

Kitlist

1.1.0

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Kitlist Documentation</b>	<b>1</b>
1.1	Introduction	1
1.2	Additional Documentation	1
<b>2</b>	<b>Namespace Index</b>	<b>3</b>
2.1	Namespace List	3
<b>3</b>	<b>Hierarchical Index</b>	<b>5</b>
3.1	Class Hierarchy	5
<b>4</b>	<b>Class Index</b>	<b>7</b>
4.1	Class List	7
<b>5</b>	<b>File Index</b>	<b>9</b>
5.1	File List	9
<b>6</b>	<b>Namespace Documentation</b>	<b>11</b>
6.1	anonymous_namespace{kitlistgui.cpp} Namespace Reference	11
6.1.1	Variable Documentation	12
6.1.1.1	DEFAULT_FILENAME	12
6.1.1.2	DEFAULT_FILENAME_EXTENSION	12
6.1.1.3	GCONF_KEY	12
6.1.1.4	GCONF_KEY_CURRENT_FILENAME	12
6.1.1.5	GCONF_KEY_MAX_RECENT_FILES	13
6.1.1.6	GCONF_KEY_PAGE_TITLE	13
6.1.1.7	GCONF_KEY_RECENT_FILES	13
6.1.1.8	GLADE_APP_FILE	13
6.1.1.9	item_target_custom	14
6.1.1.10	item_target_text	14
6.1.1.11	PDF_FILENAME_EXTENSION	14
6.1.1.12	SB_ITEM_COUNT	14
6.1.1.13	SB_MSG	15
6.1.1.14	SB_PRINT	15
6.1.1.15	SB_SAVE	15
6.1.1.16	XML_ELEMENT_ID	15

<b>7 Class Documentation</b>	<b>17</b>
7.1 Category Class Reference	17
7.1.1 Detailed Description	18
7.1.2 Constructor & Destructor Documentation	18
7.1.2.1 ~Category()	18
7.1.3 Member Function Documentation	18
7.1.3.1 add_item()	18
7.1.3.2 execute()	19
7.1.3.3 foreach_item()	19
7.1.3.4 get_id()	19
7.1.3.5 get_name()	20
7.1.3.6 has_items()	20
7.1.3.7 item_count()	20
7.1.3.8 remove_item()	20
7.1.3.9 set_id()	21
7.1.3.10 set_name()	21
7.1.4 Friends And Related Function Documentation	21
7.1.4.1 CategoryCompareId	21
7.1.4.2 CategoryCompareName	21
7.1.4.3 KitModel	22
7.1.5 Member Data Documentation	22
7.1.5.1 m_id	22
7.1.5.2 m_items	22
7.1.5.3 m_name	22
7.2 CategoryCompareId Class Reference	23
7.2.1 Detailed Description	23
7.2.2 Constructor & Destructor Documentation	23
7.2.2.1 CategoryCompareId()	23
7.2.3 Member Function Documentation	23
7.2.3.1 operator()	23

---

7.3	CategoryCompareName Class Reference	24
7.3.1	Detailed Description	24
7.3.2	Constructor & Destructor Documentation	24
7.3.2.1	CategoryCompareName()	24
7.3.3	Member Function Documentation	24
7.3.3.1	operator()	24
7.4	FilterItem Class Reference	25
7.4.1	Detailed Description	25
7.4.2	Constructor & Destructor Documentation	25
7.4.2.1	FilterItem()	25
7.4.3	Member Function Documentation	25
7.4.3.1	operator()	25
7.4.4	Member Data Documentation	26
7.4.4.1	m_model	26
7.5	GuiState Class Reference	26
7.5.1	Detailed Description	27
7.5.2	Constructor & Destructor Documentation	27
7.5.2.1	GuiState()	27
7.5.3	Member Function Documentation	27
7.5.3.1	is_deleted()	27
7.5.3.2	is_dirty()	27
7.5.3.3	is_new()	28
7.5.3.4	reset()	28
7.5.3.5	set_deleted()	28
7.5.3.6	set_dirty()	28
7.5.3.7	set_new_flag()	29
7.5.4	Member Data Documentation	29
7.5.4.1	m_deleted	29
7.5.4.2	m_dirty	29
7.5.4.3	m_new	29

