pos
 - seq64::jack_assistant, 198
- get_jack_tick
 - seq64::jack_assistant, 198
 - seq64::perform, 351
- get_key_events
 - seq64::keys_perform, 219
 - seq64::perform, 364
- get_key_events_rev
 - seq64::keys_perform, 219
 - seq64::perform, 364
- get_key_groups
 - seq64::keys_perform, 219
 - seq64::perform, 364
- get_key_groups_rev
 - seq64::keys_perform, 219
 - seq64::perform, 364
- get_keys
 - seq64::keys_perform, 215
- get_last_tick
 - seq64::sequence, 510
- get_left_tick
 - seq64::perform, 351
- get_length
 - seq64::sequence, 509
- get_linked
 - seq64::event, 119
- get_max_trigger
 - seq64::perform, 352
 - seq64::sequence, 518
- get_maximum
 - seq64::triggers, 547
- get_measures
 - seq64::seqedit, 437
 - seq64::sequence, 508
- get_midi_bus
 - seq64::sequence, 518
- get_midi_channel
 - seq64::sequence, 512
- get_midi_event
 - seq64::mastermidibus, 265
- get_midi_in_bus_name
 - seq64::mastermidibus, 264
- get_midi_out_bus_name
 - seq64::mastermidibus, 264
- get_minmax_note_events
 - seq64::sequence, 530
- get_name
 - seq64::midibus, 292
 - seq64::sequence, 509
- get_next_event

- seq64::sequence, 530, 531
- get_next_note_event
 - seq64::sequence, 529
- get_next_trigger
 - seq64::sequence, 531
- get_note
 - seq64::event, 119
- get_note_velocity
 - seq64::event, 119
- get_num_in_buses
 - seq64::mastermidibus, 263
- get_num_out_buses
 - seq64::mastermidibus, 263
- get_num_selected_events
 - seq64::sequence, 520
- get_num_selected_notes
 - seq64::sequence, 520
- get_offset
 - seq64::perform, 364
- get_playing
 - seq64::sequence, 510
- get_playing_screenset
 - seq64::perform, 355
- get_port
 - seq64::midibus, 293
- get_ppqn
 - seq64::jack_assistant, 194
 - seq64::mastermidibus, 264
 - seq64::sequence, 508
- get_quantized_rec
 - seq64::sequence, 511
- get_queued
 - seq64::sequence, 511
- get_queued_tick
 - seq64::sequence, 511
- get_raise
 - seq64::sequence, 509
- get_rank
 - seq64::event, 120
- get_recording
 - seq64::sequence, 511
- get_right_tick
 - seq64::perform, 351
- get_screen_set_notepad
 - seq64::perform, 353
- get_screenset
 - seq64::perform, 354
- get_selected_box
 - seq64::seqroll, 483
 - seq64::sequence, 522
- get_selected_end
 - seq64::triggers, 547
- get_selected_start
 - seq64::triggers, 547
- get_sequence
 - seq64::mastermidibus, 266
 - seq64::perform, 358
 - seq64::seqmenu, 468
- get_song_mute
 - seq64::sequence, 509
- get_state
 - seq64::triggers, 545
- get_status
 - seq64::event, 117
- get_sysex
 - seq64::event, 118
- get_sysex_size
 - seq64::event, 118
- get_thru
 - seq64::sequence, 511
- get_tick
 - seq64::perform, 350
- get_timestamp
 - seq64::event, 114
- get_trigger_offset
 - seq64::sequence, 518
- get_trigger_state
 - seq64::sequence, 515
- global_seq_feature
 - seq64::user_settings, 567, 568
- gmute_tracks
 - seq64::user_settings, 567
- green
 - seq64::gui_palette_gtk2, 186
- grey
 - seq64::gui_palette_gtk2, 186
- grid_brackets
 - seq64::user_settings, 567, 568
- grid_is_black
 - seq64::user_settings, 567
- grid_is_normal
 - seq64::user_settings, 567
- grid_is_white
 - seq64::user_settings, 567
- grid_style
 - seq64::user_settings, 566, 568
- grid_style_black
 - seq64::user_settings, 565
- grid_style_max
 - seq64::user_settings, 565
- grid_style_normal
 - seq64::user_settings, 565
- grid_style_white
 - seq64::user_settings, 565
- group_learn
 - seq64::keys_perform, 217
- group_off
 - seq64::keys_perform, 217
- group_on
 - seq64::keys_perform, 217
- groups
 - seq64::keybindentry, 207
- grow
 - seq64::perfedit, 324
 - seq64::triggers, 545
- grow_selected

- seq64::sequence, 527
- grow_selected_notes
 - seq64::seqroll, 484
- grow_trigger
 - seq64::sequence, 514
- growing
 - seq64::seqroll, 487
- gs_mainwid_pointer
 - seq64, 80
- gs_perfedit_pointer_0
 - seq64, 81
- gs_perfedit_pointer_1
 - seq64, 81
- Gtk, 41
- gtk_drawarea_init
 - seq64::gui_drawingarea_gtk2, 181
- gui
 - seq64::perform, 348
- gui_assistant
 - seq64::gui_assistant, 168
- gui_assistant_gtk2
 - seq64::gui_assistant_gtk2, 170
- gui_drawingarea_gtk2
 - seq64::gui_drawingarea_gtk2, 174
- gui_palette_gtk2
 - seq64::gui_palette_gtk2, 185
- gui_window_gtk2
 - seq64::gui_window_gtk2, 189
- handle_cancel
 - seq64::eventedit, 138
- handle_close
 - seq64::eventedit, 138
 - seq64::seqedit, 440
- handle_config
 - seq64::lash, 233
- handle_delete
 - seq64::eventedit, 138
- handle_event
 - seq64::lash, 233
- handle_insert
 - seq64::eventedit, 138
- handle_midi_control
 - seq64::perform, 353
- handle_modify
 - seq64::eventedit, 138
- handle_motion_key
 - seq64::Seq24PerfInput, 417
- handle_save
 - seq64::eventedit, 138
- handle_signal
 - seq64::mainwnd, 254
- height
 - seq64::gui_drawingarea_gtk2::rect, 415
 - seq64::rect, 414
- help_check
 - seq64, 64
- highlight
 - seq64::perform, 368
- home_config_directory
 - seq64::rc_settings, 412
- horizontal_adjust
 - seq64::perfroll, 391
 - seq64::seqedit, 436
 - seq64::seqroll, 481
- horizontal_set
 - seq64::perfroll, 392
 - seq64::seqedit, 436
- idle_progress
 - seq64::maintime, 236
 - seq64::perftime, 403
 - seq64::seqroll, 483
 - seq64::seqtime, 496
- idle_redraw
 - seq64::seqdata, 423
 - seq64::seqevent, 451
 - seq64::seqroll, 483
- in_range
 - seq64::midi_control, 275
- increment
 - seq64::midi_splitter, 281
- increment_beats_per_minute
 - seq64::perform, 368
- increment_bottom
 - seq64::eventslots, 152
- increment_current
 - seq64::eventslots, 152
- increment_data1
 - seq64::event, 118
- increment_data2
 - seq64::event, 118
- increment_offset
 - seq64::trigger, 539
- increment_screenset
 - seq64::perform, 368
- increment_selected
 - seq64::sequence, 526
- increment_size
 - seq64::perfroll, 389
 - seq64::perftime, 401
- increment_tick_end
 - seq64::trigger, 539
- increment_tick_start
 - seq64::trigger, 539
- increment_top
 - seq64::eventslots, 152
- info_message
 - seq64::jack_assistant, 198
- init
 - seq64::font, 156
 - seq64::jack_assistant, 195
 - seq64::lash, 232
 - seq64::mastermidibus, 263
- init_before_show
 - seq64::perfedit, 323
 - seq64::perfroll, 388
- init_clock

- seq64::mastermidibus, 265
- seq64::midibus, 292
- init_in
 - seq64::midibus, 291
- init_in_sub
 - seq64::midibus, 291
- init_jack
 - seq64::perform, 371
- init_out
 - seq64::midibus, 291
- init_out_sub
 - seq64::midibus, 291
- initialize
 - seq64::midi_splitter, 281
- inner_start
 - seq64::perform, 374
- inner_stop
 - seq64::perform, 374
- input_callback
 - seq64::options, 313
- input_func
 - seq64::perform, 363
- input_thread_func
 - seq64, 71
- insert_event
 - seq64::eventslots, 147
- install_sequence
 - seq64::perform, 374
- install_signal_handlers
 - seq64::mainwnd, 257
- instrument
 - seq64::user_instrument_t, 553
 - seq64::user_midi_bus, 556
 - seq64::user_midi_bus_t, 557
 - seq64::user_settings, 566
- instrument_controller_active
 - seq64::user_settings, 566
- instrument_controller_name
 - seq64::user_settings, 566
- instrument_count
 - seq64::user_settings, 566
- instrument_name
 - seq64::user_settings, 566
- InstrumentConstIterator
 - seq64::user_settings, 565
- InstrumentIterator
 - seq64::user_settings, 565
- Instruments
 - seq64::user_settings, 565
- interaction_method
 - seq64::rc_settings, 410, 411
- interaction_method_t
 - seq64, 53
- intersect
 - seq64::triggers, 546
- intersect_events
 - seq64::sequence, 516
- intersect_notes
 - seq64::sequence, 516
- intersect_triggers
 - seq64::sequence, 515
- inverse_active
 - seq64::midi_control, 274
- is
 - seq64::keystroke, 230
- is_active
 - seq64::perform, 358
- is_adding
 - seq64::Seq24PerfInput, 417
- is_black_key
 - seq64::seqkeys, 461
- is_channel_msg
 - seq64::event, 115
- is_control_status
 - seq64::perform, 347
- is_current_seq_active
 - seq64::seqmenu, 468
- is_current_seq_in_edit
 - seq64::seqmenu, 468
- is_delete
 - seq64::keystroke, 230
- is_desired_cc_or_not_cc
 - seq64::event, 116
- is_dirty_edit
 - seq64::perform, 357
 - seq64::sequence, 511
- is_dirty_main
 - seq64::perform, 357
 - seq64::sequence, 511
- is_dirty_names
 - seq64::perform, 358
 - seq64::sequence, 512
- is_dirty_perf
 - seq64::perform, 357
 - seq64::sequence, 512
- is_dumping
 - seq64::mastermidibus, 266
- is_edit_sequence
 - seq64::perform, 347
 - seq64::seqmenu, 468
- is_group_learning
 - seq64::perform, 356
- is_jack_running
 - seq64::perform, 348
- is_left
 - seq64::click, 88
- is_letter
 - seq64::keystroke, 230
- is_linked
 - seq64::event, 119
- is_marked
 - seq64::event, 119
- is_master
 - seq64::jack_assistant, 194
- is_middle
 - seq64::click, 88

- is_midi_control_valid
 - seq64::perform, 372
- is_modified
 - seq64::event_list, 127
 - seq64::perform, 347, 372
 - seq64::seqmenu, 468
- is_more_input
 - seq64::mastermidibus, 265
- is_mseq_valid
 - seq64::perform, 373
- is_note
 - seq64::event, 120
- is_note_msg
 - seq64::event, 115
- is_note_off
 - seq64::event, 120
- is_note_on
 - seq64::event, 120
- is_one_byte_msg
 - seq64::event, 115
- is_painted
 - seq64::event, 119
- is_pattern_playing
 - seq64::rc_settings, 410
- is_pausable
 - seq64::perform, 348
- is_paused
 - seq64::perform, 348
- is_press
 - seq64::click, 88
 - seq64::keystroke, 230
- is_realized
 - seq64::gui_window_gtk2, 190
- is_right
 - seq64::click, 88
- is_running
 - seq64::jack_assistant, 194
 - seq64::perform, 348
- is_save
 - seq64::mainwnd, 257
- is_screenset_valid
 - seq64::perform, 372
- is_selected
 - seq64::event, 119
- is_seq_valid
 - seq64::perform, 373
- is_sequence_in_edit
 - seq64::perform, 350
- is_smf_0
 - seq64::perform, 369
 - seq64::sequence, 512
- is_sysex_special_id
 - seq64::midifile, 308
- is_two_byte_msg
 - seq64::event, 115
- is_valid
 - seq64::user_instrument, 551
 - seq64::user_midi_bus, 555
- iterator
 - seq64::editable_events, 105
 - seq64::event_list, 126
- jack_assistant
 - seq64::jack_assistant, 194
 - seq64::perform, 376
- jack_idle_connect
 - seq64::gui_assistant, 168
 - seq64::gui_assistant_gtk2, 170
- jack_process_callback
 - seq64, 69
 - seq64::jack_assistant, 202
- jack_session_callback
 - seq64, 69
 - seq64::jack_assistant, 203
- jack_session_uuid
 - seq64::rc_settings, 411
- jack_shutdown_callback
 - seq64, 68
 - seq64::jack_assistant, 202
- jack_start_mode
 - seq64::rc_settings, 410
- jack_sync_callback
 - seq64, 68
 - seq64::jack_assistant, 202
 - seq64::perform, 376
- jack_timebase_callback
 - seq64, 68
 - seq64::jack_assistant, 203
- jf_bit
 - seq64::jack_status_pair_t, 206
- jf_meaning
 - seq64::jack_status_pair_t, 206
- js_clock_tick
 - seq64::jack_scratchpad, 206
- js_current_tick
 - seq64::jack_scratchpad, 206
- js_dumping
 - seq64::jack_scratchpad, 206
- js_init_clock
 - seq64::jack_scratchpad, 206
- js_jack_stopped
 - seq64::jack_scratchpad, 206
- js_looping
 - seq64::jack_scratchpad, 206
- js_playback_mode
 - seq64::jack_scratchpad, 206
- js_ticks_converted_last
 - seq64::jack_scratchpad, 206
- js_total_tick
 - seq64::jack_scratchpad, 206
- keep_queue
 - seq64::keys_perform, 215, 216
- Key
 - seq64::editable_events, 105
- key
 - seq64::keystroke, 230

- key_name
 - seq64::keys_perform, 219
 - seq64::keys_perform_gtk2, 226
 - seq64::perform, 364
- key_press_event
 - seq64::perftime, 404
- keybindentry
 - seq64::keybindentry, 207
 - seq64::perform, 376
- keys
 - seq64::gui_assistant, 168
 - seq64::perform, 348
- keys_perform
 - seq64::keys_perform, 214
- keys_perform_gtk2
 - seq64::keys_perform_gtk2, 226
- keystroke
 - seq64::keystroke, 229
- keyval_name
 - seq64, 69
- keyval_normalize
 - seq64, 70
- kpt_bpm_dn
 - seq64::keys_perform_transfer, 227
- kpt_bpm_up
 - seq64::keys_perform_transfer, 227
- kpt_event_edit
 - seq64::keys_perform_transfer, 228
- kpt_group_learn
 - seq64::keys_perform_transfer, 227
- kpt_group_off
 - seq64::keys_perform_transfer, 227
- kpt_group_on
 - seq64::keys_perform_transfer, 227
- kpt_keep_queue
 - seq64::keys_perform_transfer, 227
- kpt_pattern_edit
 - seq64::keys_perform_transfer, 228
- kpt_pause
 - seq64::keys_perform_transfer, 228
- kpt_queue
 - seq64::keys_perform_transfer, 227
- kpt_replace
 - seq64::keys_perform_transfer, 227
- kpt_screenset_dn
 - seq64::keys_perform_transfer, 227
- kpt_screenset_up
 - seq64::keys_perform_transfer, 227
- kpt_set_playing_screenset
 - seq64::keys_perform_transfer, 227
- kpt_show_ui_sequence_key
 - seq64::keys_perform_transfer, 228
- kpt_show_ui_sequence_number
 - seq64::keys_perform_transfer, 228
- kpt_snapshot_1
 - seq64::keys_perform_transfer, 227
- kpt_snapshot_2
 - seq64::keys_perform_transfer, 227
- kpt_start
 - seq64::keys_perform_transfer, 228
- kpt_stop
 - seq64::keys_perform_transfer, 228
- lash
 - seq64::lash, 232
- lash_driver
 - seq64, 70
- lash_support
 - seq64::rc_settings, 409
- lash_support_callback
 - seq64::options, 314
- lash_timeout_connect
 - seq64::gui_assistant, 168
 - seq64::gui_assistant_gtk2, 170
- last_used_dir
 - seq64::rc_settings, 411
- launch
 - seq64::perform, 349
- launch_input_thread
 - seq64::perform, 371
- launch_output_thread
 - seq64::perform, 371
- learn_toggle
 - seq64::mainwnd, 256
 - seq64::perform, 368
- legacy_format
 - seq64::rc_settings, 409
- light_grey
 - seq64::gui_palette_gtk2, 186
- line_after
 - seq64::configfile, 92
- line_color
 - seq64::gui_palette_gtk2, 185
- line_count
 - seq64::eventslots, 146
- line_increment
 - seq64::eventslots, 146
- line_maximum
 - seq64::eventslots, 146
- link
 - seq64::event, 118
- link_new
 - seq64::event_list, 128
 - seq64::sequence, 528
- List
 - seq64::triggers, 542
- load_events
 - seq64::editable_events, 106
 - seq64::eventslots, 146
- location
 - seq64::keybindentry, 207
- lock
 - seq64::mutex, 311
- log2_time_sig_value
 - seq64, 59
- log_main_sequence
 - seq64::midi_splitter, 281

- long_options
 - seq64, 79
- lookup_keyevent_key
 - seq64::keys_perform, 219
 - seq64::perform, 365
- lookup_keyevent_seq
 - seq64::keys_perform, 219
 - seq64::perform, 365
- lookup_keygroup_group
 - seq64::keys_perform, 219
 - seq64::perform, 365
- lookup_keygroup_key
 - seq64::keys_perform, 219
 - seq64::perform, 365
- m_32nds_per_quarter
 - seq64::sequence, 537
- m_4bar_offset
 - seq64::perfroll, 395
 - seq64::perftime, 404
- m_active
 - seq64::midi_control, 275
- m_adding
 - seq64::Seq24PerfInput, 418
 - seq64::Seq24SeqEventInput, 419
 - seq64::seqroll, 491
- m_adding_pressed
 - seq64::AbstractPerfInput, 84
- m_adjust_bpm
 - seq64::mainwnd, 259
- m_adjust_load_offset
 - seq64::mainwnd, 259
- m_adjust_ss
 - seq64::mainwnd, 259
- m_allow_mod4_mode
 - seq64::rc_settings, 413
- m_allow_two_perfedit
 - seq64::user_settings, 574
- m_alsa_seq
 - seq64::mastermidibus, 268
- m_auto_option_save
 - seq64::rc_settings, 413
- m_b_on_c_pixmap
 - seq64::font, 158
- m_b_on_y_pixmap
 - seq64::font, 158
- m_background
 - seq64::gui_drawingarea_gtk2, 181
- m_background_sequence
 - seq64::seqroll, 492
 - seq64::sequence, 537
- m_background_x
 - seq64::perfroll, 395
- m_bar_width
 - seq64::maintime, 236
- m_beat_length
 - seq64::perfroll, 395
- m_beat_width
 - seq64::jack_assistant, 205
- seq64::maintime, 236
- seq64::midi_timing, 285
- seq64::perform, 379
- m_beats
 - seq64::midi_measures, 279
- m_beats_per_bar
 - seq64::perform, 379
- m_beats_per_measure
 - seq64::jack_assistant, 205
 - seq64::midi_timing, 285
- m_beats_per_minute
 - seq64::jack_assistant, 205
 - seq64::mastermidibus, 268
 - seq64::midi_timing, 285
- m_bg_color
 - seq64::gui_palette_gtk2, 187
- m_bgsequence
 - seq64::seqedit, 442
- m_black
 - seq64::gui_palette_gtk2, 186
- m_black_pixmap
 - seq64::font, 157
- m_blue
 - seq64::gui_palette_gtk2, 187
- m_bottbox
 - seq64::eventedit, 140
- m_bottom_iterator
 - seq64::eventslots, 154
- m_box_height
 - seq64::maintime, 237
- m_box_less_pill
 - seq64::maintime, 237
- m_box_width
 - seq64::maintime, 237
- m_bpm
 - seq64::perfedit, 328
- m_bus
 - seq64::sequence, 535
- m_bus_announce
 - seq64::mastermidibus, 268
- m_buses_in
 - seq64::mastermidibus, 268
- m_buses_in_active
 - seq64::mastermidibus, 268
- m_buses_in_init
 - seq64::mastermidibus, 268
- m_buses_out
 - seq64::mastermidibus, 268
- m_buses_out_active
 - seq64::mastermidibus, 268
- m_buses_out_init
 - seq64::mastermidibus, 268
- m_button
 - seq64::click, 88
- m_button_bpm
 - seq64::perfedit, 328
 - seq64::seqedit, 445
- m_button_bus

- seq64::seqedit, [444](#)
- m_button_bw
 - seq64::perfedit, [328](#)
 - seq64::seqedit, [445](#)
- m_button_cancel
 - seq64::eventedit, [140](#)
- m_button_channel
 - seq64::seqedit, [444](#)
- m_button_collapse
 - seq64::perfedit, [328](#)
- m_button_copy
 - seq64::perfedit, [328](#)
- m_button_data
 - seq64::seqedit, [445](#)
- m_button_del
 - seq64::eventedit, [140](#)
- m_button_down
 - seq64::mainwid, [247](#)
- m_button_expand
 - seq64::perfedit, [328](#)
- m_button_grow
 - seq64::perfedit, [328](#)
- m_button_ins
 - seq64::eventedit, [140](#)
- m_button_jack_connect
 - seq64::options, [314](#)
- m_button_jack_disconnect
 - seq64::options, [314](#)
- m_button_jack_master
 - seq64::options, [314](#)
- m_button_jack_master_cond
 - seq64::options, [314](#)
- m_button_jack_transport
 - seq64::options, [314](#)
- m_button_key
 - seq64::seqedit, [445](#)
- m_button_learn
 - seq64::mainwnd, [259](#)
- m_button_length
 - seq64::seqedit, [445](#)
- m_button_loop
 - seq64::perfedit, [327](#)
- m_button_modify
 - seq64::eventedit, [140](#)
- m_button_note_length
 - seq64::seqedit, [444](#)
- m_button_ok
 - seq64::options, [314](#)
- m_button_perfedit
 - seq64::mainwnd, [259](#)
- m_button_play
 - seq64::mainwnd, [259](#)
 - seq64::perfedit, [327](#)
- m_button_quantize
 - seq64::seqedit, [444](#)
- m_button_rec_vol
 - seq64::seqedit, [445](#)
- m_button_redo
 - seq64::seqedit, [444](#)
- m_button_save
 - seq64::eventedit, [140](#)
- m_button_scale
 - seq64::seqedit, [445](#)
- m_button_sequence
 - seq64::seqedit, [444](#)
- m_button_snap
 - seq64::perfedit, [327](#)
 - seq64::seqedit, [444](#)
- m_button_stop
 - seq64::mainwnd, [259](#)
 - seq64::perfedit, [327](#)
- m_button_tools
 - seq64::seqedit, [444](#)
- m_button_undo
 - seq64::perfedit, [328](#)
 - seq64::seqedit, [444](#)
- m_button_zoom
 - seq64::seqedit, [445](#)
- m_bw
 - seq64::perfedit, [328](#)
- m_c_on_b_pixmap
 - seq64::font, [158](#)
- m_call_seq_edit
 - seq64::mainwnd, [260](#)
- m_call_seq_eventedit
 - seq64::mainwnd, [260](#)
- m_category
 - seq64::editable_event, [103](#)
- m_cc
 - seq64::seqdata, [427](#)
 - seq64::seqevent, [456](#)
 - seq64::seqroll, [492](#)
- m_cell_h
 - seq64::font, [157](#)
- m_cell_w
 - seq64::font, [157](#)
- m_channel
 - seq64::event, [120](#)
- m_channel_count
 - seq64::user_midi_bus, [557](#)
- m_char_list
 - seq64::midi_list, [278](#)
 - seq64::midifile, [309](#)
- m_char_vector
 - seq64::midi_vector, [288](#)
- m_char_w
 - seq64::eventslots, [153](#)
 - seq64::perfnames, [335](#)
- m_client
 - seq64::lash, [233](#)
- m_clip_mask
 - seq64::font, [158](#)
- m_clipboard
 - seq64::seqmenu, [471](#)
 - seq64::triggers, [549](#)
- m_clock_mod