---

7.6	Item Class Reference	30
7.6.1	Detailed Description	30
7.6.2	Constructor & Destructor Documentation	31
7.6.2.1	Item() [1/2]	31
7.6.2.2	Item() [2/2]	31
7.6.3	Member Function Documentation	31
7.6.3.1	get_checked()	31
7.6.3.2	get_description()	31
7.6.3.3	get_id()	32
7.6.3.4	set_checked()	32
7.6.3.5	set_description()	32
7.6.3.6	set_id()	32
7.6.4	Friends And Related Function Documentation	32
7.6.4.1	ItemCompareId	33
7.6.4.2	ItemCompareName	33
7.6.4.3	operator<<	33
7.6.5	Member Data Documentation	33
7.6.5.1	m_checked	33
7.6.5.2	m_desc	33
7.6.5.3	m_id	34
7.7	ItemCompareId Class Reference	34
7.7.1	Detailed Description	34
7.7.2	Constructor & Destructor Documentation	34
7.7.2.1	ItemCompareId()	34
7.7.3	Member Function Documentation	35
7.7.3.1	operator>()	35
7.8	ItemCompareName Class Reference	35
7.8.1	Detailed Description	35
7.8.2	Constructor & Destructor Documentation	35
7.8.2.1	ItemCompareName()	36

---

---

7.8.3	Member Function Documentation	36
7.8.3.1	operator()	36
7.9	ItemFuncor Class Reference	36
7.9.1	Detailed Description	36
7.9.2	Member Function Documentation	37
7.9.2.1	operator()	37
7.10	KitList Class Reference	37
7.10.1	Detailed Description	38
7.10.2	Constructor & Destructor Documentation	38
7.10.2.1	KitList()	38
7.10.2.2	~KitList()	39
7.10.3	Member Function Documentation	39
7.10.3.1	add_item() [1/2]	39
7.10.3.2	add_item() [2/2]	39
7.10.3.3	append_items_to_category()	39
7.10.3.4	associate_item_with_category()	41
7.10.3.5	delete_category()	41
7.10.3.6	delete_item()	42
7.10.3.7	execute()	42
7.10.3.8	get_dao()	42
7.10.3.9	list_categories()	42
7.10.3.10	list_item()	43
7.10.3.11	list_items() [1/3]	43
7.10.3.12	list_items() [2/3]	43
7.10.3.13	list_items() [3/3]	43
7.10.3.14	list_items_end()	44
7.10.3.15	list_items_start()	44
7.10.3.16	new_category()	45
7.10.3.17	on_list_item()	45
7.10.3.18	remove_item_from_category()	45

7.10.3.19	<a href="#">set_all_flags()</a>	45
7.10.3.20	<a href="#">set_category_flag()</a>	46
7.10.3.21	<a href="#">set_item_flag()</a>	46
7.10.3.22	<a href="#">tick_items()</a> [1/3]	46
7.10.3.23	<a href="#">tick_items()</a> [2/3]	47
7.10.3.24	<a href="#">tick_items()</a> [3/3]	47
7.10.3.25	<a href="#">unset_all_flags()</a>	47
7.10.3.26	<a href="#">unset_category_flag()</a>	47
7.10.3.27	<a href="#">unset_item_flag()</a>	48
7.10.4	<a href="#">Member Data Documentation</a>	48
7.10.4.1	<a href="#">m_dao</a>	48
7.11	<a href="#">KitListDao Class Reference</a>	48
7.11.1	<a href="#">Detailed Description</a>	49
7.11.2	<a href="#">Constructor &amp; Destructor Documentation</a>	50
7.11.2.1	<a href="#">KitListDao()</a>	50
7.11.2.2	<a href="#">~KitListDao()</a>	50
7.11.3	<a href="#">Member Function Documentation</a>	50
7.11.3.1	<a href="#">add_item()</a> [1/2]	50
7.11.3.2	<a href="#">add_item()</a> [2/2]	51
7.11.3.3	<a href="#">append_items_to_category()</a>	51
7.11.3.4	<a href="#">associate_item_with_category()</a>	51
7.11.3.5	<a href="#">delete_category()</a>	52
7.11.3.6	<a href="#">delete_item()</a>	52
7.11.3.7	<a href="#">get_all_items()</a>	52
7.11.3.8	<a href="#">get_categories()</a>	53
7.11.3.9	<a href="#">get_category()</a>	53
7.11.3.10	<a href="#">get_model()</a>	53
7.11.3.11	<a href="#">get_next_category_id()</a>	54
7.11.3.12	<a href="#">get_next_item_id()</a>	54
7.11.3.13	<a href="#">is_verbose()</a>	54



---

7.11.3.14 new_category()	54
7.11.3.15 remove_item_from_category()	55
7.11.3.16 require_filename()	55
7.11.3.17 save_model()	56
7.11.3.18 set_all_flags()	56
7.11.3.19 set_category_flag()	56
7.11.3.20 set_item_flag()	56
7.11.3.21 set_verbose()	57
7.11.3.22 unset_all_flags()	57
7.11.3.23 unset_category_flag()	57
7.11.3.24 unset_item_flag()	58
7.11.3.25 update_item_checked_state()	58
7.11.4 Member Data Documentation	58
7.11.4.1 m_verbose_flag	58
7.12 KitListGui Class Reference	58
7.12.1 Detailed Description	62
7.12.2 Constructor & Destructor Documentation	62
7.12.2.1 KitListGui()	62
7.12.2.2 ~KitListGui()	63
7.12.3 Member Function Documentation	63
7.12.3.1 add_items()	63
7.12.3.2 cancel_add_category_window()	63
7.12.3.3 cancel_add_item_window()	64
7.12.3.4 cancel_preferences_window()	64
7.12.3.5 choose_filename()	64
7.12.3.6 choose_pdf_filename()	65
7.12.3.7 close_add_category_window()	65
7.12.3.8 close_add_item_window()	65
7.12.3.9 close_preferences_window()	66
7.12.3.10 confirm_lose_changes()	66

---

7.12.3.11 <code>copy_selected_items_to_clipboard()</code> . . . . .	66
7.12.3.12 <code>delete_selected_items()</code> . . . . .	67
7.12.3.13 <code>get_max_recent_files()</code> . . . . .	67
7.12.3.14 <code>get_selected_category()</code> . . . . .	67
7.12.3.15 <code>get_selected_items()</code> . . . . .	68
7.12.3.16 <code>init()</code> . . . . .	68
7.12.3.17 <code>init_add_item_window()</code> . . . . .	68
7.12.3.18 <code>on_category_change()</code> . . . . .	69
7.12.3.19 <code>on_cell_edit()</code> . . . . .	69
7.12.3.20 <code>on_clipboard_clear()</code> . . . . .	69
7.12.3.21 <code>on_clipboard_get()</code> . . . . .	70
7.12.3.22 <code>on_clipboard_received()</code> . . . . .	70
7.12.3.23 <code>on_delete_event()</code> . . . . .	70
7.12.3.24 <code>on_menu_add()</code> . . . . .	71
7.12.3.25 <code>on_menu_check_selected()</code> . . . . .	71
7.12.3.26 <code>on_menu_copy()</code> . . . . .	71
7.12.3.27 <code>on_menu_create_category()</code> . . . . .	71
7.12.3.28 <code>on_menu_cut()</code> . . . . .	72
7.12.3.29 <code>on_menu_delete()</code> . . . . .	72
7.12.3.30 <code>on_menu_delete_category()</code> . . . . .	72
7.12.3.31 <code>on_menu_export_to_pdf()</code> . . . . .	73
7.12.3.32 <code>on_menu_file_new()</code> . . . . .	73
7.12.3.33 <code>on_menu_file_open()</code> . . . . .	73
7.12.3.34 <code>on_menu_help_about()</code> . . . . .	74
7.12.3.35 <code>on_menu_paste()</code> . . . . .	74
7.12.3.36 <code>on_menu_preferences()</code> . . . . .	74
7.12.3.37 <code>on_menu_print()</code> . . . . .	74
7.12.3.38 <code>on_menu_quit()</code> . . . . .	75
7.12.3.39 <code>on_menu_recent_file()</code> . . . . .	75
7.12.3.40 <code>on_menu_rename_category()</code> . . . . .	75