- seq64::midibus, [293](#)
- m_clock_type
 - seq64::midibus, [294](#)
- m_clocks_per_metronome
 - seq64::sequence, [537](#)
- m_cond
 - seq64::condition_var, [90](#)
- m_condition_var
 - seq64::perform, [381](#)
- m_config_directory
 - seq64::rc_settings, [414](#)
- m_config_filename
 - seq64::rc_settings, [414](#)
- m_config_filename_alt
 - seq64::rc_settings, [414](#)
- m_control_height
 - seq64::user_settings, [573](#)
- m_control_status
 - seq64::perform, [380](#)
- m_controller_count
 - seq64::user_instrument, [553](#)
- m_current_event
 - seq64::editable_events, [108](#)
- m_current_index
 - seq64::eventslots, [154](#)
- m_current_iterator
 - seq64::eventslots, [154](#)
- m_current_seq
 - seq64::seqmenu, [471](#)
- m_current_x
 - seq64::FruityPerfInput, [162](#)
 - seq64::gui_drawingarea_gtk2, [182](#)
- m_current_y
 - seq64::FruityPerfInput, [162](#)
 - seq64::gui_drawingarea_gtk2, [182](#)
- m_current_zoom
 - seq64::user_settings, [573](#)
- m_d
 - seq64::configfile, [93](#)
- m_data
 - seq64::event, [121](#)
 - seq64::midi_control, [275](#)
 - seq64::midifile, [309](#)
- m_dest_addr_client
 - seq64::midibus, [294](#)
- m_dest_addr_port
 - seq64::midibus, [294](#)
- m_device_ignore
 - seq64::rc_settings, [413](#)
- m_device_ignore_num
 - seq64::rc_settings, [414](#)
- m_dirty_edit
 - seq64::sequence, [535](#)
- m_dirty_main
 - seq64::sequence, [535](#)
- m_dirty_names
 - seq64::sequence, [536](#)
- m_dirty_perf
 - seq64::sequence, [536](#)
- m_disable_reported
 - seq64::midifile, [308](#)
- m_divisions
 - seq64::midi_measures, [279](#)
- m_divs_per_beat
 - seq64::perfroll, [395](#)
- m_dk_cyan
 - seq64::gui_palette_gtk2, [187](#)
- m_dk_grey
 - seq64::gui_palette_gtk2, [186](#)
- m_dk_orange
 - seq64::gui_palette_gtk2, [187](#)
- m_drag_paste_start_pos
 - seq64::FruitySeqRollInput, [167](#)
- m_dragging
 - seq64::seqdata, [428](#)
- m_drawing_background_seq
 - seq64::seqroll, [492](#)
- m_drop_sequence
 - seq64::perfroll, [396](#)
- m_drop_tick
 - seq64::perfroll, [396](#)
- m_drop_tick_trigger_offset
 - seq64::perfroll, [396](#)
- m_drop_x
 - seq64::gui_drawingarea_gtk2, [182](#)
- m_drop_y
 - seq64::gui_drawingarea_gtk2, [182](#)
- m_dumping_input
 - seq64::mastermidibus, [269](#)
- m_edit_sequence
 - seq64::perform, [381](#)
- m_editbox
 - seq64::eventedit, [140](#)
- m_editing
 - seq64::sequence, [536](#)
- m_editing_cc
 - seq64::seqedit, [445](#)
- m_editing_status
 - seq64::seqedit, [445](#)
- m_effective_tick
 - seq64::Seq24PerfInput, [418](#)
- m_entry_bpm
 - seq64::perfedit, [328](#)
 - seq64::seqedit, [445](#)
- m_entry_bus
 - seq64::seqedit, [444](#)
- m_entry_bw
 - seq64::perfedit, [328](#)
 - seq64::seqedit, [445](#)
- m_entry_channel
 - seq64::seqedit, [444](#)
- m_entry_data
 - seq64::seqedit, [445](#)
- m_entry_ev_data_0
 - seq64::eventedit, [141](#)
- m_entry_ev_data_1

- seq64::eventedit, 141
- m_entry_ev_name
 - seq64::eventedit, 141
- m_entry_ev_timestamp
 - seq64::eventedit, 141
- m_entry_key
 - seq64::seqedit, 445
- m_entry_length
 - seq64::seqedit, 445
- m_entry_name
 - seq64::seqedit, 445
- m_entry_note_length
 - seq64::seqedit, 444
- m_entry_notes
 - seq64::mainwnd, 259
- m_entry_scale
 - seq64::seqedit, 445
- m_entry_sequence
 - seq64::seqedit, 444
- m_entry_snap
 - seq64::perfedit, 327
 - seq64::seqedit, 444
- m_entry_zoom
 - seq64::seqedit, 445
- m_erase_painting
 - seq64::FruitySeqRollInput, 167
- m_error_is_fatal
 - seq64::midifile, 308
- m_error_message
 - seq64::midifile, 308
- m_event_container
 - seq64::eventslots, 153
- m_event_count
 - seq64::eventslots, 154
- m_eventedit
 - seq64::seqmenu, 471
- m_events
 - seq64::editable_events, 108
 - seq64::event_list, 130
 - seq64::sequence, 534
- m_events_clipboard
 - seq64::sequence, 534
- m_events_redo
 - seq64::sequence, 535
- m_events_undo
 - seq64::sequence, 535
- m_eventslots
 - seq64::eventedit, 140
- m_fg_color
 - seq64::gui_palette_gtk2, 187
- m_file_size
 - seq64::midifile, 308
- m_filename
 - seq64::rc_settings, 414
- m_flash_height
 - seq64::maintime, 237
- m_flash_width
 - seq64::maintime, 237
- m_flash_x
 - seq64::maintime, 237
- m_font_h
 - seq64::font, 157
- m_font_w
 - seq64::font, 157
- m_foreground
 - seq64::gui_drawingarea_gtk2, 181
- m_format_timestamp
 - seq64::editable_event, 103
- m_fruity_interaction
 - seq64::perfroll, 396
 - seq64::seqevent, 455
 - seq64::seqroll, 491
- m_gc
 - seq64::gui_drawingarea_gtk2, 181
- m_global_bgsequence
 - seq64::midifile, 309
- m_global_seq_feature_save
 - seq64::user_settings, 573
- m_gmute_tracks
 - seq64::user_settings, 576
- m_green
 - seq64::gui_palette_gtk2, 187
- m_grey
 - seq64::gui_palette_gtk2, 186
- m_grid_brackets
 - seq64::user_settings, 572
- m_grid_style
 - seq64::user_settings, 571
- m_grow_direction
 - seq64::perfroll, 397
- m_growing
 - seq64::perfroll, 397
 - seq64::seqevent, 456
 - seq64::seqroll, 492
- m_gui_support
 - seq64::perform, 376
- m_h_page_increment
 - seq64::perfroll, 394
- m_h_perf_page_increment
 - seq64::user_settings, 574
- m_hadjust
 - seq64::gui_drawingarea_gtk2, 181
 - seq64::perfedit, 327
 - seq64::seqedit, 443
- m_has_link
 - seq64::event, 121
- m_have_focus
 - seq64::eventedit, 141
 - seq64::seqedit, 446
- m_hbox
 - seq64::perfedit, 328
 - seq64::seqedit, 444
- m_hbox2
 - seq64::seqedit, 444
- m_hint_key
 - seq64::seqkeys, 463

- m_hint_state
 - seq64::seqkeys, [463](#)
- m_hlbox
 - seq64::perfedit, [328](#)
- m_horizontal_adjust
 - seq64::seqroll, [491](#)
- m_hscroll
 - seq64::perfedit, [327](#)
- m_hscroll_new
 - seq64::seqedit, [443](#)
- m_htopbox
 - seq64::eventedit, [140](#)
- m_id
 - seq64::midibus, [293](#)
- m_image_play
 - seq64::mainwnd, [259](#)
 - seq64::perfedit, [327](#)
- m_in_thread
 - seq64::perform, [378](#)
- m_in_thread_launched
 - seq64::perform, [378](#)
- m_init_clock
 - seq64::mastermidibus, [268](#)
- m_init_input
 - seq64::mastermidibus, [268](#)
- m_initial_note_length
 - seq64::seqedit, [442](#)
- m_initial_snap
 - seq64::seqedit, [441](#)
- m_initial_zoom
 - seq64::seqedit, [442](#)
- m_inputting
 - seq64::midibus, [294](#)
 - seq64::perform, [378](#)
- m_instrument_def
 - seq64::user_instrument, [553](#)
- m_instruments
 - seq64::user_settings, [571](#)
- m_interaction_method
 - seq64::rc_settings, [414](#)
- m_inverse_active
 - seq64::midi_control, [275](#)
- m_is_drag_pasting
 - seq64::FruitySeqEventInput, [165](#)
 - seq64::seqroll, [492](#)
- m_is_drag_pasting_start
 - seq64::FruitySeqEventInput, [165](#)
 - seq64::seqroll, [492](#)
- m_is_lash_supported
 - seq64::lash, [233](#)
- m_is_modified
 - seq64::event_list, [130](#)
 - seq64::perform, [381](#)
- m_is_pattern_playing
 - seq64::rc_settings, [413](#)
- m_is_paused
 - seq64::perform, [380](#)
- m_is_press
 - seq64::click, [88](#)
 - seq64::keystroke, [231](#)
- m_is_realized
 - seq64::gui_window_gtk2, [191](#)
- m_is_running
 - seq64::mainwnd, [259](#)
 - seq64::perfedit, [329](#)
- m_is_valid
 - seq64::user_instrument, [553](#)
 - seq64::user_midi_bus, [556](#)
- m_iterator_draw
 - seq64::sequence, [535](#)
- m_iterator_draw_trigger
 - seq64::triggers, [549](#)
- m_iterator_play_trigger
 - seq64::triggers, [549](#)
- m_jack_asst
 - seq64::perform, [381](#)
- m_jack_client
 - seq64::jack_assistant, [204](#)
- m_jack_client_name
 - seq64::jack_assistant, [204](#)
- m_jack_client_uuid
 - seq64::jack_assistant, [204](#)
- m_jack_frame_current
 - seq64::jack_assistant, [204](#)
- m_jack_frame_last
 - seq64::jack_assistant, [204](#)
- m_jack_master
 - seq64::jack_assistant, [205](#)
- m_jack_parent
 - seq64::jack_assistant, [204](#)
- m_jack_pos
 - seq64::jack_assistant, [204](#)
- m_jack_running
 - seq64::jack_assistant, [205](#)
- m_jack_session_uuid
 - seq64::rc_settings, [414](#)
- m_jack_start_mode
 - seq64::rc_settings, [413](#)
- m_jack_tick
 - seq64::jack_assistant, [205](#)
 - seq64::perform, [379](#)
- m_jack_transport_state
 - seq64::jack_assistant, [204](#)
- m_jack_transport_state_last
 - seq64::jack_assistant, [204](#)
- m_session_ev
 - seq64::jack_assistant, [205](#)
- m_justselected_one
 - seq64::FruitySeqEventInput, [165](#)
 - seq64::seqroll, [492](#)
- m_key
 - seq64::keybindentry, [208](#)
 - seq64::keystroke, [231](#)
 - seq64::seqedit, [442](#)
 - seq64::seqkeys, [464](#)
 - seq64::seqroll, [491](#)

- m_key_bpm_dn
 - seq64::keys_perform, 223
- m_key_bpm_up
 - seq64::keys_perform, 223
- m_key_event_edit
 - seq64::keys_perform, 224
- m_key_events
 - seq64::keys_perform, 222
- m_key_events_rev
 - seq64::keys_perform, 223
- m_key_group_learn
 - seq64::keys_perform, 224
- m_key_group_off
 - seq64::keys_perform, 223
- m_key_group_on
 - seq64::keys_perform, 223
- m_key_groups
 - seq64::keys_perform, 223
- m_key_groups_rev
 - seq64::keys_perform, 223
- m_key_keep_queue
 - seq64::keys_perform, 223
- m_key_pattern_edit
 - seq64::keys_perform, 224
- m_key_pause
 - seq64::keys_perform, 224
- m_key_queue
 - seq64::keys_perform, 223
- m_key_replace
 - seq64::keys_perform, 223
- m_key_screenset_dn
 - seq64::keys_perform, 223
- m_key_screenset_up
 - seq64::keys_perform, 223
- m_key_set_playing_screenset
 - seq64::keys_perform, 223
- m_key_show_ui_sequence_key
 - seq64::keys_perform, 222
- m_key_show_ui_sequence_number
 - seq64::keys_perform, 222
- m_key_snapshot_1
 - seq64::keys_perform, 223
- m_key_snapshot_2
 - seq64::keys_perform, 223
- m_key_start
 - seq64::keys_perform, 224
- m_key_stop
 - seq64::keys_perform, 224
- m_keying
 - seq64::seqkeys, 463
- m_keying_note
 - seq64::seqkeys, 463
- m_keys_perform
 - seq64::gui_assistant, 168
- m_label_category
 - seq64::eventedit, 141
- m_label_channel
 - seq64::eventedit, 141
- m_label_ev_count
 - seq64::eventedit, 141
- m_label_modified
 - seq64::eventedit, 141
- m_label_ppqn
 - seq64::eventedit, 141
- m_label_right
 - seq64::eventedit, 141
- m_label_seq_name
 - seq64::eventedit, 140
- m_label_spacer
 - seq64::eventedit, 141
- m_label_time_fmt
 - seq64::eventedit, 141
- m_label_time_sig
 - seq64::eventedit, 140
- m_lash_args
 - seq64::lash, 233
- m_lash_support
 - seq64::rc_settings, 413
- m_last_tick
 - seq64::sequence, 536
- m_last_tick_x
 - seq64::mainwid, 247
- m_last_used_dir
 - seq64::rc_settings, 414
- m_lasttick
 - seq64::midibus, 294
- m_left_marker_tick
 - seq64::perftime, 405
- m_left_tick
 - seq64::perform, 379
- m_legacy_format
 - seq64::rc_settings, 413
- m_length
 - seq64::sequence, 536
 - seq64::triggers, 549
- m_line
 - seq64::configfile, 93
- m_line_color
 - seq64::gui_palette_gtk2, 187
- m_line_count
 - seq64::eventslots, 154
- m_line_maximum
 - seq64::eventslots, 154
- m_line_overlap
 - seq64::eventslots, 154
- m_linked
 - seq64::event, 121
- m_local_addr_client
 - seq64::midibus, 294
- m_local_addr_port
 - seq64::midibus, 294
- m_looping
 - seq64::perform, 378
- m_lt_grey
 - seq64::gui_palette_gtk2, 186
- m_main_cursor

- seq64::mainwnd, 259
- m_main_time
 - seq64::mainwnd, 258
- m_main_wid
 - seq64::mainwnd, 258
- m_mainperf
 - seq64::gui_drawingarea_gtk2, 182
 - seq64::gui_window_gtk2, 190
 - seq64::options, 314
 - seq64::seqmenu, 471
- m_mainwid_border
 - seq64::mainwid, 247
 - seq64::user_settings, 572
- m_mainwid_spacing
 - seq64::mainwid, 247
 - seq64::user_settings, 572
- m_mainwid_x
 - seq64::mainwid, 247
 - seq64::user_settings, 576
- m_mainwid_y
 - seq64::mainwid, 247
 - seq64::user_settings, 576
- m_mainwnd_cols
 - seq64::mainwid, 247
 - seq64::user_settings, 572
- m_mainwnd_rows
 - seq64::mainwid, 247
 - seq64::user_settings, 572
- m_manual_alsa_ports
 - seq64::rc_settings, 413
- m_marked
 - seq64::event, 121
- m_master_bus
 - seq64::perform, 378
- m_masterbus
 - seq64::sequence, 535
- m_max_sequence
 - seq64::user_settings, 576
- m_max_sets
 - seq64::mainwid, 247
 - seq64::perform, 380
 - seq64::user_settings, 572
- m_max_value
 - seq64::midi_control, 275
- m_maxbeats
 - seq64::sequence, 536
- m_measure_length
 - seq64::perfroll, 395
 - seq64::perftime, 405
- m_measures
 - seq64::midi_measures, 279
 - seq64::seqedit, 442
- m_menu
 - seq64::seqmenu, 471
- m_menu_bpm
 - seq64::perfedit, 328
 - seq64::seqedit, 443
- m_menu_bw
 - seq64::perfedit, 328
 - seq64::seqedit, 443
- m_menu_data
 - seq64::seqedit, 443
- m_menu_file
 - seq64::mainwnd, 258
- m_menu_help
 - seq64::mainwnd, 258
- m_menu_key
 - seq64::seqedit, 443
- m_menu_length
 - seq64::seqedit, 443
- m_menu_midibus
 - seq64::seqedit, 443
- m_menu_midich
 - seq64::seqedit, 443
- m_menu_note_length
 - seq64::seqedit, 443
- m_menu_rec_vol
 - seq64::seqedit, 443
- m_menu_scale
 - seq64::seqedit, 443
- m_menu_sequences
 - seq64::seqedit, 443
- m_menu_snap
 - seq64::perfedit, 327
 - seq64::seqedit, 443
- m_menu_tools
 - seq64::seqedit, 442
- m_menu_view
 - seq64::mainwnd, 258
- m_menu_zoom
 - seq64::seqedit, 443
- m_menubar
 - seq64::mainwnd, 258
 - seq64::seqedit, 442
- m_midi_beat_width
 - seq64::user_settings, 575
- m_midi_beats_per_measure
 - seq64::user_settings, 575
- m_midi_beats_per_minute
 - seq64::user_settings, 575
- m_midi_bus_def
 - seq64::user_midi_bus, 557
- m_midi_buses
 - seq64::user_settings, 571
- m_midi_buss_override
 - seq64::user_settings, 575
- m_midi_cc_off
 - seq64::perform, 380
- m_midi_cc_on
 - seq64::perform, 380
- m_midi_cc_toggle
 - seq64::perform, 380
- m_midi_channel
 - seq64::sequence, 535
- m_midi_parameters
 - seq64::editable_events, 108

- m_midi_ppqn
 - seq64::user_settings, 575
- m_midiclockpos
 - seq64::perform, 380
- m_midiclockrunning
 - seq64::perform, 380
- m_midiclocktick
 - seq64::perform, 380
- m_min_value
 - seq64::midi_control, 275
- m_mode_group
 - seq64::perform, 377
- m_mode_group_learn
 - seq64::perform, 377
- m_modified
 - seq64::seqmenu, 471
- m_modifier
 - seq64::click, 88
 - seq64::keystroke, 231
- m_move_delta_x
 - seq64::seqroll, 492
- m_move_delta_y
 - seq64::seqroll, 492
- m_move_snap_offset_x
 - seq64::seqevent, 456
 - seq64::seqroll, 492
- m_moving
 - seq64::mainwid, 247
 - seq64::perfroll, 397
 - seq64::seqevent, 456
 - seq64::seqroll, 492
- m_moving_init
 - seq64::seqevent, 456
 - seq64::seqroll, 492
- m_moving_seq
 - seq64::mainwid, 247
- m_musical_key
 - seq64::sequence, 537
- m_musical_scale
 - seq64::sequence, 537
- m_mute_group
 - seq64::perform, 376
- m_mute_group_selected
 - seq64::perform, 377
- m_mutex
 - seq64::mastermidibus, 269
 - seq64::midibus, 294
 - seq64::sequence, 537
- m_mutex_lock
 - seq64::mutex, 311
- m_name
 - seq64::configfile, 93
 - seq64::midibus, 294
 - seq64::midifile, 309
 - seq64::sequence, 536
- m_name_category
 - seq64::editable_event, 103
- m_name_channel
 - seq64::editable_event, 103
- m_name_data
 - seq64::editable_event, 103
- m_name_meta
 - seq64::editable_event, 103
- m_name_seqspect
 - seq64::editable_event, 103
- m_name_status
 - seq64::editable_event, 103
- m_name_timestamp
 - seq64::editable_event, 103
- m_namebox_w
 - seq64::perfnames, 336
- m_names_chars
 - seq64::perfnames, 335
- m_names_x
 - seq64::perfnames, 336
- m_names_y
 - seq64::perfnames, 336
 - seq64::perfroll, 395
- m_new_format
 - seq64::midifile, 309
- m_note_length
 - seq64::seqedit, 442
 - seq64::seqroll, 491
- m_note_off_margin
 - seq64::sequence, 537
- m_notebook
 - seq64::options, 314
- m_notes_on
 - seq64::sequence, 535
- m_notify
 - seq64::perform, 381
- m_num_in_buses
 - seq64::mastermidibus, 268
- m_num_out_buses
 - seq64::mastermidibus, 268
- m_num_poll_descriptors
 - seq64::mastermidibus, 268
- m_number_h
 - seq64::seqdata, 427
- m_number_offset_y
 - seq64::seqdata, 427
- m_number_w
 - seq64::seqdata, 427
- m_numbers
 - seq64::seqdata, 427
- m_offset
 - seq64::font, 157
 - seq64::perform, 380
 - seq64::trigger, 540
- m_old
 - seq64::seqdata, 427
 - seq64::seqevent, 455
 - seq64::seqroll, 491
- m_old_progress_ticks
 - seq64::perfroll, 395
- m_old_seq

- seq64::mainwid, [247](#)
- m_one_measure
 - seq64::perform, [379](#)
- m_options
 - seq64::mainwnd, [258](#)
- m_optsbox
 - seq64::eventedit, [140](#)
- m_orange
 - seq64::gui_palette_gtk2, [186](#)
- m_out_thread
 - seq64::perform, [378](#)
- m_out_thread_launched
 - seq64::perform, [378](#)
- m_outputing
 - seq64::perform, [378](#)
- m_padded_h
 - seq64::font, [157](#)
- m_page_factor
 - seq64::perfroll, [395](#)
- m_pager_index
 - seq64::eventslots, [154](#)
- m_painted
 - seq64::event, [121](#)
- m_painting
 - seq64::seqevent, [456](#)
 - seq64::seqroll, [492](#)
- m_parent
 - seq64::editable_event, [103](#)
 - seq64::eventslots, [153](#)
 - seq64::perfnames, [335](#)
 - seq64::perfroll, [394](#)
 - seq64::perftime, [404](#)
 - seq64::sequence, [534](#)
 - seq64::triggers, [549](#)
- m_pass_sysex
 - seq64::rc_settings, [413](#)
- m_paste
 - seq64::seqevent, [456](#)
 - seq64::seqroll, [492](#)
- m_peer_perfedit
 - seq64::perfedit, [327](#)
- m_perf
 - seq64::keybindentry, [208](#)
- m_perf_edit
 - seq64::mainwnd, [258](#)
- m_perf_edit_2
 - seq64::mainwnd, [258](#)
- m_perf_scale_x
 - seq64::perfroll, [395](#)
 - seq64::perftime, [405](#)
- m_perfnames
 - seq64::perfedit, [327](#)
- m_perform
 - seq64::lash, [233](#)
- m_perfroll
 - seq64::perfedit, [327](#)
- m_perftime
 - seq64::perfedit, [327](#)
- m_pill_width
 - seq64::maintime, [237](#)
- m_pixmap
 - seq64::font, [157](#)
 - seq64::gui_drawingarea_gtk2, [181](#)
- m_playback_mode
 - seq64::perform, [378](#)
- m_playing
 - seq64::sequence, [535](#)
- m_playing_notes
 - seq64::sequence, [535](#)
- m_playing_screen
 - seq64::perform, [377](#)
- m_playscreen_offset
 - seq64::perform, [377](#)
- m_poll_descriptors
 - seq64::mastermidibus, [269](#)
- m_pos
 - seq64::midifile, [308](#)
 - seq64::seqroll, [491](#)
- m_position_for_get
 - seq64::midi_container, [272](#)
- m_ppqn
 - seq64::jack_assistant, [205](#)
 - seq64::maintime, [237](#)
 - seq64::mainwnd, [258](#)
 - seq64::mastermidibus, [268](#)
 - seq64::midi_splitter, [283](#)
 - seq64::midi_timing, [285](#)
 - seq64::midibus, [294](#)
 - seq64::midifile, [309](#)
 - seq64::perfedit, [328](#)
 - seq64::perform, [378](#)
 - seq64::perfroll, [395](#)
 - seq64::perftime, [405](#)
 - seq64::seqedit, [442](#)
 - seq64::seqevent, [455](#)
 - seq64::seqroll, [491](#)
 - seq64::seqtime, [496](#)
 - seq64::sequence, [536](#)
 - seq64::triggers, [549](#)
- m_print_keys
 - seq64::rc_settings, [413](#)
- m_priority
 - seq64::rc_settings, [413](#)
- m_progress_bar_colored
 - seq64::user_settings, [574](#)
- m_progress_bar_thick
 - seq64::user_settings, [574](#)
- m_progress_color
 - seq64::gui_palette_gtk2, [187](#)
- m_progress_height
 - seq64::mainwid, [248](#)
- m_progress_x
 - seq64::seqroll, [492](#)
- m_quantized_rec
 - seq64::sequence, [535](#)
- m_queue

- seq64::mastermidibus, 268
- seq64::midibus, 294
- m_queued
 - seq64::sequence, 535
- m_queued_tick
 - seq64::sequence, 536
- m_raise
 - seq64::sequence, 536
- m_rank
 - seq64::event_list::event_key, 123
- m_rec_vol
 - seq64::sequence, 537
- m_recording
 - seq64::sequence, 535
- m_red
 - seq64::gui_palette_gtk2, 186
- m_redo_stack
 - seq64::triggers, 549
- m_redraw_period_ms
 - seq64::gui_window_gtk2, 191
- m_reveal_alsa_ports
 - seq64::rc_settings, 413
- m_right_marker_tick
 - seq64::perftime, 405
- m_right_tick
 - seq64::perform, 379
- m_rightbox
 - seq64::eventedit, 140
- m_roll_length_ticks
 - seq64::perfroll, 396
- m_running
 - seq64::perform, 378
- m_safety_mutex
 - seq64::automutex, 85
- m_save_user_config
 - seq64::user_settings, 577
- m_scale
 - seq64::seqedit, 442
 - seq64::seqkeys, 464
 - seq64::seqroll, 491
- m_screen_set_notepad
 - seq64::perform, 380
- m_screenset
 - seq64::mainwid, 247
 - seq64::perform, 380
- m_screenset_offset
 - seq64::mainwid, 247
- m_screenset_slots
 - seq64::mainwid, 247
- m_scroll_offset_key
 - seq64::seqkeys, 463
 - seq64::seqroll, 492
- m_scroll_offset_ticks
 - seq64::seqdata, 427
 - seq64::seqevent, 455
 - seq64::seqroll, 492
 - seq64::seqtime, 496
- m_scroll_offset_x
 - seq64::seqdata, 427
 - seq64::seqevent, 455
 - seq64::seqroll, 492
 - seq64::seqtime, 496
- m_scroll_offset_y
 - seq64::seqkeys, 463
 - seq64::seqroll, 492
- m_selected
 - seq64::event, 121
 - seq64::seqevent, 455
 - seq64::seqroll, 491
 - seq64::trigger, 540
- m_selecting
 - seq64::seqevent, 456
 - seq64::seqroll, 491
- m_seq
 - seq64::eventedit, 141
 - seq64::eventslots, 153
 - seq64::mastermidibus, 269
 - seq64::midibus, 294
 - seq64::seqdata, 427
 - seq64::seqedit, 442
 - seq64::seqevent, 455
 - seq64::seqkeys, 463
 - seq64::seqroll, 491
 - seq64::seqtime, 496
- m_seq24_interaction
 - seq64::perfroll, 396
 - seq64::seqevent, 455
- m_seq_number
 - seq64::sequence, 536
- m_seqarea_seq_x
 - seq64::mainwid, 247
 - seq64::user_settings, 576
- m_seqarea_seq_y
 - seq64::mainwid, 247
 - seq64::user_settings, 576
- m_seqarea_x
 - seq64::mainwid, 247
 - seq64::user_settings, 576
- m_seqarea_y
 - seq64::mainwid, 247
 - seq64::user_settings, 576
- m_seqchars_x
 - seq64::user_settings, 575
- m_seqchars_y
 - seq64::user_settings, 575
- m_seqdata_wid
 - seq64::seqedit, 443
 - seq64::seqevent, 455
- m_seqedit
 - seq64::seqmenu, 471
- m_seqedit_bgsequence
 - seq64::user_settings, 573
- m_seqedit_key
 - seq64::user_settings, 573
- m_seqedit_scale
 - seq64::user_settings, 573

- m_seqevent_wid
 - seq64::seqedit, [444](#)
- m_seqkeys_wid
 - seq64::seqedit, [443](#)
 - seq64::seqroll, [491](#)
- m_seqroll_wid
 - seq64::seqedit, [444](#)
- m_seqs
 - seq64::perform, [377](#)
- m_seqs_active
 - seq64::perform, [377](#)
- m_seqs_in_set
 - seq64::perfnames, [336](#)
 - seq64::perform, [380](#)
 - seq64::user_settings, [576](#)
- m_seqtime_wid
 - seq64::seqedit, [443](#)
- m_sequence
 - seq64::editable_events, [108](#)
 - seq64::midi_container, [272](#)
- m_sequence_active
 - seq64::perfnames, [336](#)
 - seq64::perfroll, [396](#)
- m_sequence_count
 - seq64::perform, [380](#)
- m_sequence_max
 - seq64::perfnames, [336](#)
 - seq64::perform, [381](#)
 - seq64::perfroll, [396](#)
- m_sequence_offset
 - seq64::perfnames, [336](#)
 - seq64::perfroll, [396](#)
- m_sequence_state
 - seq64::perform, [378](#)
- m_setbox_w
 - seq64::eventslots, [154](#)
 - seq64::perfnames, [336](#)
- m_show_midi
 - seq64::rc_settings, [413](#)
- m_show_octave_letters
 - seq64::seqkeys, [464](#)
- m_showbox
 - seq64::eventedit, [140](#)
- m_sigpipe
 - seq64::mainwnd, [258](#)
- m_size_box_w
 - seq64::perfroll, [395](#)
- m_slot
 - seq64::keybindentry, [208](#)
- m_slots_chars
 - seq64::eventslots, [153](#)
- m_slots_x
 - seq64::eventslots, [154](#)
- m_slots_y
 - seq64::eventslots, [154](#)
- m_smf0_channels
 - seq64::midi_splitter, [283](#)
- m_smf0_channels_count
 - seq64::midi_splitter, [283](#)
- m_smf0_main_sequence
 - seq64::midi_splitter, [283](#)
- m_smf0_seq_number
 - seq64::midi_splitter, [283](#)
- m_smf0_splitter
 - seq64::midifile, [309](#)
- m_snap
 - seq64::perfedit, [328](#)
 - seq64::perfroll, [395](#)
 - seq64::perftime, [405](#)
 - seq64::seqedit, [442](#)
 - seq64::seqevent, [455](#)
 - seq64::seqroll, [491](#)
- m_snap_tick
 - seq64::sequence, [536](#)
- m_song_mute
 - seq64::sequence, [535](#)
- m_spinbutton_bpm
 - seq64::mainwnd, [259](#)
- m_spinbutton_load_offset
 - seq64::mainwnd, [259](#)
- m_spinbutton_ss
 - seq64::mainwnd, [259](#)
- m_standard_bpm
 - seq64::perfedit, [329](#)
- m_starting_tick
 - seq64::perform, [379](#)
- m_stats
 - seq64::rc_settings, [413](#)
- m_status
 - seq64::event, [120](#)
 - seq64::midi_control, [275](#)
 - seq64::seqdata, [427](#)
 - seq64::seqevent, [456](#)
 - seq64::seqroll, [492](#)
- m_sysex
 - seq64::event, [121](#)
- m_sysex_size
 - seq64::event, [121](#)
- m_table
 - seq64::eventedit, [140](#)
 - seq64::perfedit, [327](#)
 - seq64::seqedit, [444](#)
- m_text_size_x
 - seq64::mainwid, [247](#)
- m_text_size_y
 - seq64::mainwid, [247](#)
- m_text_x
 - seq64::user_settings, [574](#)
- m_text_y
 - seq64::user_settings, [574](#)
- m_thru
 - seq64::sequence, [535](#)
- m_tick
 - seq64::maintime, [237](#)
 - seq64::perform, [379](#)
- m_tick_end

- seq64::trigger, 540
- m_tick_offset
 - seq64::perftime, 404
- m_tick_start
 - seq64::trigger, 540
- m_ticks_per_bar
 - seq64::perftroll, 395
- m_time_beat_width
 - seq64::sequence, 536
- m_time_beats_per_measure
 - seq64::sequence, 536
- m_timearea_y
 - seq64::perftime, 405
- m_timeout_connect
 - seq64::mainwnd, 260
- m_timestamp
 - seq64::event, 120
 - seq64::event_list::event_key, 123
- m_toggle_play
 - seq64::seqedit, 445
- m_toggle_q_rec
 - seq64::seqedit, 445
- m_toggle_record
 - seq64::seqedit, 445
- m_toggle_thru
 - seq64::seqedit, 445
- m_tooltips
 - seq64::mainwnd, 258
 - seq64::options, 314
 - seq64::perfedit, 328
 - seq64::seqedit, 445
- m_top_index
 - seq64::eventslots, 154
- m_top_iterator
 - seq64::eventslots, 154
- m_total_seqs
 - seq64::user_settings, 575
- m_tracks_mute_state
 - seq64::perform, 376
- m_trigger_copied
 - seq64::triggers, 549
- m_trigger_offset
 - seq64::sequence, 536
- m_triggers
 - seq64::sequence, 535
 - seq64::triggers, 549
- m_type
 - seq64::keybindentry, 208
- m_undo_stack
 - seq64::triggers, 549
- m_us_per_quarter_note
 - seq64::sequence, 537
- m_use_default_ppqn
 - seq64::midi_splitter, 283
 - seq64::midifile, 309
- m_use_new_font
 - seq64::font, 157
 - seq64::user_settings, 573
- m_usemidiclock
 - seq64::perform, 380
- m_user_filename
 - seq64::rc_settings, 414
- m_user_filename_alt
 - seq64::rc_settings, 414
- m_v_page_increment
 - seq64::perftroll, 394
- m_v_perf_page_increment
 - seq64::user_settings, 574
- m_vadjust
 - seq64::eventedit, 140
 - seq64::gui_drawingarea_gtk2, 181
 - seq64::perfedit, 327
 - seq64::seqedit, 443
- m_vbox
 - seq64::seqedit, 444
- m_vertical_adjust
 - seq64::seqroll, 491
- m_vscroll
 - seq64::eventedit, 140
 - seq64::perfedit, 327
- m_vscroll_new
 - seq64::seqedit, 443
- m_was_active_edit
 - seq64::perform, 377
- m_was_active_main
 - seq64::perform, 377
- m_was_active_names
 - seq64::perform, 378
- m_was_active_perf
 - seq64::perform, 378
- m_was_playing
 - seq64::sequence, 535
- m_white
 - seq64::gui_palette_gtk2, 186
- m_white_pixmap
 - seq64::font, 157
- m_window
 - seq64::gui_drawingarea_gtk2, 181
- m_window_redraw_rate_ms
 - seq64::user_settings, 574
- m_window_x
 - seq64::gui_drawingarea_gtk2, 182
 - seq64::gui_window_gtk2, 191
- m_window_y
 - seq64::gui_drawingarea_gtk2, 182
 - seq64::gui_window_gtk2, 191
- m_with_jack_master
 - seq64::rc_settings, 413
- m_with_jack_master_cond
 - seq64::rc_settings, 413
- m_with_jack_transport
 - seq64::rc_settings, 413
- m_x
 - seq64::click, 88
- m_xy_offset
 - seq64::perfnames, 336

- m_y
 - seq64::click, [88](#)
- m_y_on_b_pixmap
 - seq64::font, [158](#)
- m_yellow
 - seq64::gui_palette_gtk2, [187](#)
- m_zoom
 - seq64::perffroll, [395](#)
 - seq64::seqdata, [427](#)
 - seq64::seqedit, [442](#)
 - seq64::seqevent, [455](#)
 - seq64::seqroll, [491](#)
 - seq64::seqtime, [496](#)
- maintime
 - seq64::maintime, [236](#)
- mainwid
 - seq64::mainwid, [241](#)
- mainwid_border
 - seq64::user_settings, [567](#), [569](#)
- mainwid_grid_style_t
 - seq64::user_settings, [565](#)
- mainwid_spacing
 - seq64::user_settings, [567](#), [569](#)
- mainwid_x
 - seq64::user_settings, [567](#)
- mainwid_y
 - seq64::user_settings, [567](#)
- mainwnd
 - seq64::maintime, [236](#)
 - seq64::mainwid, [246](#)
 - seq64::mainwnd, [253](#)
- mainwnd_cols
 - seq64::user_settings, [567](#), [568](#)
- mainwnd_key_event
 - seq64::perform, [370](#)
- mainwnd_rows
 - seq64::user_settings, [567](#), [568](#)
- make_clock
 - seq64::event, [119](#)
- make_directory
 - seq64, [67](#)
- make_section_name
 - seq64, [72](#)
- manual_alsa_ports
 - seq64::rc_settings, [410](#)
- mark
 - seq64::event, [119](#)
- mark_all
 - seq64::event_list, [129](#)
- mark_out_of_range
 - seq64::event_list, [129](#)
- mark_selected
 - seq64::event_list, [129](#)
 - seq64::sequence, [528](#)
- master_bus
 - seq64::perform, [348](#)
- mastermidibus
 - seq64::mastermidibus, [262](#)
- seq64::midibus, [293](#)
- match
 - seq64::midi_control, [275](#)
- max_active_set
 - seq64::perform, [371](#)
- max_sequence
 - seq64::user_settings, [567](#)
- max_sets
 - seq64::user_settings, [567](#), [568](#)
- max_value
 - seq64::midi_control, [274](#)
- max_zoom
 - seq64::user_settings, [570](#)
- mc_baseline_ppqn
 - seq64::user_settings, [577](#)
- mc_max_zoom
 - seq64::user_settings, [577](#)
- mc_min_zoom
 - seq64::user_settings, [577](#)
- measures
 - seq64::midi_measures, [279](#)
- measures_to_ticks
 - seq64, [63](#)
- measurestring_to_pulses
 - seq64, [56](#)
- merge
 - seq64::event_list, [128](#)
- meta_string
 - seq64::editable_event, [101](#)
- midi_beat_width
 - seq64::user_settings, [570](#), [571](#)
- midi_beats_per_bar
 - seq64::user_settings, [570](#)
- midi_beats_per_minute
 - seq64::user_settings, [570](#), [571](#)
- midi_buss_override
 - seq64::user_settings, [570](#)
- midi_container
 - seq64::event_list, [130](#)
 - seq64::midi_container, [270](#)
- midi_control
 - seq64::midi_control, [274](#)
- midi_control_off
 - seq64::perform, [353](#)
- midi_control_on
 - seq64::perform, [353](#)
- midi_control_toggle
 - seq64::perform, [352](#)
- midi_list
 - seq64::midi_list, [277](#)
- midi_measures
 - seq64::midi_measures, [278](#)
- midi_measures_to_pulses
 - seq64, [57](#)
- midi_ppqn
 - seq64::user_settings, [569](#), [570](#)
- midi_splitter
 - seq64::event_list, [130](#)

- seq64::midi_splitter, 281
- midi_timing
 - seq64::midi_timing, 284
- midi_vector
 - seq64::midi_vector, 287
- midibus
 - seq64::midibus, 290
- midibyte
 - seq64, 51
- midifile
 - seq64::midifile, 297
 - seq64::perform, 376
- midilong
 - seq64, 51
- midipulse
 - seq64, 51
- midishort
 - seq64, 51
- min
 - seq64, 71
- min_value
 - seq64::midi_control, 274
- min_zoom
 - seq64::user_settings, 570
- mod_control
 - seq64::click, 88
 - seq64::keystroke, 231
- mod_control_shift
 - seq64::click, 88
 - seq64::keystroke, 231
- mod_last_tick
 - seq64::sequence, 510
- mod_super
 - seq64::click, 88
 - seq64::keystroke, 231
- mod_timestamp
 - seq64::event, 116
- modifier
 - seq64::click, 88
 - seq64::keystroke, 230
- modify
 - seq64::perform, 347
- modify_current_event
 - seq64::eventslots, 148
- motion_notify
 - seq64::seqroll, 486
- mouse_action
 - seq64::seqedit, 440
- mouse_action_e
 - seq64, 54
- mouse_fruity_callback
 - seq64::options, 313
- mouse_mod4_callback
 - seq64::options, 314
- mouse_seq24_callback
 - seq64::options, 313
- move
 - seq64::triggers, 547
- move_selected
 - seq64::triggers, 546
- move_selected_notes
 - seq64::seqroll, 484
 - seq64::sequence, 523
- move_selected_triggers_to
 - seq64::sequence, 517
- move_selection_box
 - seq64::seqroll, 484
- move_triggers
 - seq64::perform, 351
 - seq64::sequence, 518
- musical_key
 - seq64::sequence, 532
- musical_scale
 - seq64::sequence, 532
- mute_all_tracks
 - seq64::perform, 362
 - seq64::seqmenu, 470
- mute_group_offset
 - seq64::perform, 373
- mute_group_tracks
 - seq64::perform, 355
- mute_screenset
 - seq64::perform, 362
- mutex
 - seq64::mutex, 311
- name
 - seq64::sequence, 509
 - seq64::user_instrument, 551
 - seq64::user_midi_bus, 555
- name_change_callback
 - seq64::seqedit, 437
- name_to_value
 - seq64::editable_event, 99
- new_current_sequence
 - seq64::seqmenu, 468
- new_file
 - seq64::mainwnd, 256
- new_sequence
 - seq64::perform, 349
 - seq64::seqmenu, 468
- next
 - seq64::triggers, 548
- next_data_line
 - seq64::configfile, 92
- next_trigger
 - seq64::triggers, 549
- normal_action
 - seq64::seqroll, 487
- normalize
 - seq64::user_settings, 565
- note_off_length
 - seq64::seqroll, 478
- note_off_margin
 - seq64::sequence, 532
- number
 - seq64::sequence, 507

- off_playing_notes
 - seq64::sequence, [529](#)
- off_queued
 - seq64::sequence, [510](#)
- off_sequences
 - seq64::perform, [356](#)
- offset
 - seq64::trigger, [539](#)
- on_button_press_event
 - seq64::AbstractPerfInput, [84](#)
 - seq64::FruityPerfInput, [160](#)
 - seq64::FruitySeqEventInput, [164](#)
 - seq64::FruitySeqRollInput, [166](#)
 - seq64::Seq24PerfInput, [416](#)
 - seq64::Seq24SeqEventInput, [419](#)
 - seq64::eventslots, [153](#)
 - seq64::mainwid, [245](#)
 - seq64::perfnames, [333](#)
 - seq64::perfroll, [393](#)
 - seq64::perftime, [403](#)
 - seq64::seqdata, [425](#)
 - seq64::seqevent, [453](#)
 - seq64::seqkeys, [461](#)
 - seq64::seqroll, [488](#)
 - seq64::seqtime, [496](#)
- on_button_release_event
 - seq64::AbstractPerfInput, [84](#)
 - seq64::FruityPerfInput, [160](#)
 - seq64::FruitySeqEventInput, [164](#)
 - seq64::FruitySeqRollInput, [166](#)
 - seq64::Seq24PerfInput, [416](#)
 - seq64::Seq24SeqEventInput, [419](#)
 - seq64::eventslots, [153](#)
 - seq64::mainwid, [245](#)
 - seq64::perfnames, [333](#)
 - seq64::perfroll, [393](#)
 - seq64::perftime, [404](#)
 - seq64::seqdata, [425](#)
 - seq64::seqevent, [453](#)
 - seq64::seqkeys, [462](#)
 - seq64::seqroll, [488](#)
 - seq64::seqtime, [496](#)
- on_delete_event
 - seq64::eventedit, [139](#)
 - seq64::mainwnd, [257](#)
 - seq64::perfedit, [326](#)
 - seq64::seqedit, [440](#)
- on_enter_notify_event
 - seq64::seqkeys, [462](#)
 - seq64::seqroll, [490](#)
- on_expose_event
 - seq64::eventslots, [152](#)
 - seq64::maintime, [236](#)
 - seq64::mainwid, [245](#)
 - seq64::perfnames, [333](#)
 - seq64::perfroll, [392](#)
 - seq64::perftime, [403](#)
 - seq64::seqdata, [425](#)
 - seq64::seqevent, [453](#)
 - seq64::seqkeys, [461](#)
 - seq64::seqroll, [488](#)
 - seq64::seqtime, [496](#)
- on_focus_in_event
 - seq64::eventedit, [139](#)
 - seq64::eventslots, [153](#)
 - seq64::mainwid, [246](#)
 - seq64::perfroll, [393](#)
 - seq64::seqedit, [440](#)
 - seq64::seqevent, [454](#)
 - seq64::seqroll, [489](#)
- on_focus_out_event
 - seq64::eventedit, [139](#)
 - seq64::eventslots, [153](#)
 - seq64::mainwid, [246](#)
 - seq64::perfroll, [393](#)
 - seq64::seqedit, [440](#)
 - seq64::seqevent, [454](#)
 - seq64::seqroll, [489](#)
- on_frame_down
 - seq64::eventslots, [153](#)
- on_frame_end
 - seq64::eventslots, [153](#)
- on_frame_home
 - seq64::eventslots, [153](#)
- on_frame_up
 - seq64::eventslots, [153](#)
- on_grouplearnchange
 - seq64::mainwnd, [257](#)
 - seq64::performcallback, [383](#)
- on_key_press_event
 - seq64::eventedit, [139](#)
 - seq64::keybindentry, [208](#)
 - seq64::mainwnd, [257](#)
 - seq64::perfedit, [326](#)
 - seq64::perfroll, [393](#)
 - seq64::seqedit, [441](#)
 - seq64::seqevent, [454](#)
 - seq64::seqroll, [489](#)
- on_key_release_event
 - seq64::mainwnd, [257](#)
- on_leave_notify_event
 - seq64::seqdata, [426](#)
 - seq64::seqkeys, [462](#)
 - seq64::seqroll, [490](#)
- on_left_button_pressed
 - seq64::FruityPerfInput, [161](#)
- on_motion_notify_event
 - seq64::AbstractPerfInput, [84](#)
 - seq64::FruityPerfInput, [160](#)
 - seq64::FruitySeqEventInput, [164](#)
 - seq64::FruitySeqRollInput, [166](#)
 - seq64::Seq24PerfInput, [417](#)
 - seq64::Seq24SeqEventInput, [419](#)
 - seq64::mainwid, [246](#)
 - seq64::perfroll, [393](#)
 - seq64::seqdata, [426](#)

- seq64::seqevent, 453
- seq64::seqkeys, 462
- seq64::seqroll, 488
- on_move_down
 - seq64::eventslots, 153
- on_move_up
 - seq64::eventslots, 153
- on_queued
 - seq64::sequence, 510
- on_realize
 - seq64::eventedit, 138
 - seq64::eventslots, 152
 - seq64::gui_drawingarea_gtk2, 181
 - seq64::gui_window_gtk2, 190
 - seq64::maintime, 236
 - seq64::mainwid, 244
 - seq64::perfedit, 326
 - seq64::perfnames, 333
 - seq64::perfroll, 392
 - seq64::perftime, 403
 - seq64::seqdata, 425
 - seq64::seqedit, 440
 - seq64::seqevent, 453
 - seq64::seqkeys, 461
 - seq64::seqmenu, 471
 - seq64::seqroll, 488
 - seq64::seqtime, 496
- on_right_button_pressed
 - seq64::FruityPerfInput, 161
- on_scroll_event
 - seq64::eventslots, 153
 - seq64::perfnames, 335
 - seq64::perfroll, 393
 - seq64::seqdata, 426
 - seq64::seqedit, 440
 - seq64::seqkeys, 463
 - seq64::seqroll, 489
- on_set_focus
 - seq64::eventedit, 138
 - seq64::seqedit, 440
- on_size_allocate
 - seq64::eventslots, 153
 - seq64::perfnames, 335
 - seq64::perfroll, 393
 - seq64::perftime, 404
 - seq64::seqdata, 426
 - seq64::seqevent, 454
 - seq64::seqkeys, 463
 - seq64::seqroll, 490
 - seq64::seqtime, 496
- on_size_request
 - seq64::perfroll, 394
- open_file
 - seq64::mainwnd, 253
- open_performance_edit
 - seq64::mainwnd, 256
- open_performance_edit_2
 - seq64::mainwnd, 256
- operator<
 - seq64::event, 113
 - seq64::event_list::event_key, 123
 - seq64::trigger, 539
- operator=
 - seq64::automutex, 85
 - seq64::click, 87
 - seq64::editable_event, 99
 - seq64::editable_events, 106
 - seq64::event, 113
 - seq64::event_list, 126
 - seq64::gui_drawingarea_gtk2, 174
 - seq64::keystroke, 229
 - seq64::maintime, 236
 - seq64::rc_settings, 408
 - seq64::sequence, 506
 - seq64::triggers, 542
 - seq64::user_instrument, 551
 - seq64::user_midi_bus, 555
 - seq64::user_settings, 565
- options
 - seq64::keybindentry, 208
 - seq64::keys_perform, 222
 - seq64::options, 313
 - seq64::perform, 376
- options_dialog
 - seq64::mainwnd, 254
- optionsfile
 - seq64::keys_perform, 222
 - seq64::optionsfile, 316
 - seq64::perform, 376
- orange
 - seq64::gui_palette_gtk2, 186
- output
 - seq64::jack_assistant, 197
- output_func
 - seq64::perform, 362
- output_thread_func
 - seq64, 70
- padded_height
 - seq64::font, 157
- page_movement
 - seq64::eventslots, 151
- page_topper
 - seq64::eventslots, 151
- pager_index
 - seq64::eventslots, 146
- paint
 - seq64::event, 119
- parent
 - seq64::editable_event, 99
 - seq64::jack_assistant, 194
- parse
 - seq64::configfile, 93
 - seq64::midifile, 297
 - seq64::optionsfile, 316
 - seq64::userfile, 579
- parse_command_line_options