---

---

7.12.3.41 on_menu_save()	76
7.12.3.42 on_menu_save_as()	76
7.12.3.43 on_menu_select_all()	76
7.12.3.44 on_menu_show_all()	77
7.12.3.45 on_menu_show_checked()	77
7.12.3.46 on_menu_show_unchecked()	77
7.12.3.47 on_menu_uncheck_selected()	77
7.12.3.48 on_printoperation_done()	78
7.12.3.49 on_printoperation_status_changed()	78
7.12.3.50 on_row_changed()	78
7.12.3.51 open_file()	79
7.12.3.52 paste_from_xml()	79
7.12.3.53 paste_status_received()	79
7.12.3.54 raise()	80
7.12.3.55 refresh_category_list()	80
7.12.3.56 refresh_item_list()	80
7.12.3.57 run()	81
7.12.3.58 safe_open_file()	81
7.12.3.59 selected_row_callback()	81
7.12.3.60 set_page_title()	81
7.12.3.61 set_selected()	82
7.12.3.62 toggle_selected()	82
7.12.3.63 update_item_count()	82
7.12.3.64 update_paste_status()	82
7.12.3.65 update_recent_files()	82
7.12.3.66 update_recent_files_menu()	83
7.12.4 Member Data Documentation	83
7.12.4.1 m_category_cols	83
7.12.4.2 m_category_combo	83
7.12.4.3 m_checkbutton_add_item	84

---

---

7.12.4.4	<a href="#">m_clipboard_items</a>	84
7.12.4.5	<a href="#">m_current_cat_id</a>	84
7.12.4.6	<a href="#">m_entry_add_category</a>	84
7.12.4.7	<a href="#">m_entry_add_item</a>	85
7.12.4.8	<a href="#">m_entry_page_title</a>	85
7.12.4.9	<a href="#">m_file_save_menu_item</a>	85
7.12.4.10	<a href="#">m_file_save_tool_button</a>	85
7.12.4.11	<a href="#">m_filename</a>	86
7.12.4.12	<a href="#">m_ignore_list_events</a>	86
7.12.4.13	<a href="#">m_item_cols</a>	86
7.12.4.14	<a href="#">m_item_tree_view</a>	86
7.12.4.15	<a href="#">m_kit</a>	87
7.12.4.16	<a href="#">m_page_title</a>	87
7.12.4.17	<a href="#">m_paste_menu_item</a>	87
7.12.4.18	<a href="#">m_paste_tool_button</a>	87
7.12.4.19	<a href="#">m_recent_files_menu_item</a>	88
7.12.4.20	<a href="#">m_ref_category_list_store</a>	88
7.12.4.21	<a href="#">m_ref_item_tree_model</a>	88
7.12.4.22	<a href="#">m_ref_page_setup</a>	88
7.12.4.23	<a href="#">m_ref_printer_settings</a>	89
7.12.4.24	<a href="#">m_service</a>	89
7.12.4.25	<a href="#">m_state</a>	89
7.12.4.26	<a href="#">m_status_bar</a>	89
7.12.4.27	<a href="#">m_window</a>	90
7.12.4.28	<a href="#">m_window_add_category</a>	90
7.12.4.29	<a href="#">m_window_add_item</a>	90
7.12.4.30	<a href="#">m_window_preferences</a>	90
7.12.4.31	<a href="#">m_yaml_config</a>	91
7.13	<a href="#">KitModel Class Reference</a>	91
7.13.1	<a href="#">Detailed Description</a>	92