- seq64, 65
- parse_options_files
 - seq64, 64
- parse_prop_header
 - seq64::midifile, 299
- parse_proprietary_track
 - seq64::midifile, 300
- parse_smf_0
 - seq64::midifile, 299
- parse_smf_1
 - seq64::midifile, 299
- partial_assign
 - seq64::sequence, 506
- pass_sysex
 - seq64::rc_settings, 410
- paste
 - seq64::triggers, 546
- paste_selected
 - seq64::sequence, 521
- paste_trigger
 - seq64::sequence, 517
- pattern_edit
 - seq64::keys_perform, 217, 218
- pause
 - seq64::keys_perform, 217
 - seq64::sequence, 529
- pause_key
 - seq64::perform, 368
- pause_playing
 - seq64::mainwnd, 255
 - seq64::perfedit, 326
 - seq64::perform, 367
- perf
 - seq64::gui_drawingarea_gtk2, 174
 - seq64::gui_window_gtk2, 189
 - seq64::options, 313
- perf_h_page_increment
 - seq64::user_settings, 568, 570
- perf_modify
 - seq64::eventedit, 137
- perf_v_page_increment
 - seq64::user_settings, 568, 570
- perfedit
 - seq64::perfedit, 323
 - seq64::perfnames, 335
 - seq64::perfroll, 394
 - seq64::perftime, 404
- perfnames
 - seq64::perfnames, 331
- perform
 - seq64::keys_perform, 222
 - seq64::perform, 346
 - seq64::sequence, 534
- perfroll
 - seq64::FruityPerfInput, 162
 - seq64::Seq24PerfInput, 418
 - seq64::perfroll, 388
- perfroll_key_event
 - seq64::perform, 370
- perftime
 - seq64::perftime, 400
- pixel_to_tick
 - seq64::perftime, 402
- play
 - seq64::mastermidibus, 266
 - seq64::midibus, 292
 - seq64::perform, 359
 - seq64::sequence, 513
 - seq64::triggers, 543
- play_change_callback
 - seq64::seqedit, 438
- play_note_off
 - seq64::sequence, 529
- play_note_on
 - seq64::sequence, 528
- playback_key_event
 - seq64::perform, 370
- poll_for_midi
 - seq64::mastermidibus, 265
- pop_redo
 - seq64::sequence, 507
- pop_trigger_undo
 - seq64::perform, 352
 - seq64::sequence, 507
- pop_undo
 - seq64::sequence, 507
 - seq64::triggers, 543
- popup_event_menu
 - seq64::seqedit, 439
- popup_menu
 - seq64::perfedit, 325
 - seq64::seqedit, 439
 - seq64::seqmenu, 468
- popup_midibus_menu
 - seq64::seqedit, 439
- popup_midich_menu
 - seq64::seqedit, 439
- popup_sequence_menu
 - seq64::seqedit, 439
- popup_tool_menu
 - seq64::seqedit, 439
- port_exit
 - seq64::mastermidibus, 266
- port_start
 - seq64::mastermidibus, 266
- position
 - seq64::jack_assistant, 196
 - seq64::midi_container, 272
- position_increment
 - seq64::midi_container, 272
- position_jack
 - seq64::perform, 356
- position_reset
 - seq64::midi_container, 272
- pow2
 - seq64::midifile, 301

- ppqn
 - seq64::mainwnd, [254](#)
 - seq64::midi_splitter, [282](#)
 - seq64::midi_timing, [285](#)
 - seq64::midifile, [299](#)
- ppqn_is_valid
 - seq64, [67](#)
- print
 - seq64::event, [120](#)
 - seq64::event_list, [130](#)
 - seq64::mastermidibus, [264](#)
 - seq64::midibus, [291](#)
 - seq64::sequence, [512](#)
 - seq64::triggers, [543](#)
- print_keys
 - seq64::rc_settings, [410](#)
- print_triggers
 - seq64::perform, [350](#)
 - seq64::sequence, [513](#)
- priority
 - seq64::rc_settings, [409](#)
- private_bus
 - seq64::user_settings, [571](#)
- private_instrument
 - seq64::user_settings, [571](#)
- process_events
 - seq64::lash, [232](#)
- progress_bar_colored
 - seq64::user_settings, [568](#), [570](#)
- progress_bar_thick
 - seq64::user_settings, [568](#), [570](#)
- progress_color
 - seq64::gui_palette_gtk2, [185](#)
- prop_item_size
 - seq64::midifile, [307](#)
- pulse_length_us
 - seq64, [61](#)
- pulses_to_measurestring
 - seq64, [55](#)
- pulses_to_midi_measures
 - seq64, [55](#)
- pulses_to_string
 - seq64, [55](#)
- pulses_to_timestring
 - seq64, [56](#)
- push_quantize
 - seq64::sequence, [531](#)
- push_trigger_undo
 - seq64::perform, [352](#)
 - seq64::sequence, [507](#)
- push_undo
 - seq64::sequence, [507](#)
 - seq64::triggers, [543](#)
- put
 - seq64::midi_container, [271](#)
 - seq64::midi_list, [277](#)
 - seq64::midi_vector, [287](#)
- put_event_on_bus
 - seq64::sequence, [533](#)
- q_rec_change_callback
 - seq64::seqedit, [438](#)
- quantize_events
 - seq64::sequence, [531](#)
- query_save_changes
 - seq64::mainwnd, [257](#)
- queue
 - seq64::keys_perform, [215](#)
- quit
 - seq64::gui_assistant, [168](#)
 - seq64::gui_assistant_gtk2, [170](#)
 - seq64::gui_window_gtk2, [190](#)
- rc
 - seq64, [71](#)
- rc_settings
 - seq64::rc_settings, [408](#)
- read_byte
 - seq64::midifile, [302](#)
- read_byte_array
 - seq64::midifile, [303](#)
- read_long
 - seq64::midifile, [302](#)
- read_seq_number
 - seq64::midifile, [304](#)
- read_short
 - seq64::midifile, [302](#)
- read_track_name
 - seq64::midifile, [304](#)
- read_varinum
 - seq64::midifile, [302](#)
- record_change_callback
 - seq64::seqedit, [438](#)
- red
 - seq64::gui_palette_gtk2, [186](#)
- redo_callback
 - seq64::seqedit, [438](#)
- redraw
 - seq64::mainwid, [242](#)
 - seq64::perfnames, [332](#)
 - seq64::seqdata, [423](#)
 - seq64::sequevent, [449](#)
 - seq64::seqmenu, [471](#)
 - seq64::seqroll, [480](#)
 - seq64::seqtime, [495](#)
- redraw_dirty_sequences
 - seq64::perfnames, [332](#)
 - seq64::perfroll, [389](#)
- redraw_events
 - seq64::seqroll, [480](#)
- redraw_period_ms
 - seq64::gui_window_gtk2, [190](#)
- redraw_progress
 - seq64::perfroll, [389](#)
- remove
 - seq64::editable_events, [108](#)
 - seq64::event_list, [127](#)

- seq64::sequence, 534
- seq64::triggers, 545
- remove_all
 - seq64::sequence, 534
- remove_marked
 - seq64::event_list, 129
 - seq64::sequence, 528
- remove_selected
 - seq64::sequence, 528
 - seq64::triggers, 546
- render_number
 - seq64::seqdata, 424
- render_string
 - seq64::gui_drawingarea_gtk2, 177
- render_string_on_drawable
 - seq64::font, 156
- render_string_on_pixmap
 - seq64::gui_drawingarea_gtk2, 177
- replace
 - seq64::editable_events, 108
 - seq64::keys_perform, 215
- reset
 - seq64::mainwid, 242
 - seq64::perftime, 401
 - seq64::seqdata, 423
 - seq64::seqevent, 449
 - seq64::seqkeys, 461
 - seq64::seqroll, 480
 - seq64::seqtime, 495
 - seq64::sequence, 529
- reset_draw_marker
 - seq64::sequence, 529
- reset_draw_trigger_marker
 - seq64::sequence, 529
 - seq64::triggers, 549
- reset_sequences
 - seq64::perform, 358
- restart_sysex
 - seq64::event, 118
- restore_playing_state
 - seq64::perform, 364
- RevSlotMap
 - seq64::keys_perform, 214
- reveal_alsa_ports
 - seq64::rc_settings, 410
- s_arg_list
 - seq64, 79
- s_build_chord_generator
 - seq64, 79
- s_build_edit_highlight
 - seq64, 80
- s_build_follow_progress
 - seq64, 80
- s_build_highlight_empty
 - seq64, 79
- s_build_jack_session
 - seq64, 79
- s_build_jack_support
 - seq64, 79
- s_build_lash_support
 - seq64, 79
- s_build_midi_vector
 - seq64, 80
- s_build_pause_support
 - seq64, 79
- s_build_solid_grid
 - seq64, 80
- s_build_timesig_tempo
 - seq64, 80
- s_build_use_event_map
 - seq64, 79
- s_character_mapping
 - seq64, 80
- s_global_lash_driver
 - seq64, 80
- s_handlesize
 - seq64, 80, 81
- s_help_1a
 - seq64, 79
- s_help_1b
 - seq64, 79
- s_help_2
 - seq64, 79
- s_help_3
 - seq64, 79
- s_help_4
 - seq64, 79
- s_jitter_amount
 - seq64, 80
- SEQ64_2BUTTON_PRESS
 - seq64, 52
- SEQ64_3BUTTON_PRESS
 - seq64, 52
- SEQ64_BUTTON1_MASK
 - seq64, 52
- SEQ64_BUTTON2_MASK
 - seq64, 52
- SEQ64_BUTTON3_MASK
 - seq64, 52
- SEQ64_BUTTON4_MASK
 - seq64, 52
- SEQ64_BUTTON5_MASK
 - seq64, 52
- SEQ64_BUTTON_PRESS
 - seq64, 52
- SEQ64_BUTTON_RELEASE
 - seq64, 52
- SEQ64_CONTROL_MASK
 - seq64, 51
- SEQ64_DELETE
 - seq64, 52
- SEQ64_DESTROY
 - seq64, 52
- SEQ64_EVENT_LAST
 - seq64, 52
- SEQ64_EXPOSE

- seq64, [52](#)
- SEQ64_HYPER_MASK
 - seq64, [52](#)
- SEQ64_KEY_PRESS
 - seq64, [52](#)
- SEQ64_KEY_RELEASE
 - seq64, [52](#)
- SEQ64_LOCK_MASK
 - seq64, [51](#)
- SEQ64_MASK_MAX
 - seq64, [52](#)
- SEQ64_META_MASK
 - seq64, [52](#)
- SEQ64_MOD1_MASK
 - seq64, [51](#)
- SEQ64_MOD2_MASK
 - seq64, [51](#)
- SEQ64_MOD3_MASK
 - seq64, [51](#)
- SEQ64_MOD4_MASK
 - seq64, [51](#)
- SEQ64_MOD5_MASK
 - seq64, [52](#)
- SEQ64_MOTION_NOTIFY
 - seq64, [52](#)
- SEQ64_NO_MASK
 - seq64, [51](#)
- SEQ64_NOTHING
 - seq64, [52](#)
- SEQ64_RELEASE_MASK
 - seq64, [52](#)
- SEQ64_SCROLL_DOWN
 - seq64, [52](#)
- SEQ64_SCROLL_LEFT
 - seq64, [52](#)
- SEQ64_SCROLL_RIGHT
 - seq64, [52](#)
- SEQ64_SCROLL_UP
 - seq64, [52](#)
- SEQ64_SCROLL
 - seq64, [52](#)
- SEQ64_SHIFT_MASK
 - seq64, [51](#)
- SEQ64_SUPER_MASK
 - seq64, [52](#)
- save_events
 - seq64::editable_events, [106](#)
 - seq64::eventslots, [149](#)
- save_file
 - seq64::mainwnd, [256](#)
- save_playing_state
 - seq64::perform, [364](#)
- save_user_config
 - seq64::user_settings, [568](#)
- screenset_dn
 - seq64::keys_perform, [216](#)
- screenset_up
 - seq64::keys_perform, [216](#)
- scroll_hadjust
 - seq64::gui_drawingarea_gtk2, [180](#)
 - seq64::gui_window_gtk2, [190](#)
- scroll_hset
 - seq64::gui_drawingarea_gtk2, [181](#)
 - seq64::gui_window_gtk2, [190](#)
- scroll_offset_x
 - seq64::seqroll, [486](#)
- scroll_offset_y
 - seq64::seqroll, [486](#)
- scroll_vadjust
 - seq64::gui_drawingarea_gtk2, [180](#)
 - seq64::gui_window_gtk2, [190](#)
- scroll_vset
 - seq64::gui_drawingarea_gtk2, [181](#)
 - seq64::gui_window_gtk2, [190](#)
- select
 - seq64::event, [119](#)
 - seq64::triggers, [546](#)
- select_action
 - seq64::seqroll, [487](#)
- select_action_e
 - seq64::sequence, [506](#)
- select_all
 - seq64::event_list, [130](#)
 - seq64::sequence, [521](#)
- select_all_notes
 - seq64::sequence, [520](#)
- select_and_mute_group
 - seq64::perform, [355](#)
- select_event
 - seq64::eventslots, [149](#)
- select_events
 - seq64::sequence, [519](#), [520](#)
- select_fg_bg_colors
 - seq64::mainwid, [244](#)
- select_group_mute
 - seq64::perform, [355](#)
- select_note_events
 - seq64::sequence, [519](#)
- select_trigger
 - seq64::sequence, [515](#)
- selected
 - seq64::trigger, [539](#), [540](#)
- selected_trigger_end
 - seq64::sequence, [517](#)
- selected_trigger_start
 - seq64::sequence, [517](#)
- selecting
 - seq64::seqroll, [487](#)
- Seq24PerfInput
 - seq64::Seq24PerfInput, [416](#)
 - seq64::perfroll, [394](#)
- Seq24SeqEventInput
 - seq64::Seq24SeqEventInput, [418](#)
 - seq64::seqevent, [455](#)
- seq64, [41](#)
 - adjustment_dummy, [72](#)

beats_per_minute_from_tempo, 60
 build_details, 65
 bussbyte, 51
 c_backsequence, 76
 c_bpmtag, 76
 c_chord_text, 78
 c_controller_names, 73
 c_interval_text, 78
 c_key_text, 78
 c_mainwid_x, 80
 c_mainwid_y, 81
 c_max_busses, 78
 c_max_instruments, 78
 c_midi_control_bpm_dn, 76
 c_midi_control_bpm_up, 76
 c_midi_control_mod_glearn, 77
 c_midi_control_mod_gmute, 76
 c_midi_control_mod_queue, 76
 c_midi_control_mod_replace, 76
 c_midi_control_mod_snapshot, 76
 c_midi_control_play_ss, 77
 c_midi_control_ss_dn, 76
 c_midi_control_ss_up, 76
 c_midi_controls, 77
 c_midi_track_ctrl, 76
 c_midibus, 75
 c_midibus_input_size, 75
 c_midibus_output_size, 75
 c_midibus_sysex_chunk, 75
 c_midich, 75
 c_midiclocks, 76
 c_midictrl, 76
 c_music_scales, 53
 c_musickey, 76
 c_musicscale, 76
 c_mutegroups, 76
 c_notes, 76
 c_quantize_events, 81
 c_quantize_notes, 81
 c_reserved, 81
 c_scale_blues, 53
 c_scale_c_whole_tone, 53
 c_scale_harmonic_minor, 53
 c_scale_major, 53
 c_scale_major_pentatonic, 53
 c_scale_melodic_minor, 53
 c_scale_minor, 53
 c_scale_minor_pentatonic, 53
 c_scale_off, 53
 c_scale_size, 53
 c_scales_policy, 77
 c_scales_text, 78
 c_scales_transpose_dn, 77
 c_scales_transpose_up, 77
 c_select_all_events, 81
 c_select_all_notes, 81
 c_select_inverse_events, 81
 c_select_inverse_notes, 81
 c_status_queue, 80
 c_status_replace, 80
 c_status_snapshot, 80
 c_swing_notes, 81
 c_tighten_events, 81
 c_tighten_notes, 81
 c_timesig, 76
 c_transpose_h, 81
 c_transpose_notes, 81
 c_triggers, 76
 c_triggers_new, 76
 choose_ppqn, 71
 clamp, 73
 clock_e, 52
 clock_tick_duration_bogus, 62
 clock_ticks_from_ppqn, 63
 create_lash_driver, 70
 DRAW_FIN, 54
 DRAW_NORMAL_LINKED, 54
 DRAW_NOTE_OFF, 54
 DRAW_NOTE_ON, 54
 delete_lash_driver, 70
 delta_time_us_to_ticks, 61
 double_ticks_from_ppqn, 63
 draw_type, 53
 e_action_draw, 54
 e_action_grow, 54
 e_action_select, 54
 e_clock_mod, 53
 e_clock_off, 53
 e_clock_pos, 53
 e_fruity_interaction, 53
 e_number_of_interactions, 53
 e_seq24_interaction, 53
 EVENT_AFTERTOUCH, 73
 EVENT_ANY, 73
 EVENT_CHANNEL_PRESSURE, 73
 EVENT_CLEAR_CHAN_MASK, 75
 EVENT_CONTROL_CHANGE, 73
 EVENT_GET_CHAN_MASK, 74
 EVENT_MIDI_ACTIVE_SENS, 74
 EVENT_MIDI_CLOCK, 74
 EVENT_MIDI_CONTINUE, 74
 EVENT_MIDI_META, 74
 EVENT_MIDI_QUARTER_FRAME, 73
 EVENT_MIDI_RESET, 74
 EVENT_MIDI_SONG_F4, 74
 EVENT_MIDI_SONG_F5, 74
 EVENT_MIDI_SONG_F9, 74
 EVENT_MIDI_SONG_FD, 74
 EVENT_MIDI_SONG_POS, 74
 EVENT_MIDI_SONG_SELECT, 74
 EVENT_MIDI_START, 74
 EVENT_MIDI_STOP, 74
 EVENT_MIDI_SYSEX_END, 74
 EVENT_MIDI_SYSEX, 73
 EVENT_MIDI_TUNE_SELECT, 74
 EVENT_NOTE_OFF, 73

EVENT_NOTE_ON, 73
EVENT_NULL_CHANNEL, 74
EVENT_PITCH_WHEEL, 73
EVENT_PROGRAM_CHANGE, 73
EVENT_STATUS_BIT, 73
EVENT_SYSEX_CONTINUE, 74
EVENT_SYSEX_END, 74
EVENT_SYSEX, 74
extract_timing_numbers, 54
file_access, 66
file_accessible, 66
file_executable, 67
file_exists, 66
file_is_directory, 67
file_readable, 66
file_writable, 66
font_render, 72
g_rc_settings, 80
g_user_settings, 80
gs_mainwid_pointer, 80
gs_perfedit_pointer_0, 81
gs_perfedit_pointer_1, 81
help_check, 64
input_thread_func, 71
interaction_method_t, 53
jack_process_callback, 69
jack_session_callback, 69
jack_shutdown_callback, 68
jack_sync_callback, 68
jack_timebase_callback, 68
keyval_name, 69
keyval_normalize, 70
lash_driver, 70
log2_time_sig_value, 59
long_options, 79
make_directory, 67
make_section_name, 72
measures_to_ticks, 63
measurestring_to_pulses, 56
midi_measures_to_pulses, 57
midibyte, 51
midilong, 51
midipulse, 51
midishort, 51
min, 71
mouse_action_e, 54
output_thread_func, 70
parse_command_line_options, 65
parse_options_files, 64
ppqn_is_valid, 67
pulse_length_us, 61
pulses_to_measurestring, 55
pulses_to_midi_measures, 55
pulses_to_string, 55
pulses_to_timestring, 56
rc, 71
s_arg_list, 79
s_build_chord_generator, 79
s_build_edit_highlight, 80
s_build_follow_progress, 80
s_build_highlight_empty, 79
s_build_jack_session, 79
s_build_jack_support, 79
s_build_lash_support, 79
s_build_midi_vector, 80
s_build_pause_support, 79
s_build_solid_grid, 80
s_build_timesig_tempo, 80
s_build_use_event_map, 79
s_character_mapping, 80
s_global_lash_driver, 80
s_handlesize, 80, 81
s_help_1a, 79
s_help_1b, 79
s_help_2, 79
s_help_3, 79
s_help_4, 79
s_jitter_amount, 80
SEQ64_2BUTTON_PRESS, 52
SEQ64_3BUTTON_PRESS, 52
SEQ64_BUTTON1_MASK, 52
SEQ64_BUTTON2_MASK, 52
SEQ64_BUTTON3_MASK, 52
SEQ64_BUTTON4_MASK, 52
SEQ64_BUTTON5_MASK, 52
SEQ64_BUTTON_PRESS, 52
SEQ64_BUTTON_RELEASE, 52
SEQ64_CONTROL_MASK, 51
SEQ64_DELETE, 52
SEQ64_DESTROY, 52
SEQ64_EVENT_LAST, 52
SEQ64_EXPOSE, 52
SEQ64_HYPER_MASK, 52
SEQ64_KEY_PRESS, 52
SEQ64_KEY_RELEASE, 52
SEQ64_LOCK_MASK, 51
SEQ64_MASK_MAX, 52
SEQ64_META_MASK, 52
SEQ64_MOD1_MASK, 51
SEQ64_MOD2_MASK, 51
SEQ64_MOD3_MASK, 51
SEQ64_MOD4_MASK, 51
SEQ64_MOD5_MASK, 52
SEQ64_MOTION_NOTIFY, 52
SEQ64_NO_MASK, 51
SEQ64_NOTHING, 52
SEQ64_RELEASE_MASK, 52
SEQ64_SCROLL_DOWN, 52
SEQ64_SCROLL_LEFT, 52
SEQ64_SCROLL_RIGHT, 52
SEQ64_SCROLL_UP, 52
SEQ64_SCROLL, 52
SEQ64_SHIFT_MASK, 51
SEQ64_SUPER_MASK, 52
seq_event_type_t, 52
seq_modifier_t, 51

- seq_scroll_direction_t, 52
- shorten_file_spec, 58
- string_is_void, 59
- string_not_void, 58
- string_to_midibyte, 58
- string_to_pulses, 57
- strings_match, 59
- tempo_from_beats_per_minute, 61
- tempo_to_bytes, 60
- ticks_to_delta_time_us, 62
- timestring_to_pulses, 57
- to_string, 65
- update_mainwid_sequences, 72
- update_perfedit_sequences, 72
- usr, 71
- versiontext, 79
- write_options_files, 65
- zoom_power_of_2, 60
- seq64::AbstractPerfInput, 83
 - ~AbstractPerfInput, 84
 - AbstractPerfInput, 84
 - m_adding_pressed, 84
 - on_button_press_event, 84
 - on_button_release_event, 84
 - on_motion_notify_event, 84
- seq64::FruityPerfInput, 158
 - FruityPerfInput, 160
 - m_current_x, 162
 - m_current_y, 162
 - on_button_press_event, 160
 - on_button_release_event, 160
 - on_left_button_pressed, 161
 - on_motion_notify_event, 160
 - on_right_button_pressed, 161
 - perfroll, 162
 - update_mouse_pointer, 161
- seq64::FruitySeqEventInput, 162
 - FruitySeqEventInput, 163
 - m_is_drag_pasting, 165
 - m_is_drag_pasting_start, 165
 - m_justselected_one, 165
 - on_button_press_event, 164
 - on_button_release_event, 164
 - on_motion_notify_event, 164
 - update_mouse_pointer, 163
- seq64::FruitySeqRollInput, 165
 - FruitySeqRollInput, 165
 - m_drag_paste_start_pos, 167
 - m_erase_painting, 167
 - on_button_press_event, 166
 - on_button_release_event, 166
 - on_motion_notify_event, 166
 - update_mouse_pointer, 165
- seq64::Seq24PerfInput, 415
 - handle_motion_key, 417
 - is_adding, 417
 - m_adding, 418
 - m_effective_tick, 418
 - on_button_press_event, 416
 - on_button_release_event, 416
 - on_motion_notify_event, 417
 - perfroll, 418
 - Seq24PerfInput, 416
 - set_adding, 417
- seq64::Seq24SeqEventInput, 418
 - m_adding, 419
 - on_button_press_event, 419
 - on_button_release_event, 419
 - on_motion_notify_event, 419
 - Seq24SeqEventInput, 418
 - set_adding, 418
- seq64::automutex, 84
 - ~automutex, 85
 - automutex, 85
 - m_safety_mutex, 85
 - operator=, 85
- seq64::click, 86
 - button, 88
 - click, 87
 - is_left, 88
 - is_middle, 88
 - is_press, 88
 - is_right, 88
 - m_button, 88
 - m_is_press, 88
 - m_modifier, 88
 - m_x, 88
 - m_y, 88
 - mod_control, 88
 - mod_control_shift, 88
 - mod_super, 88
 - modifier, 88
 - operator=, 87
 - x, 88
 - y, 88
- seq64::condition_var, 89
 - condition_var, 90
 - m_cond, 90
 - signal, 90
 - sm_cond, 90
 - wait, 90
- seq64::configfile, 90
 - ~configfile, 92
 - configfile, 92
 - line_after, 92
 - m_d, 93
 - m_line, 93
 - m_name, 93
 - next_data_line, 92
 - parse, 93
 - write, 93
- seq64::editable_event, 93
 - ~editable_event, 98
 - analyze, 101
 - category, 99, 100
 - category_channel_message, 97

- category_meta_event, 97
- category_name, 97
- category_prop_event, 97
- category_string, 99
- category_system_message, 97
- category_t, 97
- channel_string, 101
- data_string, 101
- editable_event, 98
- format_timestamp, 101
- m_category, 103
- m_format_timestamp, 103
- m_name_category, 103
- m_name_channel, 103
- m_name_data, 103
- m_name_meta, 103
- m_name_seqspect, 103
- m_name_status, 103
- m_name_timestamp, 103
- m_parent, 103
- meta_string, 101
- name_to_value, 99
- operator=, 99
- parent, 99
- seqspec_string, 101
- set_status_from_string, 100
- sm_category_arrays, 102
- sm_category_names, 102
- sm_channel_event_names, 102
- sm_meta_event_names, 102
- sm_prop_event_names, 102
- sm_system_event_names, 102
- status_string, 101
- stock_event_string, 101
- time_as_measures, 100
- time_as_minutes, 100
- time_as_pulses, 100
- timestamp, 100
- timestamp_format_t, 97
- timestamp_measures, 98
- timestamp_pulses, 98
- timestamp_string, 100
- timestamp_time, 98
- value_to_name, 99
- seq64::editable_event::name_value_t, 311
 - event_name, 311
 - event_value, 311
- seq64::editable_events, 103
 - ~editable_events, 106
 - add, 107
 - begin, 107
 - clear, 108
 - const_iterator, 105
 - count, 107
 - current_event, 108
 - editable_events, 105, 106
 - end, 107
 - Events, 105
 - events, 107
 - EventsPair, 105
 - eventslots, 108
 - iterator, 105
 - Key, 105
 - load_events, 106
 - m_current_event, 108
 - m_events, 108
 - m_midi_parameters, 108
 - m_sequence, 108
 - operator=, 106
 - remove, 108
 - replace, 108
 - save_events, 106
 - string_to_pulses, 106
 - timing, 106
- seq64::event, 109
 - ~event, 113
 - append_sysex, 118
 - check_channel, 114
 - clear_link, 119
 - data, 119
 - decrement_data1, 118
 - decrement_data2, 118
 - event, 113
 - get_channel, 114
 - get_data, 118
 - get_linked, 119
 - get_note, 119
 - get_note_velocity, 119
 - get_rank, 120
 - get_status, 117
 - get_sysex, 118
 - get_sysex_size, 118
 - get_timestamp, 114
 - increment_data1, 118
 - increment_data2, 118
 - is_channel_msg, 115
 - is_desired_cc_or_not_cc, 116
 - is_linked, 119
 - is_marked, 119
 - is_note, 120
 - is_note_msg, 115
 - is_note_off, 120
 - is_note_on, 120
 - is_one_byte_msg, 115
 - is_painted, 119
 - is_selected, 119
 - is_two_byte_msg, 115
 - link, 118
 - m_channel, 120
 - m_data, 121
 - m_has_link, 121
 - m_linked, 121
 - m_marked, 121
 - m_painted, 121
 - m_selected, 121
 - m_status, 120

- m_sysex, 121
- m_sysex_size, 121
- m_timestamp, 120
- make_clock, 119
- mark, 119
- mod_timestamp, 116
- operator<, 113
- operator=, 113
- paint, 119
- print, 120
- restart_sysex, 118
- select, 119
- set_channel, 117
- set_data, 117
- set_note, 119
- set_note_velocity, 119
- set_status, 116, 117
- set_sysex_size, 118
- set_timestamp, 114
- unmark, 119
- unpaint, 119
- unselect, 119
- seq64::event_list, 123
 - ~event_list, 126
 - add, 127
 - any_selected_notes, 130
 - begin, 126
 - clear, 128
 - clear_links, 129
 - const_iterator, 126
 - count, 127
 - count_selected_events, 130
 - count_selected_notes, 130
 - dref, 128
 - editable_events, 130
 - empty, 127
 - end, 126
 - event_list, 126
 - Events, 126
 - events, 130
 - EventsPair, 126
 - is_modified, 127
 - iterator, 126
 - link_new, 128
 - m_events, 130
 - m_is_modified, 130
 - mark_all, 129
 - mark_out_of_range, 129
 - mark_selected, 129
 - merge, 128
 - midi_container, 130
 - midi_splitter, 130
 - operator=, 126
 - print, 130
 - remove, 127
 - remove_marked, 129
 - select_all, 130
 - sequence, 130
 - sort, 128
 - unmark_all, 129
 - unmodify, 127
 - unpaint_all, 129
 - unselect_all, 130
 - verify_and_link, 129
- seq64::event_list::event_key, 121
 - event_key, 122, 123
 - m_rank, 123
 - m_timestamp, 123
 - operator<, 123
- seq64::eventedit, 131
 - ~eventedit, 135
 - change_focus, 137
 - close_out, 138
 - enqueue_draw, 136
 - eventedit, 134
 - eventslots, 140
 - handle_cancel, 138
 - handle_close, 138
 - handle_delete, 138
 - handle_insert, 138
 - handle_modify, 138
 - handle_save, 138
 - m_bottbox, 140
 - m_button_cancel, 140
 - m_button_del, 140
 - m_button_ins, 140
 - m_button_modify, 140
 - m_button_save, 140
 - m_editbox, 140
 - m_entry_ev_data_0, 141
 - m_entry_ev_data_1, 141
 - m_entry_ev_name, 141
 - m_entry_ev_timestamp, 141
 - m_eventslots, 140
 - m_have_focus, 141
 - m_htopbox, 140
 - m_label_category, 141
 - m_label_channel, 141
 - m_label_ev_count, 141
 - m_label_modified, 141
 - m_label_ppqn, 141
 - m_label_right, 141
 - m_label_seq_name, 140
 - m_label_spacer, 141
 - m_label_time_fmt, 141
 - m_label_time_sig, 140
 - m_optsbox, 140
 - m_rightbox, 140
 - m_seq, 141
 - m_showbox, 140
 - m_table, 140
 - m_vadjust, 140
 - m_vscroll, 140
 - on_delete_event, 139
 - on_focus_in_event, 139
 - on_focus_out_event, 139

- on_key_press_event, 139
- on_realize, 138
- on_set_focus, 138
- perf_modify, 137
- set_dirty, 137
- set_event_category, 136
- set_event_data_0, 136
- set_event_data_1, 137
- set_event_name, 136
- set_event_timestamp, 136
- set_seq_count, 136
- set_seq_ppqn, 136
- set_seq_time_sig, 136
- set_seq_title, 136
- v_adjustment, 137
- seq64::eventslots, 142
 - ~eventslots, 146
 - change_vert, 151
 - convert_y, 150
 - current_index, 146
 - decrement_bottom, 152
 - decrement_current, 152
 - decrement_top, 151
 - delete_current_event, 147
 - draw_event, 150
 - draw_events, 150
 - enqueue_draw, 150
 - event_count, 146
 - eventedit, 153
 - eventslots, 146
 - increment_bottom, 152
 - increment_current, 152
 - increment_top, 152
 - insert_event, 147
 - line_count, 146
 - line_increment, 146
 - line_maximum, 146
 - load_events, 146
 - m_bottom_iterator, 154
 - m_char_w, 153
 - m_current_index, 154
 - m_current_iterator, 154
 - m_event_container, 153
 - m_event_count, 154
 - m_line_count, 154
 - m_line_maximum, 154
 - m_line_overlap, 154
 - m_pager_index, 154
 - m_parent, 153
 - m_seq, 153
 - m_setbox_w, 154
 - m_slots_chars, 153
 - m_slots_x, 154
 - m_slots_y, 154
 - m_top_index, 154
 - m_top_iterator, 154
 - modify_current_event, 148
 - on_button_press_event, 153
 - on_button_release_event, 153
 - on_expose_event, 152
 - on_focus_in_event, 153
 - on_focus_out_event, 153
 - on_frame_down, 153
 - on_frame_end, 153
 - on_frame_home, 153
 - on_frame_up, 153
 - on_move_down, 153
 - on_move_up, 153
 - on_realize, 152
 - on_scroll_event, 153
 - on_size_allocate, 153
 - page_movement, 151
 - page_topper, 151
 - pager_index, 146
 - save_events, 149
 - select_event, 149
 - set_current_event, 146
 - set_text, 149
 - top_index, 146
- seq64::font, 154
 - BLACK_ON_CYAN, 156
 - BLACK_ON_YELLOW, 156
 - BLACK, 156
 - CYAN_ON_BLACK, 156
 - char_height, 157
 - char_width, 157
 - Color, 156
 - font, 156
 - init, 156
 - m_b_on_c_pixmap, 158
 - m_b_on_y_pixmap, 158
 - m_black_pixmap, 157
 - m_c_on_b_pixmap, 158
 - m_cell_h, 157
 - m_cell_w, 157
 - m_clip_mask, 158
 - m_font_h, 157
 - m_font_w, 157
 - m_offset, 157
 - m_padded_h, 157
 - m_pixmap, 157
 - m_use_new_font, 157
 - m_white_pixmap, 157
 - m_y_on_b_pixmap, 158
 - padded_height, 157
 - render_string_on_drawable, 156
 - WHITE, 156
 - YELLOW_ON_BLACK, 156
- seq64::gui_assistant, 167
 - ~gui_assistant, 168
 - gui_assistant, 168
 - jack_idle_connect, 168
 - keys, 168
 - lash_timeout_connect, 168
 - m_keys_perform, 168
 - quit, 168

- seq64::gui_assistant_gtk2, 169
 - ~gui_assistant_gtk2, 170
 - gui_assistant_gtk2, 170
 - jack_idle_connect, 170
 - lash_timeout_connect, 170
 - quit, 170
 - sm_internal_keys, 170
- seq64::gui_drawingarea_gtk2, 170
 - ~gui_drawingarea_gtk2, 174
 - clear_window, 175
 - current_x, 174
 - current_y, 174
 - draw_drawable, 180
 - draw_line, 175, 176
 - draw_line_on_pixmap, 175
 - draw_normal_rectangle_on_pixmap, 180
 - draw_rectangle, 177–179
 - draw_rectangle_on_pixmap, 179
 - drop_x, 174
 - drop_y, 174
 - force_draw, 174
 - gtk_drawarea_init, 181
 - gui_drawingarea_gtk2, 174
 - m_background, 181
 - m_current_x, 182
 - m_current_y, 182
 - m_drop_x, 182
 - m_drop_y, 182
 - m_foreground, 181
 - m_gc, 181
 - m_hadjust, 181
 - m_mainperf, 182
 - m_pixmap, 181
 - m_vadjust, 181
 - m_window, 181
 - m_window_x, 182
 - m_window_y, 182
 - on_realize, 181
 - operator=, 174
 - perf, 174
 - render_string, 177
 - render_string_on_pixmap, 177
 - scroll_hadjust, 180
 - scroll_hset, 181
 - scroll_vadjust, 180
 - scroll_vset, 181
 - set_current_drop_x, 181
 - set_current_drop_y, 181
 - set_line, 175
 - window_x, 174
 - window_y, 174
- seq64::gui_drawingarea_gtk2::rect, 414
 - height, 415
 - width, 415
 - x, 415
 - y, 415
- seq64::gui_palette_gtk2, 182
 - ~gui_palette_gtk2, 185
 - bg_color, 186
 - black, 185
 - blue, 186
 - Color, 184
 - dark_cyan, 186
 - dark_grey, 186
 - dark_orange, 186
 - fg_color, 186
 - green, 186
 - grey, 186
 - gui_palette_gtk2, 185
 - light_grey, 186
 - line_color, 185
 - m_bg_color, 187
 - m_black, 186
 - m_blue, 187
 - m_dk_cyan, 187
 - m_dk_grey, 186
 - m_dk_orange, 187
 - m_fg_color, 187
 - m_green, 187
 - m_grey, 186
 - m_line_color, 187
 - m_lt_grey, 186
 - m_orange, 186
 - m_progress_color, 187
 - m_red, 186
 - m_white, 186
 - m_yellow, 187
 - orange, 186
 - progress_color, 185
 - red, 186
 - white, 186
 - yellow, 186
- seq64::gui_window_gtk2, 187
 - ~gui_window_gtk2, 189
 - gui_window_gtk2, 189
 - is_realized, 190
 - m_is_realized, 191
 - m_mainperf, 190
 - m_redraw_period_ms, 191
 - m_window_x, 191
 - m_window_y, 191
 - on_realize, 190
 - perf, 189
 - quit, 190
 - redraw_period_ms, 190
 - scroll_hadjust, 190
 - scroll_hset, 190
 - scroll_vadjust, 190
 - scroll_vset, 190
- seq64::jack_assistant, 191
 - ~jack_assistant, 194
 - client, 198
 - client_name, 198
 - client_open, 199
 - client_uuid, 198
 - deinit, 195

- error_message, 199
- get_beat_width, 194
- get_beats_per_measure, 195
- get_beats_per_minute, 195
- get_jack_client_info, 200
- get_jack_pos, 198
- get_jack_tick, 198
- get_ppqn, 194
- info_message, 198
- init, 195
- is_master, 194
- is_running, 194
- jack_assistant, 194
- jack_process_callback, 202
- jack_session_callback, 203
- jack_shutdown_callback, 202
- jack_sync_callback, 202
- jack_timebase_callback, 203
- m_beat_width, 205
- m_beats_per_measure, 205
- m_beats_per_minute, 205
- m_jack_client, 204
- m_jack_client_name, 204
- m_jack_client_uuid, 204
- m_jack_frame_current, 204
- m_jack_frame_last, 204
- m_jack_master, 205
- m_jack_parent, 204
- m_jack_pos, 204
- m_jack_running, 205
- m_jack_tick, 205
- m_jack_transport_state, 204
- m_jack_transport_state_last, 204
- m_jsession_ev, 205
- m_ppqn, 205
- output, 197
- parent, 194
- position, 196
- session_event, 196
- set_beat_width, 194
- set_beats_per_measure, 195
- set_beats_per_minute, 195
- set_jack_running, 198
- set_position, 202
- set_ppqn, 198
- show_position, 200
- show_statuses, 200
- sm_status_pairs, 204
- start, 196
- stop, 196
- sync, 201
- seq64::jack_scratchpad, 205
 - js_clock_tick, 206
 - js_current_tick, 206
 - js_dumping, 206
 - js_init_clock, 206
 - js_jack_stopped, 206
 - js_looping, 206
 - js_playback_mode, 206
 - js_ticks_converted_last, 206
 - js_total_tick, 206
- seq64::jack_status_pair_t, 206
 - jf_bit, 206
 - jf_meaning, 206
- seq64::keybindentry, 206
 - events, 207
 - groups, 207
 - keybindentry, 207
 - location, 207
 - m_key, 208
 - m_perf, 208
 - m_slot, 208
 - m_type, 208
 - on_key_press_event, 208
 - options, 208
 - set, 208
 - type, 207
- seq64::keys_perform, 209
 - ~keys_perform, 214
 - at_bpm_dn, 220
 - at_bpm_up, 220
 - at_event_edit, 222
 - at_group_learn, 221
 - at_group_off, 221
 - at_group_on, 221
 - at_keep_queue, 221
 - at_pattern_edit, 222
 - at_pause, 222
 - at_queue, 221
 - at_replace, 220
 - at_screensets_dn, 221
 - at_screensets_up, 221
 - at_set_playing_screensets, 221
 - at_show_ui_sequence_key, 222
 - at_show_ui_sequence_number, 222
 - at_snapshot_1, 221
 - at_snapshot_2, 221
 - at_start, 222
 - at_stop, 222
 - bpm_dn, 215
 - bpm_up, 215
 - event_edit, 218
 - get_key_events, 219
 - get_key_events_rev, 219
 - get_key_groups, 219
 - get_key_groups_rev, 219
 - get_keys, 215
 - group_learn, 217
 - group_off, 217
 - group_on, 217
 - keep_queue, 215, 216
 - key_name, 219
 - keys_perform, 214
 - lookup_keyevent_key, 219
 - lookup_keyevent_seq, 219
 - lookup_keygroup_group, 219

- lookup_keygroup_key, [219](#)
- m_key_bpm_dn, [223](#)
- m_key_bpm_up, [223](#)
- m_key_event_edit, [224](#)
- m_key_events, [222](#)
- m_key_events_rev, [223](#)
- m_key_group_learn, [224](#)
- m_key_group_off, [223](#)
- m_key_group_on, [223](#)
- m_key_groups, [223](#)
- m_key_groups_rev, [223](#)
- m_key_keep_queue, [223](#)
- m_key_pattern_edit, [224](#)
- m_key_pause, [224](#)
- m_key_queue, [223](#)
- m_key_replace, [223](#)
- m_key_screenset_dn, [223](#)
- m_key_screenset_up, [223](#)
- m_key_set_playing_screenset, [223](#)
- m_key_show_ui_sequence_key, [222](#)
- m_key_show_ui_sequence_number, [222](#)
- m_key_snapshot_1, [223](#)
- m_key_snapshot_2, [223](#)
- m_key_start, [224](#)
- m_key_stop, [224](#)
- options, [222](#)
- optionsfile, [222](#)
- pattern_edit, [217](#), [218](#)
- pause, [217](#)
- perform, [222](#)
- queue, [215](#)
- replace, [215](#)
- RevSlotMap, [214](#)
- screenset_dn, [216](#)
- screenset_up, [216](#)
- set_all_key_events, [220](#)
- set_all_key_groups, [220](#)
- set_key_event, [220](#)
- set_key_group, [220](#)
- set_keys, [214](#)
- set_playing_screenset, [216](#)
- show_ui_sequence_key, [218](#)
- show_ui_sequence_number, [218](#)
- SlotMap, [214](#)
- snapshot_1, [216](#)
- snapshot_2, [216](#)
- start, [217](#)
- stop, [218](#)
- seq64::keys_perform_gtk2, [224](#)
- ~keys_perform_gtk2, [226](#)
- key_name, [226](#)
- keys_perform_gtk2, [226](#)
- set_all_key_events, [226](#)
- set_all_key_groups, [226](#)
- seq64::keys_perform_transfer, [226](#)
- kpt_bpm_dn, [227](#)
- kpt_bpm_up, [227](#)
- kpt_event_edit, [228](#)
- kpt_group_learn, [227](#)
- kpt_group_off, [227](#)
- kpt_group_on, [227](#)
- kpt_keep_queue, [227](#)
- kpt_pattern_edit, [228](#)
- kpt_pause, [228](#)
- kpt_queue, [227](#)
- kpt_replace, [227](#)
- kpt_screenset_dn, [227](#)
- kpt_screenset_up, [227](#)
- kpt_set_playing_screenset, [227](#)
- kpt_show_ui_sequence_key, [228](#)
- kpt_show_ui_sequence_number, [228](#)
- kpt_snapshot_1, [227](#)
- kpt_snapshot_2, [227](#)
- kpt_start, [228](#)
- kpt_stop, [228](#)
- seq64::keystroke, [228](#)
- is, [230](#)
- is_delete, [230](#)
- is_letter, [230](#)
- is_press, [230](#)
- key, [230](#)
- keystroke, [229](#)
- m_is_press, [231](#)
- m_key, [231](#)
- m_modifier, [231](#)
- mod_control, [231](#)
- mod_control_shift, [231](#)
- mod_super, [231](#)
- modifier, [230](#)
- operator=, [229](#)
- shift_lock, [230](#)
- seq64::lash, [231](#)
- handle_config, [233](#)
- handle_event, [233](#)
- init, [232](#)
- lash, [232](#)
- m_client, [233](#)
- m_is_lash_supported, [233](#)
- m_lash_args, [233](#)
- m_perform, [233](#)
- process_events, [232](#)
- set_alsa_client_id, [232](#)
- start, [232](#)
- seq64::maintime, [233](#)
- ~maintime, [236](#)
- idle_progress, [236](#)
- m_bar_width, [236](#)
- m_beat_width, [236](#)
- m_box_height, [237](#)
- m_box_less_pill, [237](#)
- m_box_width, [237](#)
- m_flash_height, [237](#)
- m_flash_width, [237](#)
- m_flash_x, [237](#)
- m_pill_width, [237](#)
- m_ppqn, [237](#)