---

---

7.13.2	Constructor & Destructor Documentation	92
7.13.2.1	KitModel()	92
7.13.2.2	~KitModel()	93
7.13.3	Member Function Documentation	93
7.13.3.1	add_category()	93
7.13.3.2	add_item() [1/2]	93
7.13.3.3	add_item() [2/2]	94
7.13.3.4	copy_items()	94
7.13.3.5	filter()	94
7.13.3.6	find_category()	95
7.13.3.7	find_item()	95
7.13.3.8	foreach_category()	95
7.13.3.9	foreach_category_iter()	96
7.13.3.10	foreach_item()	96
7.13.3.11	foreach_item_iter()	96
7.13.3.12	get_all_items() [1/2]	96
7.13.3.13	get_all_items() [2/2]	97
7.13.3.14	get_categories()	97
7.13.3.15	is_dirty()	97
7.13.3.16	purge()	97
7.13.3.17	reset()	98
7.13.3.18	set_dirty()	98
7.13.3.19	show_all()	98
7.13.3.20	show_checked_only()	99
7.13.3.21	show_unchecked_only()	99
7.13.4	Member Data Documentation	99
7.13.4.1	m_category_map	99
7.13.4.2	m_dirty	100
7.13.4.3	m_item_filter	100
7.13.4.4	m_item_map	100

7.14 KitParser Class Reference . . . . .	100
7.14.1 Detailed Description . . . . .	102
7.14.2 Constructor & Destructor Documentation . . . . .	102
7.14.2.1 KitParser() . . . . .	102
7.14.2.2 ~KitParser() . . . . .	102
7.14.3 Member Function Documentation . . . . .	102
7.14.3.1 on_characters() . . . . .	102
7.14.3.2 on_comment() . . . . .	103
7.14.3.3 on_end_document() . . . . .	103
7.14.3.4 on_end_element() . . . . .	103
7.14.3.5 on_error() . . . . .	103
7.14.3.6 on_fatal_error() . . . . .	104
7.14.3.7 on_start_document() . . . . .	104
7.14.3.8 on_start_element() . . . . .	104
7.14.3.9 on_warning() . . . . .	104
7.14.3.10 process_category() . . . . .	104
7.14.3.11 process_category_item() . . . . .	105
7.14.3.12 process_item() . . . . .	105
7.14.4 Member Data Documentation . . . . .	106
7.14.4.1 m_category . . . . .	106
7.14.4.2 m_cdata . . . . .	106
7.14.4.3 m_item . . . . .	106
7.14.4.4 m_model . . . . .	106
7.15 KitPrintOperation Class Reference . . . . .	107
7.15.1 Detailed Description . . . . .	108
7.15.2 Constructor & Destructor Documentation . . . . .	108
7.15.2.1 ~KitPrintOperation() . . . . .	108
7.15.3 Member Function Documentation . . . . .	108
7.15.3.1 create() . . . . .	108
7.15.3.2 new_footer() . . . . .	108