- m_tick, 237
- maintime, 236
- mainwnd, 236
- on_expose_event, 236
- on_realize, 236
- operator=, 236
- seq64::mainwid, 238
 - ~mainwid, 241
 - calculate_base_sizes, 244
 - draw_marker_on_sequence, 242
 - draw_pixmap_on_window, 242
 - draw_sequence_on_pixmap, 243
 - draw_sequence_pixmap_on_window, 243
 - draw_sequences_on_pixmap, 243
 - fill_background_window, 242
 - m_button_down, 247
 - m_last_tick_x, 247
 - m_mainwid_border, 247
 - m_mainwid_spacing, 247
 - m_mainwid_x, 247
 - m_mainwid_y, 247
 - m_mainwnd_cols, 247
 - m_mainwnd_rows, 247
 - m_max_sets, 247
 - m_moving, 247
 - m_moving_seq, 247
 - m_old_seq, 247
 - m_progress_height, 248
 - m_screenset, 247
 - m_screenset_offset, 247
 - m_screenset_slots, 247
 - m_seqarea_seq_x, 247
 - m_seqarea_seq_y, 247
 - m_seqarea_x, 247
 - m_seqarea_y, 247
 - m_text_size_x, 247
 - m_text_size_y, 247
 - mainwid, 241
 - mainwnd, 246
 - on_button_press_event, 245
 - on_button_release_event, 245
 - on_expose_event, 245
 - on_focus_in_event, 246
 - on_focus_out_event, 246
 - on_motion_notify_event, 246
 - on_realize, 244
 - redraw, 242
 - reset, 242
 - select_fg_bg_colors, 244
 - seq_from_xy, 244
 - seq_set_and_edit, 242
 - seq_set_and_eventedit, 242
 - set_screenset, 241
 - timeout, 244
 - update_mainwid_sequences, 246
 - update_markers, 243
 - update_sequences_on_window, 242
 - valid_sequence, 243
- seq64::mainwnd, 248
 - ~mainwnd, 253
 - about_dialog, 254
 - adj_callback_bpm, 254
 - adj_callback_ss, 254
 - choose_file, 257
 - edit_callback_notepad, 254
 - enregister_perfedits, 256
 - file_exit, 256
 - file_import_dialog, 254
 - file_new, 256
 - file_open, 256
 - file_save, 256
 - file_save_as, 256
 - handle_signal, 254
 - install_signal_handlers, 257
 - is_save, 257
 - learn_toggle, 256
 - m_adjust_bpm, 259
 - m_adjust_load_offset, 259
 - m_adjust_ss, 259
 - m_button_learn, 259
 - m_button_perfedit, 259
 - m_button_play, 259
 - m_button_stop, 259
 - m_call_seq_edit, 260
 - m_call_seq_eventedit, 260
 - m_entry_notes, 259
 - m_image_play, 259
 - m_is_running, 259
 - m_main_cursor, 259
 - m_main_time, 258
 - m_main_wid, 258
 - m_menu_file, 258
 - m_menu_help, 258
 - m_menu_view, 258
 - m_menubar, 258
 - m_options, 258
 - m_perf_edit, 258
 - m_perf_edit_2, 258
 - m_ppqn, 258
 - m_sigpipe, 258
 - m_spinbutton_bpm, 259
 - m_spinbutton_load_offset, 259
 - m_spinbutton_ss, 259
 - m_timeout_connect, 260
 - m_tooltips, 258
 - mainwnd, 253
 - new_file, 256
 - on_delete_event, 257
 - on_grouplearnchange, 257
 - on_key_press_event, 257
 - on_key_release_event, 257
 - open_file, 253
 - open_performance_edit, 256
 - open_performance_edit_2, 256
 - options_dialog, 254
 - pause_playing, 255

- ppqn, 254
- query_save_changes, 257
- save_file, 256
- sequence_key, 256
- set_image, 255
- signal_action, 257
- start_playing, 255
- stop_playing, 255
- timer_callback, 255
- toLower, 256
- toggle_playing, 255
- update_window_title, 256
- seq64::mastermidibus, 260
 - ~mastermidibus, 262
 - clock, 264
 - continue_from, 265
 - flush, 264
 - get_alsa_seq, 263
 - get_beats_per_minute, 263
 - get_clock, 267
 - get_input, 267
 - get_midi_event, 265
 - get_midi_in_bus_name, 264
 - get_midi_out_bus_name, 264
 - get_num_in_buses, 263
 - get_num_out_buses, 263
 - get_ppqn, 264
 - get_sequence, 266
 - init, 263
 - init_clock, 265
 - is_dumping, 266
 - is_more_input, 265
 - m_alsa_seq, 268
 - m_beats_per_minute, 268
 - m_bus_announce, 268
 - m_buses_in, 268
 - m_buses_in_active, 268
 - m_buses_in_init, 268
 - m_buses_out, 268
 - m_buses_out_active, 268
 - m_buses_out_init, 268
 - m_dumping_input, 269
 - m_init_clock, 268
 - m_init_input, 268
 - m_mutex, 269
 - m_num_in_buses, 268
 - m_num_out_buses, 268
 - m_num_poll_descriptors, 268
 - m_poll_descriptors, 269
 - m_ppqn, 268
 - m_queue, 268
 - m_seq, 269
 - mastermidibus, 262
 - play, 266
 - poll_for_midi, 265
 - port_exit, 266
 - port_start, 266
 - print, 264
 - set_beats_per_minute, 263
 - set_clock, 267
 - set_input, 267
 - set_ppqn, 263
 - set_sequence_input, 265
 - start, 264
 - stop, 264
 - sysex, 266
- seq64::midi_container, 269
 - ~midi_container, 271
 - add_long, 272
 - add_variable, 272
 - done, 271
 - fill, 271
 - get, 272
 - m_position_for_get, 272
 - m_sequence, 272
 - midi_container, 270
 - position, 272
 - position_increment, 272
 - position_reset, 272
 - put, 271
 - size, 271
- seq64::midi_control, 272
 - active, 274
 - data, 274
 - in_range, 275
 - inverse_active, 274
 - m_active, 275
 - m_data, 275
 - m_inverse_active, 275
 - m_max_value, 275
 - m_min_value, 275
 - m_status, 275
 - match, 275
 - max_value, 274
 - midi_control, 274
 - min_value, 274
 - set, 274
 - status, 274
- seq64::midi_list, 275
 - ~midi_list, 277
 - CharList, 277
 - done, 277
 - get, 277
 - m_char_list, 278
 - midi_list, 277
 - put, 277
 - size, 277
- seq64::midi_measures, 278
 - beats, 279
 - divisions, 279
 - m_beats, 279
 - m_divisions, 279
 - m_measures, 279
 - measures, 279
 - midi_measures, 278
- seq64::midi_splitter, 280

- [~midi_splitter](#), 281
 - [count](#), 282
 - [increment](#), 281
 - [initialize](#), 281
 - [log_main_sequence](#), 281
 - [m_ppqn](#), 283
 - [m_smf0_channels](#), 283
 - [m_smf0_channels_count](#), 283
 - [m_smf0_main_sequence](#), 283
 - [m_smf0_seq_number](#), 283
 - [m_use_default_ppqn](#), 283
 - [midi_splitter](#), 281
 - [ppqn](#), 282
 - [split](#), 282
 - [split_channel](#), 282
- [seq64::midi_timing](#), 283
 - [beat_width](#), 285
 - [beats_per_measure](#), 284
 - [beats_per_minute](#), 284
 - [m_beat_width](#), 285
 - [m_beats_per_measure](#), 285
 - [m_beats_per_minute](#), 285
 - [m_ppqn](#), 285
 - [midi_timing](#), 284
 - [ppqn](#), 285
- [seq64::midi_vector](#), 286
 - [~midi_vector](#), 287
 - [CharVector](#), 287
 - [done](#), 287
 - [get](#), 288
 - [m_char_vector](#), 288
 - [midi_vector](#), 287
 - [put](#), 287
 - [size](#), 287
- [seq64::midibus](#), 288
 - [~midibus](#), 291
 - [clock](#), 292
 - [continue_from](#), 292
 - [deinit_in](#), 291
 - [flush](#), 293
 - [get_client](#), 293
 - [get_clock](#), 293
 - [get_clock_mod](#), 293
 - [get_id](#), 292
 - [get_input](#), 293
 - [get_name](#), 292
 - [get_port](#), 293
 - [init_clock](#), 292
 - [init_in](#), 291
 - [init_in_sub](#), 291
 - [init_out](#), 291
 - [init_out_sub](#), 291
 - [m_clock_mod](#), 293
 - [m_clock_type](#), 294
 - [m_dest_addr_client](#), 294
 - [m_dest_addr_port](#), 294
 - [m_id](#), 293
 - [m_inputting](#), 294
 - [m_lasttick](#), 294
 - [m_local_addr_client](#), 294
 - [m_local_addr_port](#), 294
 - [m_mutex](#), 294
 - [m_name](#), 294
 - [m_ppqn](#), 294
 - [m_queue](#), 294
 - [m_seq](#), 294
 - [mastermidibus](#), 293
 - [midibus](#), 290
 - [play](#), 292
 - [print](#), 291
 - [set_clock](#), 292
 - [set_clock_mod](#), 293
 - [set_input](#), 293
 - [start](#), 292
 - [stop](#), 292
 - [sysex](#), 292
- [seq64::midifile](#), 294
 - [~midifile](#), 297
 - [add_trigger](#), 301
 - [checklen](#), 301
 - [errdump](#), 307
 - [error_is_fatal](#), 299
 - [error_message](#), 299
 - [is_sysex_special_id](#), 308
 - [m_char_list](#), 309
 - [m_data](#), 309
 - [m_disable_reported](#), 308
 - [m_error_is_fatal](#), 308
 - [m_error_message](#), 308
 - [m_file_size](#), 308
 - [m_global_bgsequence](#), 309
 - [m_name](#), 309
 - [m_new_format](#), 309
 - [m_pos](#), 308
 - [m_ppqn](#), 309
 - [m_smf0_splitter](#), 309
 - [m_use_default_ppqn](#), 309
 - [midifile](#), 297
 - [parse](#), 297
 - [parse_prop_header](#), 299
 - [parse_proprietary_track](#), 300
 - [parse_smf_0](#), 299
 - [parse_smf_1](#), 299
 - [pow2](#), 301
 - [ppqn](#), 299
 - [prop_item_size](#), 307
 - [read_byte](#), 302
 - [read_byte_array](#), 303
 - [read_long](#), 302
 - [read_seq_number](#), 304
 - [read_short](#), 302
 - [read_track_name](#), 304
 - [read_varinum](#), 302
 - [seq_number_size](#), 308
 - [track_end_size](#), 308
 - [track_name_size](#), 307

- varinum_size, 306
- write, 298
- write_byte, 303
- write_long, 302
- write_prop_header, 305
- write_proprietary_track, 306
- write_seq_number, 304
- write_short, 303
- write_track_end, 305
- write_track_name, 304
- write_varinum, 303
- seq64::mutex, 310
 - lock, 311
 - m_mutex_lock, 311
 - mutex, 311
 - sm_recursive_mutex, 311
 - unlock, 311
- seq64::options, 311
 - add_jack_sync_page, 314
 - add_keyboard_page, 314
 - add_midi_clock_page, 314
 - add_midi_input_page, 314
 - add_mouse_page, 314
 - button, 313
 - clock_callback_mod, 313
 - clock_callback_off, 313
 - clock_callback_on, 313
 - clock_mod_callback, 313
 - e_jack_connect, 313
 - e_jack_disconnect, 313
 - e_jack_master, 313
 - e_jack_master_cond, 313
 - e_jack_start_mode_live, 313
 - e_jack_start_mode_song, 313
 - e_jack_transport, 313
 - input_callback, 313
 - lash_support_callback, 314
 - m_button_jack_connect, 314
 - m_button_jack_disconnect, 314
 - m_button_jack_master, 314
 - m_button_jack_master_cond, 314
 - m_button_jack_transport, 314
 - m_button_ok, 314
 - m_mainperf, 314
 - m_notebook, 314
 - m_tooltips, 314
 - mouse_fruity_callback, 313
 - mouse_mod4_callback, 314
 - mouse_seq24_callback, 313
 - options, 313
 - perf, 313
 - transport_callback, 313
- seq64::optionsfile, 315
 - ~optionsfile, 316
 - error_message, 318
 - optionsfile, 316
 - parse, 316
 - write, 317
- seq64::perfedit, 318
 - ~perfedit, 323
 - collapse, 325
 - copy, 325
 - draw_sequences, 325
 - enqueue_draw, 323
 - enregister_peer, 324
 - expand, 325
 - grow, 324
 - init_before_show, 323
 - m_bpm, 328
 - m_button_bpm, 328
 - m_button_bw, 328
 - m_button_collapse, 328
 - m_button_copy, 328
 - m_button_expand, 328
 - m_button_grow, 328
 - m_button_loop, 327
 - m_button_play, 327
 - m_button_snap, 327
 - m_button_stop, 327
 - m_button_undo, 328
 - m_bw, 328
 - m_entry_bpm, 328
 - m_entry_bw, 328
 - m_entry_snap, 327
 - m_hadjust, 327
 - m_hbox, 328
 - m_hlbox, 328
 - m_hscroll, 327
 - m_image_play, 327
 - m_is_running, 329
 - m_menu_bpm, 328
 - m_menu_bw, 328
 - m_menu_snap, 327
 - m_peer_perfedit, 327
 - m_perfnames, 327
 - m_perfroll, 327
 - m_perftime, 327
 - m_ppqn, 328
 - m_snap, 328
 - m_standard_bpm, 329
 - m_table, 327
 - m_tooltips, 328
 - m_vadjust, 327
 - m_vscroll, 327
 - on_delete_event, 326
 - on_key_press_event, 326
 - on_realize, 326
 - pause_playing, 326
 - perfedit, 323
 - popup_menu, 325
 - set_beat_width, 324
 - set_beats_per_bar, 324
 - set_guides, 324
 - set_image, 325
 - set_looped, 325
 - set_snap, 324

- set_zoom, [324](#)
- start_playing, [326](#)
- stop_playing, [326](#)
- timeout, [325](#)
- toggle_playing, [326](#)
- undo, [325](#)
- update_perfedit_sequences, [326](#)
- zoom_check, [323](#)
- seq64::perfnames, [329](#)
 - ~perfnames, [331](#)
 - change_vert, [332](#)
 - convert_y, [332](#)
 - draw_sequence, [332](#)
 - draw_sequences, [332](#)
 - enqueue_draw, [332](#)
 - m_char_w, [335](#)
 - m_namebox_w, [336](#)
 - m_names_chars, [335](#)
 - m_names_x, [336](#)
 - m_names_y, [336](#)
 - m_parent, [335](#)
 - m_seqs_in_set, [336](#)
 - m_sequence_active, [336](#)
 - m_sequence_max, [336](#)
 - m_sequence_offset, [336](#)
 - m_setbox_w, [336](#)
 - m_xy_offset, [336](#)
 - on_button_press_event, [333](#)
 - on_button_release_event, [333](#)
 - on_expose_event, [333](#)
 - on_realize, [333](#)
 - on_scroll_event, [335](#)
 - on_size_allocate, [335](#)
 - perfedit, [335](#)
 - perfnames, [331](#)
 - redraw, [332](#)
 - redraw_dirty_sequences, [332](#)
- seq64::perform, [336](#)
 - ~perform, [346](#)
 - add_sequence, [349](#)
 - all_notes_off, [356](#)
 - any_group_unmutes, [355](#)
 - clamp_track, [375](#)
 - clear_all, [348](#)
 - clear_sequence_triggers, [350](#)
 - collapse, [352](#)
 - copy, [352](#)
 - copy_triggers, [352](#)
 - current_screen_set_notepad, [354](#)
 - decrement_beats_per_minute, [368](#)
 - decrement_screenset, [368](#)
 - deinit_jack, [371](#)
 - delete_sequence, [350](#)
 - enregister, [348](#)
 - expand, [352](#)
 - finish, [350](#)
 - get_beat_width, [348](#)
 - get_beats_per_bar, [348](#)
 - get_beats_per_minute, [359](#)
 - get_group_mute_state, [363](#)
 - get_jack_tick, [351](#)
 - get_key_events, [364](#)
 - get_key_events_rev, [364](#)
 - get_key_groups, [364](#)
 - get_key_groups_rev, [364](#)
 - get_left_tick, [351](#)
 - get_max_trigger, [352](#)
 - get_offset, [364](#)
 - get_playing_screenset, [355](#)
 - get_right_tick, [351](#)
 - get_screen_set_notepad, [353](#)
 - get_screenset, [354](#)
 - get_sequence, [358](#)
 - get_tick, [350](#)
 - gui, [348](#)
 - handle_midi_control, [353](#)
 - highlight, [368](#)
 - increment_beats_per_minute, [368](#)
 - increment_screenset, [368](#)
 - init_jack, [371](#)
 - inner_start, [374](#)
 - inner_stop, [374](#)
 - input_func, [363](#)
 - install_sequence, [374](#)
 - is_active, [358](#)
 - is_control_status, [347](#)
 - is_dirty_edit, [357](#)
 - is_dirty_main, [357](#)
 - is_dirty_names, [358](#)
 - is_dirty_perf, [357](#)
 - is_edit_sequence, [347](#)
 - is_group_learning, [356](#)
 - is_jack_running, [348](#)
 - is_midi_control_valid, [372](#)
 - is_modified, [347](#), [372](#)
 - is_mseq_valid, [373](#)
 - is_pausable, [348](#)
 - is_paused, [348](#)
 - is_running, [348](#)
 - is_screenset_valid, [372](#)
 - is_seq_valid, [373](#)
 - is_sequence_in_edit, [350](#)
 - is_smf_0, [369](#)
 - jack_assistant, [376](#)
 - jack_sync_callback, [376](#)
 - key_name, [364](#)
 - keybindentry, [376](#)
 - keys, [348](#)
 - launch, [349](#)
 - launch_input_thread, [371](#)
 - launch_output_thread, [371](#)
 - learn_toggle, [368](#)
 - lookup_keyevent_key, [365](#)
 - lookup_keyevent_seq, [365](#)
 - lookup_keygroup_group, [365](#)
 - lookup_keygroup_key, [365](#)

`m_beat_width`, 379
`m_beats_per_bar`, 379
`m_condition_var`, 381
`m_control_status`, 380
`m_edit_sequence`, 381
`m_gui_support`, 376
`m_in_thread`, 378
`m_in_thread_launched`, 378
`m_inputting`, 378
`m_is_modified`, 381
`m_is_paused`, 380
`m_jack_asst`, 381
`m_jack_tick`, 379
`m_left_tick`, 379
`m_looping`, 378
`m_master_bus`, 378
`m_max_sets`, 380
`m_midi_cc_off`, 380
`m_midi_cc_on`, 380
`m_midi_cc_toggle`, 380
`m_midiclockpos`, 380
`m_midiclockrunning`, 380
`m_midiclocktick`, 380
`m_mode_group`, 377
`m_mode_group_learn`, 377
`m_mute_group`, 376
`m_mute_group_selected`, 377
`m_notify`, 381
`m_offset`, 380
`m_one_measure`, 379
`m_out_thread`, 378
`m_out_thread_launched`, 378
`m_outputting`, 378
`m_playback_mode`, 378
`m_playing_screen`, 377
`m_playscreen_offset`, 377
`m_ppqn`, 378
`m_right_tick`, 379
`m_running`, 378
`m_screen_set_notepad`, 380
`m_screenset`, 380
`m_seqs`, 377
`m_seqs_active`, 377
`m_seqs_in_set`, 380
`m_sequence_count`, 380
`m_sequence_max`, 381
`m_sequence_state`, 378
`m_starting_tick`, 379
`m_tick`, 379
`m_tracks_mute_state`, 376
`m_usemidiclock`, 380
`m_was_active_edit`, 377
`m_was_active_main`, 377
`m_was_active_names`, 378
`m_was_active_perf`, 378
`mainwnd_key_event`, 370
`master_bus`, 348
`max_active_set`, 371
`midi_control_off`, 353
`midi_control_on`, 353
`midi_control_toggle`, 352
`midifile`, 376
`modify`, 347
`move_triggers`, 351
`mute_all_tracks`, 362
`mute_group_offset`, 373
`mute_group_tracks`, 355
`mute_screenset`, 362
`new_sequence`, 349
`off_sequences`, 356
`options`, 376
`optionsfile`, 376
`output_func`, 362
`pause_key`, 368
`pause_playing`, 367
`perform`, 346
`perffroll_key_event`, 370
`play`, 359
`playback_key_event`, 370
`pop_trigger_undo`, 352
`position_jack`, 356
`print_triggers`, 350
`push_trigger_undo`, 352
`reset_sequences`, 358
`restore_playing_state`, 364
`save_playing_state`, 364
`select_and_mute_group`, 355
`select_group_mute`, 355
`seq_in_playing_screen`, 372
`sequence_count`, 347
`sequence_key`, 369
`sequence_label`, 369
`sequence_max`, 347
`sequence_playing_change`, 361
`sequence_playing_off`, 362
`sequence_playing_on`, 361
`sequence_playing_toggle`, 361
`set_active`, 356
`set_all_key_events`, 375
`set_all_key_groups`, 375
`set_and_copy_mute_group`, 356
`set_beat_width`, 348
`set_beats_per_bar`, 348
`set_beats_per_minute`, 359
`set_edit_sequence`, 347
`set_group_mute_state`, 363
`set_input_bus`, 369
`set_jack_tick`, 351
`set_key_event`, 375
`set_key_group`, 375
`set_left_tick`, 351
`set_looping`, 359
`set_mode_group_learn`, 355
`set_mode_group_mute`, 355
`set_offset`, 363
`set_orig_ticks`, 359

- set_playback_mode, 373
- set_playing_screenset, 354
- set_right_tick, 351
- set_running, 372
- set_screen_set_notepad, 354
- set_screenset, 354
- set_sequence_control_status, 361
- set_start_tick, 351
- set_was_active, 357
- show_ui_sequence_key, 364
- show_ui_sequence_number, 364
- sm_mc_dummy, 376
- split_trigger, 352
- start, 356
- start_jack, 356
- start_key, 368
- start_playing, 367
- stop, 356
- stop_jack, 356
- stop_key, 368
- stop_playing, 367
- toggle_all_tracks, 362
- unset_edit_sequence, 347
- unset_mode_group_learn, 355
- unset_mode_group_mute, 355
- unset_sequence_control_status, 361
- seq64::performcallback, 381
 - on_grouplearnchange, 383
- seq64::perfull, 383
 - ~perfull, 388
 - change_horz, 390
 - change_vert, 390
 - convert_x, 390
 - convert_xy, 389
 - draw_all, 389
 - draw_background_on, 390
 - draw_drawable_row, 390
 - draw_progress, 389
 - draw_sequence_on, 390
 - enqueue_draw, 390
 - fill_background_pixmap, 388
 - follow_progress, 389
 - FruityPerfInput, 394
 - horizontal_adjust, 391
 - horizontal_set, 392
 - increment_size, 389
 - init_before_show, 388
 - m_4bar_offset, 395
 - m_background_x, 395
 - m_beat_length, 395
 - m_divs_per_beat, 395
 - m_drop_sequence, 396
 - m_drop_tick, 396
 - m_drop_tick_trigger_offset, 396
 - m_fruity_interaction, 396
 - m_grow_direction, 397
 - m_growing, 397
 - m_h_page_increment, 394
 - m_measure_length, 395
 - m_moving, 397
 - m_names_y, 395
 - m_old_progress_ticks, 395
 - m_page_factor, 395
 - m_parent, 394
 - m_perf_scale_x, 395
 - m_ppqn, 395
 - m_roll_length_ticks, 396
 - m_seq24_interaction, 396
 - m_sequence_active, 396
 - m_sequence_max, 396
 - m_sequence_offset, 396
 - m_size_box_w, 395
 - m_snap, 395
 - m_ticks_per_bar, 395
 - m_v_page_increment, 394
 - m_zoom, 395
 - on_button_press_event, 393
 - on_button_release_event, 393
 - on_expose_event, 392
 - on_focus_in_event, 393
 - on_focus_out_event, 393
 - on_key_press_event, 393
 - on_motion_notify_event, 393
 - on_realize, 392
 - on_scroll_event, 393
 - on_size_allocate, 393
 - on_size_request, 394
 - perfedit, 394
 - perfull, 388
 - redraw_dirty_sequences, 389
 - redraw_progress, 389
 - Seq24PerfInput, 394
 - set_guides, 388
 - set_ppqn, 389
 - set_zoom, 390
 - snap_x, 390
 - split_trigger, 390
 - update_sizes, 388
 - vertical_adjust, 392
 - vertical_set, 392
- seq64::perftime, 397
 - ~perftime, 401
 - change_horz, 401
 - draw_background, 401
 - draw_pixmap_on_window, 403
 - draw_progress_on_window, 401
 - enqueue_draw, 401
 - idle_progress, 403
 - increment_size, 401
 - key_press_event, 404
 - m_4bar_offset, 404
 - m_left_marker_tick, 405
 - m_measure_length, 405
 - m_parent, 404
 - m_perf_scale_x, 405
 - m_ppqn, 405

- m_right_marker_tick, 405
- m_snap, 405
- m_tick_offset, 404
- m_timearea_y, 405
- on_button_press_event, 403
- on_button_release_event, 404
- on_expose_event, 403
- on_realize, 403
- on_size_allocate, 404
- perfdit, 404
- perftime, 400
- pixel_to_tick, 402
- reset, 401
- set_guides, 401
- set_ppqn, 402
- set_scale, 401
- set_zoom, 401
- tick_offset, 402
- tick_to_pixel, 402
- update_pixmap, 403
- update_sizes, 402
- seq64::rc_settings, 405
 - allow_mod4_mode, 409
 - auto_option_save, 409
 - config_directory, 411, 412
 - config_filename, 411, 412
 - config_filename_alt, 411, 412
 - config_filespec, 409
 - device_ignore, 410
 - device_ignore_num, 410, 411
 - filename, 411
 - home_config_directory, 412
 - interaction_method, 410, 411
 - is_pattern_playing, 410
 - jack_session_uuid, 411
 - jack_start_mode, 410
 - lash_support, 409
 - last_used_dir, 411
 - legacy_format, 409
 - m_allow_mod4_mode, 413
 - m_auto_option_save, 413
 - m_config_directory, 414
 - m_config_filename, 414
 - m_config_filename_alt, 414
 - m_device_ignore, 413
 - m_device_ignore_num, 414
 - m_filename, 414
 - m_interaction_method, 414
 - m_is_pattern_playing, 413
 - m_jack_session_uuid, 414
 - m_jack_start_mode, 413
 - m_lash_support, 413
 - m_last_used_dir, 414
 - m_legacy_format, 413
 - m_manual_alsa_ports, 413
 - m_pass_sysex, 413
 - m_print_keys, 413
 - m_priority, 413
 - m_reveal_alsa_ports, 413
 - m_show_midi, 413
 - m_stats, 413
 - m_user_filename, 414
 - m_user_filename_alt, 414
 - m_with_jack_master, 413
 - m_with_jack_master_cond, 413
 - m_with_jack_transport, 413
 - manual_alsa_ports, 410
 - operator=, 408
 - pass_sysex, 410
 - print_keys, 410
 - priority, 409
 - rc_settings, 408
 - reveal_alsa_ports, 410
 - set_config_files, 412
 - set_defaults, 409
 - show_midi, 409
 - stats, 409
 - user_filename, 411, 412
 - user_filename_alt, 411, 412
 - user_filespec, 409
 - with_jack, 410
 - with_jack_master, 410
 - with_jack_master_cond, 410
 - with_jack_transport, 410
- seq64::rect, 414
 - height, 414
 - width, 414
 - x, 414
 - y, 414
- seq64::seqdata, 420
 - ~seqdata, 423
 - change_horz, 424
 - convert_x, 424
 - draw_events_on, 424
 - draw_events_on_pixmap, 425
 - draw_line_on_window, 424
 - draw_pixmap_on_window, 425
 - idle_redraw, 423
 - m_cc, 427
 - m_dragging, 428
 - m_number_h, 427
 - m_number_offset_y, 427
 - m_number_w, 427
 - m_numbers, 427
 - m_old, 427
 - m_scroll_offset_ticks, 427
 - m_scroll_offset_x, 427
 - m_seq, 427
 - m_status, 427
 - m_zoom, 427
 - on_button_press_event, 425
 - on_button_release_event, 425
 - on_expose_event, 425
 - on_leave_notify_event, 426
 - on_motion_notify_event, 426
 - on_realize, 425

- on_scroll_event, 426
- on_size_allocate, 426
- redraw, 423
- render_number, 424
- reset, 423
- seqdata, 422
- sequevent, 427
- seqroll, 427
- set_data_type, 423
- set_zoom, 423
- update_pixmap, 424
- update_sizes, 423
- xy_to_rect, 424
- seq64::seqedit, 428
 - ~seqedit, 434
 - apply_length, 437
 - change_focus, 440
 - create_menu_image, 439
 - create_menus, 438
 - do_action, 439
 - fill_top_bar, 438
 - get_measures, 437
 - handle_close, 440
 - horizontal_adjust, 436
 - horizontal_set, 436
 - m_bgsequence, 442
 - m_button_bpm, 445
 - m_button_bus, 444
 - m_button_bw, 445
 - m_button_channel, 444
 - m_button_data, 445
 - m_button_key, 445
 - m_button_length, 445
 - m_button_note_length, 444
 - m_button_quantize, 444
 - m_button_rec_vol, 445
 - m_button_redo, 444
 - m_button_scale, 445
 - m_button_sequence, 444
 - m_button_snap, 444
 - m_button_tools, 444
 - m_button_undo, 444
 - m_button_zoom, 445
 - m_editing_cc, 445
 - m_editing_status, 445
 - m_entry_bpm, 445
 - m_entry_bus, 444
 - m_entry_bw, 445
 - m_entry_channel, 444
 - m_entry_data, 445
 - m_entry_key, 445
 - m_entry_length, 445
 - m_entry_name, 445
 - m_entry_note_length, 444
 - m_entry_scale, 445
 - m_entry_sequence, 444
 - m_entry_snap, 444
 - m_entry_zoom, 445
 - m_hadjust, 443
 - m_have_focus, 446
 - m_hbox, 444
 - m_hbox2, 444
 - m_hscroll_new, 443
 - m_initial_note_length, 442
 - m_initial_snap, 441
 - m_initial_zoom, 442
 - m_key, 442
 - m_measures, 442
 - m_menu_bpm, 443
 - m_menu_bw, 443
 - m_menu_data, 443
 - m_menu_key, 443
 - m_menu_length, 443
 - m_menu_midibus, 443
 - m_menu_midich, 443
 - m_menu_note_length, 443
 - m_menu_rec_vol, 443
 - m_menu_scale, 443
 - m_menu_sequences, 443
 - m_menu_snap, 443
 - m_menu_tools, 442
 - m_menu_zoom, 443
 - m_menubar, 442
 - m_note_length, 442
 - m_ppqn, 442
 - m_scale, 442
 - m_seq, 442
 - m_seqdata_wid, 443
 - m_sequevent_wid, 444
 - m_seqkeys_wid, 443
 - m_seqroll_wid, 444
 - m_seqtime_wid, 443
 - m_snap, 442
 - m_table, 444
 - m_toggle_play, 445
 - m_toggle_q_rec, 445
 - m_toggle_record, 445
 - m_toggle_thru, 445
 - m_tooltips, 445
 - m_vadjust, 443
 - m_vbox, 444
 - m_vscroll_new, 443
 - m_zoom, 442
 - mouse_action, 440
 - name_change_callback, 437
 - on_delete_event, 440
 - on_focus_in_event, 440
 - on_focus_out_event, 440
 - on_key_press_event, 441
 - on_realize, 440
 - on_scroll_event, 440
 - on_set_focus, 440
 - play_change_callback, 438
 - popup_event_menu, 439
 - popup_menu, 439
 - popup_midibus_menu, 439

- popup_midich_menu, 439
- popup_sequence_menu, 439
- popup_tool_menu, 439
- q_rec_change_callback, 438
- record_change_callback, 438
- redo_callback, 438
- seqedit, 434
- seqmenu, 441
- set_background_sequence, 437
- set_beat_width, 435
- set_beats_per_bar, 435
- set_data_type, 438
- set_key, 437
- set_measures, 436
- set_midi_bus, 437
- set_midi_channel, 437
- set_note_length, 435
- set_rec_vol, 436
- set_scale, 437
- set_snap, 435
- set_zoom, 435
- thru_change_callback, 438
- timeout, 439
- undo_callback, 438
- update_all_windows, 438
- vertical_adjust, 436
- vertical_set, 436
- seq64::seqevent, 446
 - ~seqevent, 449
 - change_horz, 452
 - convert_t, 452
 - convert_x, 452
 - draw_background, 450
 - draw_events_on, 451
 - draw_events_on_pixmap, 450
 - draw_pixmap_on_window, 450
 - draw_selection_on_window, 450
 - drop_event, 451
 - force_draw, 451
 - FruitySeqEventInput, 455
 - idle_redraw, 451
 - m_cc, 456
 - m_fruity_interaction, 455
 - m_growing, 456
 - m_move_snap_offset_x, 456
 - m_moving, 456
 - m_moving_init, 456
 - m_old, 455
 - m_painting, 456
 - m_paste, 456
 - m_ppqn, 455
 - m_scroll_offset_ticks, 455
 - m_scroll_offset_x, 455
 - m_selected, 455
 - m_selecting, 456
 - m_seq, 455
 - m_seq24_interaction, 455
 - m_seqdata_wid, 455
 - m_snap, 455
 - m_status, 456
 - m_zoom, 455
 - on_button_press_event, 453
 - on_button_release_event, 453
 - on_expose_event, 453
 - on_focus_in_event, 454
 - on_focus_out_event, 454
 - on_key_press_event, 454
 - on_motion_notify_event, 453
 - on_realize, 453
 - on_size_allocate, 454
 - redraw, 449
 - reset, 449
 - Seq24SeqEventInput, 455
 - seqevent, 449
 - set_data_type, 450
 - set_snap, 450
 - set_zoom, 450
 - snap_x, 452
 - snap_y, 452
 - start_paste, 451
 - update_pixmap, 451
 - update_sizes, 450
 - x_to_w, 451
- seq64::seqkeys, 456
 - ~seqkeys, 459
 - change_vert, 461
 - convert_y, 460
 - draw_area, 460
 - draw_key, 461
 - force_draw, 460
 - is_black_key, 461
 - m_hint_key, 463
 - m_hint_state, 463
 - m_key, 464
 - m_keying, 463
 - m_keying_note, 463
 - m_scale, 464
 - m_scroll_offset_key, 463
 - m_scroll_offset_y, 463
 - m_seq, 463
 - m_show_octave_letters, 464
 - on_button_press_event, 461
 - on_button_release_event, 462
 - on_enter_notify_event, 462
 - on_expose_event, 461
 - on_leave_notify_event, 462
 - on_motion_notify_event, 462
 - on_realize, 461
 - on_scroll_event, 463
 - on_size_allocate, 463
 - reset, 461
 - seqkeys, 459
 - set_hint_key, 460
 - set_hint_state, 460
 - set_key, 459
 - set_scale, 459

- update_pixmap, 460
- update_sizes, 461
- seq64::seqmenu, 464
 - ~seqmenu, 468
 - current_seq, 468
 - delete_current_sequence, 468
 - get_current_sequence, 468
 - get_sequence, 468
 - is_current_seq_active, 468
 - is_current_seq_in_edit, 468
 - is_edit_sequence, 468
 - is_modified, 468
 - m_clipboard, 471
 - m_current_seq, 471
 - m_eventedit, 471
 - m_mainperf, 471
 - m_menu, 471
 - m_modified, 471
 - m_seqedit, 471
 - mute_all_tracks, 470
 - new_current_sequence, 468
 - new_sequence, 468
 - on_realize, 471
 - popup_menu, 468
 - redraw, 471
 - seq_clear_perf, 470
 - seq_copy, 469
 - seq_cut, 470
 - seq_edit, 468
 - seq_event_edit, 469
 - seq_new, 469
 - seq_paste, 470
 - seq_set_and_edit, 469
 - seq_set_and_eventedit, 469
 - seqmenu, 467
 - set_bus_and_midi_channel, 470
 - set_edit_sequence, 468
 - set_transposable, 470
 - toggle_all_tracks, 470
 - toggle_current_sequence, 468
 - unmute_all_tracks, 470
 - unset_edit_sequence, 468
- seq64::seqroll, 471
 - ~seqroll, 478
 - add_note, 478
 - adding, 487
 - align_selection, 485
 - button_press, 485
 - button_press_initial, 485
 - button_release, 485
 - change_horz, 483
 - change_vert, 483
 - clear_flags, 486
 - clear_old, 486
 - clear_selected, 486
 - complete_paste, 481
 - convert_sel_box_to_rect, 483
 - convert_tn, 482
 - convert_tn_box_to_rect, 482
 - convert_xy, 482
 - draw_background_on_pixmap, 479
 - draw_events_on, 483
 - draw_events_on_pixmap, 479
 - draw_progress_on_window, 480
 - draw_selection_on_window, 479
 - drop_action, 487
 - follow_progress, 481
 - force_draw, 481
 - FruitySeqRollInput, 490
 - get_selected_box, 483
 - grow_selected_notes, 484
 - growing, 487
 - horizontal_adjust, 481
 - idle_progress, 483
 - idle_redraw, 483
 - m_adding, 491
 - m_background_sequence, 492
 - m_cc, 492
 - m_drawing_background_seq, 492
 - m_fruity_interaction, 491
 - m_growing, 492
 - m_horizontal_adjust, 491
 - m_is_drag_pasting, 492
 - m_is_drag_pasting_start, 492
 - m_justselected_one, 492
 - m_key, 491
 - m_move_delta_x, 492
 - m_move_delta_y, 492
 - m_move_snap_offset_x, 492
 - m_moving, 492
 - m_moving_init, 492
 - m_note_length, 491
 - m_old, 491
 - m_painting, 492
 - m_paste, 492
 - m_pos, 491
 - m_ppqn, 491
 - m_progress_x, 492
 - m_scale, 491
 - m_scroll_offset_key, 492
 - m_scroll_offset_ticks, 492
 - m_scroll_offset_x, 492
 - m_scroll_offset_y, 492
 - m_selected, 491
 - m_selecting, 491
 - m_seq, 491
 - m_seqkeys_wid, 491
 - m_snap, 491
 - m_status, 492
 - m_vertical_adjust, 491
 - m_zoom, 491
 - motion_notify, 486
 - move_selected_notes, 484
 - move_selection_box, 484
 - normal_action, 487
 - note_off_length, 478

- on_button_press_event, 488
- on_button_release_event, 488
- on_enter_notify_event, 490
- on_expose_event, 488
- on_focus_in_event, 489
- on_focus_out_event, 489
- on_key_press_event, 489
- on_leave_notify_event, 490
- on_motion_notify_event, 488
- on_realize, 488
- on_scroll_event, 489
- on_size_allocate, 490
- redraw, 480
- redraw_events, 480
- reset, 480
- scroll_offset_x, 486
- scroll_offset_y, 486
- select_action, 487
- selecting, 487
- seqroll, 477
- set_adding, 484
- set_background_sequence, 479
- set_current_offset_x_y, 487
- set_data_type, 479
- set_key, 478
- set_note_length, 478
- set_scale, 478
- set_snap, 478
- set_zoom, 478
- snap_x, 481
- snap_y, 481
- start_paste, 480
- update_and_draw, 480
- update_background, 479
- update_mouse_pointer, 485
- update_pixmap, 479
- update_sizes, 479
- vertical_adjust, 481
- xy_to_rect, 482
- seq64::seqtime, 493
 - ~seqtime, 495
 - change_horz, 495
 - draw_pixmap_on_window, 495
 - draw_progress_on_window, 495
 - idle_progress, 496
 - m_ppqn, 496
 - m_scroll_offset_ticks, 496
 - m_scroll_offset_x, 496
 - m_seq, 496
 - m_zoom, 496
 - on_button_press_event, 496
 - on_button_release_event, 496
 - on_expose_event, 496
 - on_realize, 496
 - on_size_allocate, 496
 - redraw, 495
 - reset, 495
 - seqtime, 495
 - set_zoom, 495
 - update_pixmap, 495
 - update_sizes, 496
- seq64::sequence, 496
 - ~sequence, 506
 - add_event, 513, 524
 - add_note, 524
 - add_trigger, 514
 - adjust_offset, 534
 - adjust_timestamp, 523
 - adjust_trigger_offsets_to_length, 533
 - any_selected_notes, 507
 - background_sequence, 532
 - change_event_data_range, 525
 - check_queued_tick, 511
 - clear_triggers, 518
 - clip_timestamp, 523
 - clocks_per_metronome, 508
 - copy_events, 532
 - copy_selected, 521
 - copy_selected_trigger, 517
 - copy_triggers, 518
 - cut_selected, 521
 - cut_selected_trigger, 517
 - decrement_selected, 526
 - del_selected_trigger, 517
 - del_trigger, 514
 - e_deselect, 506
 - e_is_selected, 506
 - e_remove_one, 506
 - e_select, 506
 - e_select_one, 506
 - e_toggle_selection, 506
 - e_would_select, 506
 - event_count, 507
 - EventStack, 506
 - events, 507
 - fill_container, 531
 - get_32nds_per_quarter, 508
 - get_beat_width, 508
 - get_beats_per_bar, 508
 - get_clipboard_box, 522
 - get_editing, 509
 - get_last_tick, 510
 - get_length, 509
 - get_max_trigger, 518
 - get_measures, 508
 - get_midi_bus, 518
 - get_midi_channel, 512
 - get_minmax_note_events, 530
 - get_name, 509
 - get_next_event, 530, 531
 - get_next_note_event, 529
 - get_next_trigger, 531
 - get_num_selected_events, 520
 - get_num_selected_notes, 520
 - get_playing, 510
 - get_ppqn, 508

get_quantized_rec, 511
get_queued, 511
get_queued_tick, 511
get_raise, 509
get_recording, 511
get_selected_box, 522
get_song_mute, 509
get_thru, 511
get_trigger_offset, 518
get_trigger_state, 515
grow_selected, 527
grow_trigger, 514
increment_selected, 526
intersect_events, 516
intersect_notes, 516
intersect_triggers, 515
is_dirty_edit, 511
is_dirty_main, 511
is_dirty_names, 512
is_dirty_perf, 512
is_smf_0, 512
link_new, 528
m_32nds_per_quarter, 537
m_background_sequence, 537
m_bus, 535
m_clocks_per_metronome, 537
m_dirty_edit, 535
m_dirty_main, 535
m_dirty_names, 536
m_dirty_perf, 536
m_editing, 536
m_events, 534
m_events_clipboard, 534
m_events_redo, 535
m_events_undo, 535
m_iterator_draw, 535
m_last_tick, 536
m_length, 536
m_masterbus, 535
m_maxbeats, 536
m_midi_channel, 535
m_musical_key, 537
m_musical_scale, 537
m_mutex, 537
m_name, 536
m_note_off_margin, 537
m_notes_on, 535
m_parent, 534
m_playing, 535
m_playing_notes, 535
m_ppqn, 536
m_quantized_rec, 535
m_queued, 535
m_queued_tick, 536
m_raise, 536
m_rec_vol, 537
m_recording, 535
m_seq_number, 536
m_snap_tick, 536
m_song_mute, 535
m_thru, 535
m_time_beat_width, 536
m_time_beats_per_measure, 536
m_trigger_offset, 536
m_triggers, 535
m_us_per_quarter_note, 537
m_was_playing, 535
mark_selected, 528
mod_last_tick, 510
move_selected_notes, 523
move_selected_triggers_to, 517
move_triggers, 518
musical_key, 532
musical_scale, 532
name, 509
note_off_margin, 532
number, 507
off_playing_notes, 529
off_queued, 510
on_queued, 510
operator=, 506
partial_assign, 506
paste_selected, 521
paste_trigger, 517
pause, 529
perform, 534
play, 513
play_note_off, 529
play_note_on, 528
pop_redo, 507
pop_trigger_undo, 507
pop_undo, 507
print, 512
print_triggers, 513
push_quantize, 531
push_trigger_undo, 507
push_undo, 507
put_event_on_bus, 533
quantize_events, 531
remove, 534
remove_all, 534
remove_marked, 528
remove_selected, 528
reset, 529
reset_draw_marker, 529
reset_draw_trigger_marker, 529
select_action_e, 506
select_all, 521
select_all_notes, 520
select_events, 519, 520
select_note_events, 519
select_trigger, 515
selected_trigger_end, 517
selected_trigger_start, 517
sequence, 506
set_32nds_per_quarter, 508

- set_beat_width, 508
- set_beats_per_bar, 508
- set_dirty, 512
- set_dirty_mp, 512
- set_editing, 509
- set_last_tick, 510
- set_length, 509
- set_master_midi_bus, 519
- set_measures, 508
- set_midi_bus, 518
- set_midi_channel, 512
- set_name, 508
- set_parent, 532
- set_playing, 510
- set_quantized_rec, 511
- set_raise, 509
- set_rec_vol, 509
- set_recording, 511
- set_snap_tick, 511
- set_song_mute, 509
- set_thru, 511
- set_trigger_offset, 533
- show_events, 532
- split_trigger, 514, 533
- stream_event, 525
- stretch_selected, 527
- toggle_playing, 510
- toggle_queued, 510
- toggle_song_mute, 509
- transpose_notes, 531
- triggerlist, 507
- triggers, 534
- unpaint_all, 528
- unselect, 528
- unselect_triggers, 515
- us_per_quarter_note, 508
- verify_and_link, 528
- zero_markers, 528
- seq64::trigger, 538
 - decrement_offset, 539
 - decrement_tick_end, 539
 - decrement_tick_start, 539
 - increment_offset, 539
 - increment_tick_end, 539
 - increment_tick_start, 539
 - m_offset, 540
 - m_selected, 540
 - m_tick_end, 540
 - m_tick_start, 540
 - offset, 539
 - operator<, 539
 - selected, 539, 540
 - tick_end, 539
 - tick_start, 539
 - trigger, 539
- seq64::triggers, 540
 - ~triggers, 542
 - add, 544
 - adjust_offset, 549
 - adjust_offsets_to_length, 544
 - clear, 548
 - copy, 548
 - copy_selected, 546
 - get_maximum, 547
 - get_selected_end, 547
 - get_selected_start, 547
 - get_state, 545
 - grow, 545
 - intersect, 546
 - List, 542
 - m_clipboard, 549
 - m_iterator_draw_trigger, 549
 - m_iterator_play_trigger, 549
 - m_length, 549
 - m_parent, 549
 - m_ppqn, 549
 - m_redo_stack, 549
 - m_trigger_copied, 549
 - m_triggers, 549
 - m_undo_stack, 549
 - move, 547
 - move_selected, 546
 - next, 548
 - next_trigger, 549
 - operator=, 542
 - paste, 546
 - play, 543
 - pop_undo, 543
 - print, 543
 - push_undo, 543
 - remove, 545
 - remove_selected, 546
 - reset_draw_trigger_marker, 549
 - select, 546
 - set_length, 543
 - set_ppqn, 543
 - split, 544, 545
 - Stack, 542
 - triggerlist, 543
 - triggers, 542
 - unselect, 546
- seq64::user_instrument, 550
 - controller_active, 552
 - controller_count, 552
 - controller_max, 552
 - controller_name, 552
 - copy_definitions, 553
 - is_valid, 551
 - m_controller_count, 553
 - m_instrument_def, 553
 - m_is_valid, 553
 - name, 551
 - operator=, 551
 - set_controller, 552
 - set_defaults, 551
 - set_name, 552

- user_instrument, 551
- seq64::user_instrument_t, 553
 - controllers, 553
 - controllers_active, 554
 - instrument, 553
- seq64::user_midi_bus, 554
 - channel_count, 556
 - channel_max, 556
 - copy_definitions, 556
 - instrument, 556
 - is_valid, 555
 - m_channel_count, 557
 - m_is_valid, 556
 - m_midi_bus_def, 557
 - name, 555
 - operator=, 555
 - set_defaults, 555
 - set_instrument, 556
 - set_name, 556
 - user_midi_bus, 555
- seq64::user_midi_bus_t, 557
 - alias, 557
 - instrument, 557
- seq64::user_settings, 557
 - add_bus, 566
 - add_instrument, 566
 - allow_two_perfedits, 568, 570
 - baseline_ppqn, 570
 - bus, 566
 - bus_count, 566
 - bus_instrument, 566
 - bus_name, 566
 - BussConstIterator, 565
 - BussIterator, 564
 - Busses, 564
 - control_height, 567, 569
 - controller_active, 566
 - controller_name, 566
 - dump_summary, 569
 - global_seq_feature, 567, 568
 - gmute_tracks, 567
 - grid_brackets, 567, 568
 - grid_is_black, 567
 - grid_is_normal, 567
 - grid_is_white, 567
 - grid_style, 566, 568
 - grid_style_black, 565
 - grid_style_max, 565
 - grid_style_normal, 565
 - grid_style_white, 565
 - instrument, 566
 - instrument_controller_active, 566
 - instrument_controller_name, 566
 - instrument_count, 566
 - instrument_name, 566
 - InstrumentConstIterator, 565
 - InstrumentIterator, 565
 - Instruments, 565
 - m_allow_two_perfedits, 574
 - m_control_height, 573
 - m_current_zoom, 573
 - m_global_seq_feature_save, 573
 - m_gmute_tracks, 576
 - m_grid_brackets, 572
 - m_grid_style, 571
 - m_h_perf_page_increment, 574
 - m_instruments, 571
 - m_mainwid_border, 572
 - m_mainwid_spacing, 572
 - m_mainwid_x, 576
 - m_mainwid_y, 576
 - m_mainwnd_cols, 572
 - m_mainwnd_rows, 572
 - m_max_sequence, 576
 - m_max_sets, 572
 - m_midi_beat_width, 575
 - m_midi_beats_per_measure, 575
 - m_midi_beats_per_minute, 575
 - m_midi_buses, 571
 - m_midi_buss_override, 575
 - m_midi_ppqn, 575
 - m_progress_bar_colored, 574
 - m_progress_bar_thick, 574
 - m_save_user_config, 577
 - m_seqarea_seq_x, 576
 - m_seqarea_seq_y, 576
 - m_seqarea_x, 576
 - m_seqarea_y, 576
 - m_seqchars_x, 575
 - m_seqchars_y, 575
 - m_seqedit_bgsequence, 573
 - m_seqedit_key, 573
 - m_seqedit_scale, 573
 - m_seqs_in_set, 576
 - m_text_x, 574
 - m_text_y, 574
 - m_total_seqs, 575
 - m_use_new_font, 573
 - m_v_perf_page_increment, 574
 - m_window_redraw_rate_ms, 574
 - mainwid_border, 567, 569
 - mainwid_grid_style_t, 565
 - mainwid_spacing, 567, 569
 - mainwid_x, 567
 - mainwid_y, 567
 - mainwnd_cols, 567, 568
 - mainwnd_rows, 567, 568
 - max_sequence, 567
 - max_sets, 567, 568
 - max_zoom, 570
 - mc_baseline_ppqn, 577
 - mc_max_zoom, 577
 - mc_min_zoom, 577
 - midi_beat_width, 570, 571
 - midi_beats_per_bar, 570
 - midi_beats_per_minute, 570, 571