---

7.15.3.3	<code>new_header()</code>	109
7.15.3.4	<code>on_begin_print()</code>	109
7.15.3.5	<code>on_draw_page()</code>	110
7.15.3.6	<code>set_items()</code>	110
7.15.3.7	<code>set_page_title()</code>	110
7.15.4	Member Data Documentation	110
7.15.4.1	<code>m_items</code>	111
7.15.4.2	<code>m_page_breaks</code>	111
7.15.4.3	<code>m_page_title</code>	111
7.15.4.4	<code>m_ref_footers</code>	111
7.15.4.5	<code>m_ref_headers</code>	111
7.15.4.6	<code>m_ref_layout</code>	112
7.16	ModelCategory Class Reference	112
7.16.1	Detailed Description	113
7.16.2	Constructor & Destructor Documentation	113
7.16.2.1	<code>ModelCategory()</code>	113
7.16.2.2	<code>~ModelCategory()</code>	113
7.16.3	Member Function Documentation	113
7.16.3.1	<code>add_item()</code>	114
7.16.3.2	<code>get_added_children()</code>	114
7.16.3.3	<code>get_items()</code> [1/2]	114
7.16.3.4	<code>get_items()</code> [2/2]	114
7.16.3.5	<code>get_model_items()</code>	115
7.16.3.6	<code>get_removed_children()</code>	115
7.16.3.7	<code>purge()</code>	115
7.16.3.8	<code>remove_item()</code>	116
7.16.3.9	<code>remove_items()</code>	116
7.16.3.10	<code>reset()</code>	116
7.16.4	Friends And Related Function Documentation	117
7.16.4.1	<code>KitModel</code>	117

---

---

7.16.5	Member Data Documentation	117
7.16.5.1	m_added_children	117
7.16.5.2	m_removed_children	117
7.17	ModelCategoryColumns Class Reference	118
7.17.1	Detailed Description	118
7.17.2	Constructor & Destructor Documentation	118
7.17.2.1	ModelCategoryColumns()	118
7.17.3	Member Data Documentation	118
7.17.3.1	m_col_num	119
7.17.3.2	m_col_text	119
7.18	ModelItem Class Reference	119
7.18.1	Detailed Description	120
7.18.2	Constructor & Destructor Documentation	120
7.18.2.1	ModelItem()	120
7.18.3	Member Function Documentation	120
7.18.3.1	set_checked()	120
7.18.4	Friends And Related Function Documentation	120
7.18.4.1	ModelItemCompareId	120
7.19	ModelItemColumns Class Reference	121
7.19.1	Detailed Description	121
7.19.2	Constructor & Destructor Documentation	121
7.19.2.1	ModelItemColumns()	121
7.19.3	Member Data Documentation	121
7.19.3.1	m_col_checked	122
7.19.3.2	m_col_num	122
7.19.3.3	m_col_text	122
7.20	ModelItemCompareId Class Reference	122
7.20.1	Detailed Description	123
7.20.2	Constructor & Destructor Documentation	123
7.20.2.1	ModelItemCompareId()	123

---



---

7.20.3	Member Function Documentation	123
7.20.3.1	operator()	123
7.21	Service Class Reference	123
7.21.1	Detailed Description	125
7.21.2	Constructor & Destructor Documentation	125
7.21.2.1	Service()	125
7.21.2.2	~Service()	125
7.21.3	Member Function Documentation	126
7.21.3.1	copy_items()	126
7.21.3.2	create_category()	126
7.21.3.3	create_default_model()	126
7.21.3.4	create_item()	127
7.21.3.5	delete_category()	127
7.21.3.6	delete_item()	127
7.21.3.7	filter()	128
7.21.3.8	find_category()	129
7.21.3.9	find_item()	129
7.21.3.10	get_categories()	130
7.21.3.11	get_filtered_items()	130
7.21.3.12	get_items()	130
7.21.3.13	get_next_category_id()	131
7.21.3.14	get_next_item_id()	131
7.21.3.15	is_model_dirty()	131
7.21.3.16	load_model()	132
7.21.3.17	open_as_xml()	132
7.21.3.18	require_filename()	132
7.21.3.19	save()	132
7.21.3.20	save_as_xml()	132
7.21.3.21	select_items()	133
7.21.3.22	set_model_dirty()	133