- midi_buss_override, 570
 - midi_ppqn, 569, 570
 - min_zoom, 570
 - normalize, 565
 - operator=, 565
 - perf_h_page_increment, 568, 570
 - perf_v_page_increment, 568, 570
 - private_bus, 571
 - private_instrument, 571
 - progress_bar_colored, 568, 570
 - progress_bar_thick, 568, 570
 - save_user_config, 568
 - seqarea_seq_x, 567, 569
 - seqarea_seq_y, 567, 569
 - seqarea_x, 567, 569
 - seqarea_y, 567, 569
 - seqchars_x, 567, 569
 - seqchars_y, 567, 569
 - seqedit_bgsequence, 568
 - seqedit_key, 568
 - seqedit_scale, 568
 - seqs_in_set, 567
 - set_bus_instrument, 566
 - set_defaults, 565
 - set_instrument_controllers, 566
 - text_x, 567, 569
 - text_y, 567, 569
 - use_new_font, 568, 570
 - user_settings, 565
 - userfile, 571
 - window_redraw_rate, 568, 570
 - zoom, 567
- seq64::userfile, 577
 - ~userfile, 579
 - dump_setting_summary, 579
 - parse, 579
 - userfile, 578
 - write, 579
- seq_clear_perf
 - seq64::seqmenu, 470
- seq_copy
 - seq64::seqmenu, 469
- seq_cut
 - seq64::seqmenu, 470
- seq_edit
 - seq64::seqmenu, 468
- seq_event_edit
 - seq64::seqmenu, 469
- seq_event_type_t
 - seq64, 52
- seq_from_xy
 - seq64::mainwid, 244
- seq_in_playing_screen
 - seq64::perform, 372
- seq_modifier_t
 - seq64, 51
- seq_new
 - seq64::seqmenu, 469
- seq_number_size
 - seq64::midifile, 308
- seq_paste
 - seq64::seqmenu, 470
- seq_scroll_direction_t
 - seq64, 52
- seq_set_and_edit
 - seq64::mainwid, 242
 - seq64::seqmenu, 469
- seq_set_and_eventedit
 - seq64::mainwid, 242
 - seq64::seqmenu, 469
- seqarea_seq_x
 - seq64::user_settings, 567, 569
- seqarea_seq_y
 - seq64::user_settings, 567, 569
- seqarea_x
 - seq64::user_settings, 567, 569
- seqarea_y
 - seq64::user_settings, 567, 569
- seqchars_x
 - seq64::user_settings, 567, 569
- seqchars_y
 - seq64::user_settings, 567, 569
- seqdata
 - seq64::seqdata, 422
- seqedit
 - seq64::seqedit, 434
- seqedit_bgsequence
 - seq64::user_settings, 568
- seqedit_key
 - seq64::user_settings, 568
- seqedit_scale
 - seq64::user_settings, 568
- seqevent
 - seq64::seqdata, 427
 - seq64::seqevent, 449
- seqkeys
 - seq64::seqkeys, 459
- seqmenu
 - seq64::seqedit, 441
 - seq64::seqmenu, 467
- seqroll
 - seq64::seqdata, 427
 - seq64::seqroll, 477
- seqs_in_set
 - seq64::user_settings, 567
- seqspec_string
 - seq64::editable_event, 101
- seqtime
 - seq64::seqtime, 495
- sequence
 - seq64::event_list, 130
 - seq64::sequence, 506
- sequence_count
 - seq64::perform, 347
- sequence_key
 - seq64::mainwnd, 256

- seq64::perform, 369
- sequence_label
 - seq64::perform, 369
- sequence_max
 - seq64::perform, 347
- sequence_playing_change
 - seq64::perform, 361
- sequence_playing_off
 - seq64::perform, 362
- sequence_playing_on
 - seq64::perform, 361
- sequence_playing_toggle
 - seq64::perform, 361
- session_event
 - seq64::jack_assistant, 196
- set
 - seq64::keybindentry, 208
 - seq64::midi_control, 274
- set_32nds_per_quarter
 - seq64::sequence, 508
- set_active
 - seq64::perform, 356
- set_adding
 - seq64::Seq24PerfInput, 417
 - seq64::Seq24SeqEventInput, 418
 - seq64::seqroll, 484
- set_all_key_events
 - seq64::keys_perform, 220
 - seq64::keys_perform_gtk2, 226
 - seq64::perform, 375
- set_all_key_groups
 - seq64::keys_perform, 220
 - seq64::keys_perform_gtk2, 226
 - seq64::perform, 375
- set_alsa_client_id
 - seq64::lash, 232
- set_and_copy_mute_group
 - seq64::perform, 356
- set_background_sequence
 - seq64::seqedit, 437
 - seq64::seqroll, 479
- set_beat_width
 - seq64::jack_assistant, 194
 - seq64::perfedit, 324
 - seq64::perform, 348
 - seq64::seqedit, 435
 - seq64::sequence, 508
- set_beats_per_bar
 - seq64::perfedit, 324
 - seq64::perform, 348
 - seq64::seqedit, 435
 - seq64::sequence, 508
- set_beats_per_measure
 - seq64::jack_assistant, 195
- set_beats_per_minute
 - seq64::jack_assistant, 195
 - seq64::mastermidibus, 263
 - seq64::perform, 359
- set_bus_and_midi_channel
 - seq64::seqmenu, 470
- set_bus_instrument
 - seq64::user_settings, 566
- set_channel
 - seq64::event, 117
- set_clock
 - seq64::mastermidibus, 267
 - seq64::midibus, 292
- set_clock_mod
 - seq64::midibus, 293
- set_config_files
 - seq64::rc_settings, 412
- set_controller
 - seq64::user_instrument, 552
- set_current_drop_x
 - seq64::gui_drawingarea_gtk2, 181
- set_current_drop_y
 - seq64::gui_drawingarea_gtk2, 181
- set_current_event
 - seq64::eventslots, 146
- set_current_offset_x_y
 - seq64::seqroll, 487
- set_data
 - seq64::event, 117
- set_data_type
 - seq64::seqdata, 423
 - seq64::seqedit, 438
 - seq64::seqevent, 450
 - seq64::seqroll, 479
- set_defaults
 - seq64::rc_settings, 409
 - seq64::user_instrument, 551
 - seq64::user_midi_bus, 555
 - seq64::user_settings, 565
- set_dirty
 - seq64::eventedit, 137
 - seq64::sequence, 512
- set_dirty_mp
 - seq64::sequence, 512
- set_edit_sequence
 - seq64::perform, 347
 - seq64::seqmenu, 468
- set_editing
 - seq64::sequence, 509
- set_event_category
 - seq64::eventedit, 136
- set_event_data_0
 - seq64::eventedit, 136
- set_event_data_1
 - seq64::eventedit, 137
- set_event_name
 - seq64::eventedit, 136
- set_event_timestamp
 - seq64::eventedit, 136
- set_group_mute_state
 - seq64::perform, 363
- set_guides

- seq64::perfedit, 324
- seq64::perfroll, 388
- seq64::perftime, 401
- set_hint_key
 - seq64::seqkeys, 460
- set_hint_state
 - seq64::seqkeys, 460
- set_image
 - seq64::mainwnd, 255
 - seq64::perfedit, 325
- set_input
 - seq64::mastermidibus, 267
 - seq64::midibus, 293
- set_input_bus
 - seq64::perform, 369
- set_instrument
 - seq64::user_midi_bus, 556
- set_instrument_controllers
 - seq64::user_settings, 566
- set_jack_running
 - seq64::jack_assistant, 198
- set_jack_tick
 - seq64::perform, 351
- set_key
 - seq64::seqedit, 437
 - seq64::seqkeys, 459
 - seq64::seqroll, 478
- set_key_event
 - seq64::keys_perform, 220
 - seq64::perform, 375
- set_key_group
 - seq64::keys_perform, 220
 - seq64::perform, 375
- set_keys
 - seq64::keys_perform, 214
- set_last_tick
 - seq64::sequence, 510
- set_left_tick
 - seq64::perform, 351
- set_length
 - seq64::sequence, 509
 - seq64::triggers, 543
- set_line
 - seq64::gui_drawingarea_gtk2, 175
- set_looped
 - seq64::perfedit, 325
- set_looping
 - seq64::perform, 359
- set_master_midi_bus
 - seq64::sequence, 519
- set_measures
 - seq64::seqedit, 436
 - seq64::sequence, 508
- set_midi_bus
 - seq64::seqedit, 437
 - seq64::sequence, 518
- set_midi_channel
 - seq64::seqedit, 437
- seq64::sequence, 512
- set_mode_group_learn
 - seq64::perform, 355
- set_mode_group_mute
 - seq64::perform, 355
- set_name
 - seq64::sequence, 508
 - seq64::user_instrument, 552
 - seq64::user_midi_bus, 556
- set_note
 - seq64::event, 119
- set_note_length
 - seq64::seqedit, 435
 - seq64::seqroll, 478
- set_note_velocity
 - seq64::event, 119
- set_offset
 - seq64::perform, 363
- set_orig_ticks
 - seq64::perform, 359
- set_parent
 - seq64::sequence, 532
- set_playback_mode
 - seq64::perform, 373
- set_playing
 - seq64::sequence, 510
- set_playing_screensets
 - seq64::keys_perform, 216
 - seq64::perform, 354
- set_position
 - seq64::jack_assistant, 202
- set_ppqn
 - seq64::jack_assistant, 198
 - seq64::mastermidibus, 263
 - seq64::perfroll, 389
 - seq64::perftime, 402
 - seq64::triggers, 543
- set_quantized_rec
 - seq64::sequence, 511
- set_raise
 - seq64::sequence, 509
- set_rec_vol
 - seq64::seqedit, 436
 - seq64::sequence, 509
- set_recording
 - seq64::sequence, 511
- set_right_tick
 - seq64::perform, 351
- set_running
 - seq64::perform, 372
- set_scale
 - seq64::perftime, 401
 - seq64::seqedit, 437
 - seq64::seqkeys, 459
 - seq64::seqroll, 478
- set_screen_set_notepad
 - seq64::perform, 354
- set_screensets

- seq64::mainwid, [241](#)
- seq64::perform, [354](#)
- set_seq_count
 - seq64::eventedit, [136](#)
- set_seq_ppqn
 - seq64::eventedit, [136](#)
- set_seq_time_sig
 - seq64::eventedit, [136](#)
- set_seq_title
 - seq64::eventedit, [136](#)
- set_sequence_control_status
 - seq64::perform, [361](#)
- set_sequence_input
 - seq64::mastermidibus, [265](#)
- set_snap
 - seq64::perfedit, [324](#)
 - seq64::seqedit, [435](#)
 - seq64::seqevent, [450](#)
 - seq64::seqroll, [478](#)
- set_snap_tick
 - seq64::sequence, [511](#)
- set_song_mute
 - seq64::sequence, [509](#)
- set_start_tick
 - seq64::perform, [351](#)
- set_status
 - seq64::event, [116](#), [117](#)
- set_status_from_string
 - seq64::editable_event, [100](#)
- set_sysex_size
 - seq64::event, [118](#)
- set_text
 - seq64::eventslots, [149](#)
- set_thru
 - seq64::sequence, [511](#)
- set_timestamp
 - seq64::event, [114](#)
- set_transposable
 - seq64::seqmenu, [470](#)
- set_trigger_offset
 - seq64::sequence, [533](#)
- set_was_active
 - seq64::perform, [357](#)
- set_zoom
 - seq64::perfedit, [324](#)
 - seq64::perfroll, [390](#)
 - seq64::perftime, [401](#)
 - seq64::seqdata, [423](#)
 - seq64::seqedit, [435](#)
 - seq64::seqevent, [450](#)
 - seq64::seqroll, [478](#)
 - seq64::seqtime, [495](#)
- shift_lock
 - seq64::keystroke, [230](#)
- shorten_file_spec
 - seq64, [58](#)
- show_events
 - seq64::sequence, [532](#)
- show_midi
 - seq64::rc_settings, [409](#)
- show_position
 - seq64::jack_assistant, [200](#)
- show_statuses
 - seq64::jack_assistant, [200](#)
- show_ui_sequence_key
 - seq64::keys_perform, [218](#)
 - seq64::perform, [364](#)
- show_ui_sequence_number
 - seq64::keys_perform, [218](#)
 - seq64::perform, [364](#)
- signal
 - seq64::condition_var, [90](#)
- signal_action
 - seq64::mainwnd, [257](#)
- size
 - seq64::midi_container, [271](#)
 - seq64::midi_list, [277](#)
 - seq64::midi_vector, [287](#)
- SlotMap
 - seq64::keys_perform, [214](#)
- sm_category_arrays
 - seq64::editable_event, [102](#)
- sm_category_names
 - seq64::editable_event, [102](#)
- sm_channel_event_names
 - seq64::editable_event, [102](#)
- sm_cond
 - seq64::condition_var, [90](#)
- sm_internal_keys
 - seq64::gui_assistant_gtk2, [170](#)
- sm_mc_dummy
 - seq64::perform, [376](#)
- sm_meta_event_names
 - seq64::editable_event, [102](#)
- sm_prop_event_names
 - seq64::editable_event, [102](#)
- sm_recursive_mutex
 - seq64::mutex, [311](#)
- sm_status_pairs
 - seq64::jack_assistant, [204](#)
- sm_system_event_names
 - seq64::editable_event, [102](#)
- snap_x
 - seq64::perfroll, [390](#)
 - seq64::seqevent, [452](#)
 - seq64::seqroll, [481](#)
- snap_y
 - seq64::seqevent, [452](#)
 - seq64::seqroll, [481](#)
- snapshot_1
 - seq64::keys_perform, [216](#)
- snapshot_2
 - seq64::keys_perform, [216](#)
- sort
 - seq64::event_list, [128](#)
- split

- seq64::midi_splitter, 282
- seq64::triggers, 544, 545
- split_channel
 - seq64::midi_splitter, 282
- split_trigger
 - seq64::perform, 352
 - seq64::perfroll, 390
 - seq64::sequence, 514, 533
- Stack
 - seq64::triggers, 542
- start
 - seq64::jack_assistant, 196
 - seq64::keys_perform, 217
 - seq64::lash, 232
 - seq64::mastermidibus, 264
 - seq64::midibus, 292
 - seq64::perform, 356
- start_jack
 - seq64::perform, 356
- start_key
 - seq64::perform, 368
- start_paste
 - seq64::seqevent, 451
 - seq64::seqroll, 480
- start_playing
 - seq64::mainwnd, 255
 - seq64::perfedit, 326
 - seq64::perform, 367
- stats
 - seq64::rc_settings, 409
- status
 - seq64::midi_control, 274
- status_string
 - seq64::editable_event, 101
- stock_event_string
 - seq64::editable_event, 101
- stop
 - seq64::jack_assistant, 196
 - seq64::keys_perform, 218
 - seq64::mastermidibus, 264
 - seq64::midibus, 292
 - seq64::perform, 356
- stop_jack
 - seq64::perform, 356
- stop_key
 - seq64::perform, 368
- stop_playing
 - seq64::mainwnd, 255
 - seq64::perfedit, 326
 - seq64::perform, 367
- stream_event
 - seq64::sequence, 525
- stretch_selected
 - seq64::sequence, 527
- string_is_void
 - seq64, 59
- string_not_void
 - seq64, 58
- string_to_midibyte
 - seq64, 58
- string_to_pulses
 - seq64, 57
 - seq64::editable_events, 106
- strings_match
 - seq64, 59
- sync
 - seq64::jack_assistant, 201
- sysex
 - seq64::mastermidibus, 266
 - seq64::midibus, 292
- tempo_from_beats_per_minute
 - seq64, 61
- tempo_to_bytes
 - seq64, 60
- text_x
 - seq64::user_settings, 567, 569
- text_y
 - seq64::user_settings, 567, 569
- thru_change_callback
 - seq64::seqedit, 438
- tick_end
 - seq64::trigger, 539
- tick_offset
 - seq64::perftime, 402
- tick_start
 - seq64::trigger, 539
- tick_to_pixel
 - seq64::perftime, 402
- ticks_to_delta_time_us
 - seq64, 62
- time_as_measures
 - seq64::editable_event, 100
- time_as_minutes
 - seq64::editable_event, 100
- time_as_pulses
 - seq64::editable_event, 100
- timeout
 - seq64::mainwid, 244
 - seq64::perfedit, 325
 - seq64::seqedit, 439
- timer_callback
 - seq64::mainwnd, 255
- timestamp
 - seq64::editable_event, 100
- timestamp_format_t
 - seq64::editable_event, 97
- timestamp_measures
 - seq64::editable_event, 98
- timestamp_pulses
 - seq64::editable_event, 98
- timestamp_string
 - seq64::editable_event, 100
- timestamp_time
 - seq64::editable_event, 98
- timestring_to_pulses
 - seq64, 57

- timing
 - seq64::editable_events, 106
- to_string
 - seq64, 65
- toLower
 - seq64::mainwnd, 256
- toggle_all_tracks
 - seq64::perform, 362
 - seq64::seqmenu, 470
- toggle_current_sequence
 - seq64::seqmenu, 468
- toggle_playing
 - seq64::mainwnd, 255
 - seq64::perfedit, 326
 - seq64::sequence, 510
- toggle_queued
 - seq64::sequence, 510
- toggle_song_mute
 - seq64::sequence, 509
- top_index
 - seq64::eventslots, 146
- track_end_size
 - seq64::midifile, 308
- track_name_size
 - seq64::midifile, 307
- transport_callback
 - seq64::options, 313
- transpose_notes
 - seq64::sequence, 531
- trigger
 - seq64::trigger, 539
- triggerlist
 - seq64::sequence, 507
 - seq64::triggers, 543
- triggers
 - seq64::sequence, 534
 - seq64::triggers, 542
- type
 - seq64::keybindentry, 207
- undo
 - seq64::perfedit, 325
- undo_callback
 - seq64::seqedit, 438
- unlock
 - seq64::mutex, 311
- unmark
 - seq64::event, 119
- unmark_all
 - seq64::event_list, 129
- unmodify
 - seq64::event_list, 127
- unmute_all_tracks
 - seq64::seqmenu, 470
- unpaint
 - seq64::event, 119
- unpaint_all
 - seq64::event_list, 129
 - seq64::sequence, 528
- unselect
 - seq64::event, 119
 - seq64::sequence, 528
 - seq64::triggers, 546
- unselect_all
 - seq64::event_list, 130
- unselect_triggers
 - seq64::sequence, 515
- unset_edit_sequence
 - seq64::perform, 347
 - seq64::seqmenu, 468
- unset_mode_group_learn
 - seq64::perform, 355
- unset_mode_group_mute
 - seq64::perform, 355
- unset_sequence_control_status
 - seq64::perform, 361
- update_all_windows
 - seq64::seqedit, 438
- update_and_draw
 - seq64::seqroll, 480
- update_background
 - seq64::seqroll, 479
- update_mainwid_sequences
 - seq64, 72
 - seq64::mainwid, 246
- update_markers
 - seq64::mainwid, 243
- update_mouse_pointer
 - seq64::FruityPerfInput, 161
 - seq64::FruitySeqEventInput, 163
 - seq64::FruitySeqRollInput, 165
 - seq64::seqroll, 485
- update_perfedit_sequences
 - seq64, 72
 - seq64::perfedit, 326
- update_pixmap
 - seq64::perftime, 403
 - seq64::seqdata, 424
 - seq64::seqevent, 451
 - seq64::seqkeys, 460
 - seq64::seqroll, 479
 - seq64::seqtime, 495
- update_sequences_on_window
 - seq64::mainwid, 242
- update_sizes
 - seq64::perfroll, 388
 - seq64::perftime, 402
 - seq64::seqdata, 423
 - seq64::seqevent, 450
 - seq64::seqkeys, 461
 - seq64::seqroll, 479
 - seq64::seqtime, 496
- update_window_title
 - seq64::mainwnd, 256
- us_per_quarter_note
 - seq64::sequence, 508
- use_new_font

- seq64::user_settings, 568, 570
- user_filename
 - seq64::rc_settings, 411, 412
- user_filename_alt
 - seq64::rc_settings, 411, 412
- user_filespec
 - seq64::rc_settings, 409
- user_instrument
 - seq64::user_instrument, 551
- user_midi_bus
 - seq64::user_midi_bus, 555
- user_settings
 - seq64::user_settings, 565
- userfile
 - seq64::user_settings, 571
 - seq64::userfile, 578
- usr
 - seq64, 71
- v_adjustment
 - seq64::eventedit, 137
- valid_sequence
 - seq64::mainwid, 243
- value_to_name
 - seq64::editable_event, 99
- varinum_size
 - seq64::midifile, 306
- verify_and_link
 - seq64::event_list, 129
 - seq64::sequence, 528
- versiontext
 - seq64, 79
- vertical_adjust
 - seq64::perroll, 392
 - seq64::seqedit, 436
 - seq64::seqroll, 481
- vertical_set
 - seq64::perroll, 392
 - seq64::seqedit, 436
- WHITE
 - seq64::font, 156
- wait
 - seq64::condition_var, 90
- white
 - seq64::gui_palette_gtk2, 186
- width
 - seq64::gui_drawingarea_gtk2::rect, 415
 - seq64::rect, 414
- window_redraw_rate
 - seq64::user_settings, 568, 570
- window_x
 - seq64::gui_drawingarea_gtk2, 174
- window_y
 - seq64::gui_drawingarea_gtk2, 174
- with_jack
 - seq64::rc_settings, 410
- with_jack_master
 - seq64::rc_settings, 410
- with_jack_master_cond
 - seq64::rc_settings, 410
- with_jack_transport
 - seq64::rc_settings, 410
- write
 - seq64::configfile, 93
 - seq64::midifile, 298
 - seq64::optionsfile, 317
 - seq64::userfile, 579
- write_byte
 - seq64::midifile, 303
- write_long
 - seq64::midifile, 302
- write_options_files
 - seq64, 65
- write_prop_header
 - seq64::midifile, 305
- write_proprietary_track
 - seq64::midifile, 306
- write_seq_number
 - seq64::midifile, 304
- write_short
 - seq64::midifile, 303
- write_track_end
 - seq64::midifile, 305
- write_track_name
 - seq64::midifile, 304
- write_varinum
 - seq64::midifile, 303
- x
 - seq64::click, 88
 - seq64::gui_drawingarea_gtk2::rect, 415
 - seq64::rect, 414
- x_to_w
 - seq64::seqevent, 451
- xy_to_rect
 - seq64::seqdata, 424
 - seq64::seqroll, 482
- y
 - seq64::click, 88
 - seq64::gui_drawingarea_gtk2::rect, 415
 - seq64::rect, 414
- YELLOW_ON_BLACK
 - seq64::font, 156
- yellow
 - seq64::gui_palette_gtk2, 186
- zero_markers
 - seq64::sequence, 528
- zoom
 - seq64::user_settings, 567
- zoom_check
 - seq64::perfedit, 323
- zoom_power_of_2
 - seq64, 60