---

7.21.3.23	<a href="#">show_all()</a>	134
7.21.3.24	<a href="#">show_checked_only()</a>	134
7.21.3.25	<a href="#">show_unchecked_only()</a>	134
7.21.3.26	<a href="#">toggle_selected_items()</a>	134
7.21.3.27	<a href="#">update_item()</a>	135
7.21.4	<a href="#">Member Data Documentation</a>	135
7.21.4.1	<a href="#">m_dao</a>	135
7.21.4.2	<a href="#">m_model</a>	136
7.22	<a href="#">TickItem Class Reference</a>	136
7.22.1	<a href="#">Detailed Description</a>	136
7.22.2	<a href="#">Constructor &amp; Destructor Documentation</a>	136
7.22.2.1	<a href="#">TickItem()</a>	137
7.22.3	<a href="#">Member Function Documentation</a>	137
7.22.3.1	<a href="#">operator()()</a>	137
7.22.4	<a href="#">Member Data Documentation</a>	137
7.22.4.1	<a href="#">m_changed</a>	137
7.23	<a href="#">XmlDao Class Reference</a>	137
7.23.1	<a href="#">Detailed Description</a>	139
7.23.2	<a href="#">Constructor &amp; Destructor Documentation</a>	139
7.23.2.1	<a href="#">XmlDao()</a>	139
7.23.3	<a href="#">Member Function Documentation</a>	139
7.23.3.1	<a href="#">add_category_item_to_dom()</a>	139
7.23.3.2	<a href="#">add_category_to_dom()</a>	140
7.23.3.3	<a href="#">add_item() [1/2]</a>	140
7.23.3.4	<a href="#">add_item() [2/2]</a>	140
7.23.3.5	<a href="#">add_item_to_dom()</a>	141
7.23.3.6	<a href="#">append_items_to_category()</a>	141
7.23.3.7	<a href="#">associate_item_with_category()</a>	142
7.23.3.8	<a href="#">delete_category()</a>	142
7.23.3.9	<a href="#">delete_item()</a>	142

---

---

7.23.3.10	<a href="#">get_all_items()</a>	143
7.23.3.11	<a href="#">get_categories()</a>	143
7.23.3.12	<a href="#">get_category()</a>	143
7.23.3.13	<a href="#">get_model() [1/2]</a>	144
7.23.3.14	<a href="#">get_model() [2/2]</a>	144
7.23.3.15	<a href="#">get_next_category_id()</a>	145
7.23.3.16	<a href="#">get_next_item_id()</a>	145
7.23.3.17	<a href="#">new_category()</a>	145
7.23.3.18	<a href="#">remove_item_from_category()</a>	145
7.23.3.19	<a href="#">require_filename()</a>	146
7.23.3.20	<a href="#">save_model() [1/2]</a>	146
7.23.3.21	<a href="#">save_model() [2/2]</a>	147
7.23.3.22	<a href="#">set_all_flags()</a>	147
7.23.3.23	<a href="#">set_category_flag()</a>	147
7.23.3.24	<a href="#">set_filename()</a>	148
7.23.3.25	<a href="#">set_item_flag()</a>	148
7.23.3.26	<a href="#">unset_all_flags()</a>	148
7.23.3.27	<a href="#">unset_category_flag()</a>	148
7.23.3.28	<a href="#">unset_item_flag()</a>	149
7.23.3.29	<a href="#">update_item_checked_state()</a>	149
7.23.4	<a href="#">Member Data Documentation</a>	149
7.23.4.1	<a href="#">m_cat_items_node</a>	149
7.23.4.2	<a href="#">m_categories_node</a>	149
7.23.4.3	<a href="#">m_filename</a>	150
7.23.4.4	<a href="#">m_items_node</a>	150
7.23.4.5	<a href="#">m_max_category_id</a>	150
7.23.4.6	<a href="#">m_max_item_id</a>	150
7.24	<a href="#">YamlConfig Class Reference</a>	150
7.24.1	<a href="#">Detailed Description</a>	151
7.24.2	<a href="#">Constructor &amp; Destructor Documentation</a>	152

---

7.24.2.1	YamlConfig()	152
7.24.2.2	~YamlConfig()	152
7.24.3	Member Function Documentation	152
7.24.3.1	add_recent_filename()	152
7.24.3.2	get_config_filename()	153
7.24.3.3	get_current_filename()	153
7.24.3.4	get_debug_log_filename()	153
7.24.3.5	get_page_title()	154
7.24.3.6	get_recent_filenames()	154
7.24.3.7	load()	154
7.24.3.8	save()	155
7.24.3.9	set_current_filename()	155
7.24.3.10	set_page_title()	155
7.24.4	Member Data Documentation	156
7.24.4.1	m_current_filename	156
7.24.4.2	m_debug_log_filename	156
7.24.4.3	m_max_recent_files	156
7.24.4.4	m_mru_file_history	156
7.24.4.5	m_page_title	157
<b>8</b>	<b>File Documentation</b>	<b>159</b>
8.1	/home/frank/Projects/kitlist/src/category.cpp File Reference	159
8.2	/home/frank/Projects/kitlist/src/category.hpp File Reference	159
8.2.1	Typedef Documentation	159
8.2.1.1	CategoryContainer	160
8.2.1.2	CategoryIter	160
8.3	/home/frank/Projects/kitlist/src/item.hpp File Reference	160
8.3.1	Typedef Documentation	160
8.3.1.1	ItemContainer	161
8.3.1.2	ItemIter	161
8.3.1.3	ItemList	161

---

8.3.1.4	ItemListIter	161
8.3.1.5	SlotForeachItem	161
8.4	/home/frank/Projects/kitlist/src/kitlistdao.hpp File Reference	161
8.4.1	Enumeration Type Documentation	162
8.4.1.1	item_choice	162
8.5	/home/frank/Projects/kitlist/src/kitlistgui.cpp File Reference	162
8.5.1	Typedef Documentation	164
8.5.1.1	type_children	164
8.5.2	Function Documentation	164
8.5.2.1	file_exists()	164
8.5.2.2	get_glade_ref_ptr()	165
8.5.2.3	load_resource_glade_file()	165
8.6	/home/frank/Projects/kitlist/src/kitlistgui.hpp File Reference	165
8.6.1	Enumeration Type Documentation	166
8.6.1.1	gui_state	166
8.6.2	Variable Documentation	166
8.6.2.1	CHECKED_COL_POSITION	166
8.7	/home/frank/Projects/kitlist/src/kitlistpgsqldao.cpp File Reference	167
8.8	/home/frank/Projects/kitlist/src/kitlistpgsqldao.hpp File Reference	167
8.9	/home/frank/Projects/kitlist/src/kitmodel.cpp File Reference	167
8.10	/home/frank/Projects/kitlist/src/kitmodel.hpp File Reference	167
8.10.1	Typedef Documentation	168
8.10.1.1	CategoryMap	168
8.10.1.2	CategoryMapIter	168
8.10.1.3	ItemMap	168
8.10.1.4	ItemMapIter	169
8.10.1.5	ModelCategoryContainer	169
8.10.1.6	ModelCategoryIter	169
8.10.1.7	ModelItemContainer	169
8.10.1.8	ModelItemIter	169

8.10.1.9	SlotForeachCategory	169
8.10.1.10	SlotForeachCategoryIter	170
8.10.1.11	SlotForeachModelItem	170
8.10.1.12	SlotForeachModelItemIter	170
8.10.2	Enumeration Type Documentation	170
8.10.2.1	item_filter_types	170
8.11	/home/frank/Projects/kitlist/src/kitparser.cpp File Reference	170
8.12	/home/frank/Projects/kitlist/src/kitparser.hpp File Reference	171
8.13	/home/frank/Projects/kitlist/src/main.cpp File Reference	171
8.13.1	Function Documentation	171
8.13.1.1	main()	172
8.13.2	Variable Documentation	172
8.13.2.1	html_flag	172
8.13.2.2	verbose_flag	172
8.14	/home/frank/Projects/kitlist/src/printing.cpp File Reference	172
8.14.1	Variable Documentation	173
8.14.1.1	BORDER_SPACING	173
8.14.1.2	FOOTER_SPACING	173
8.14.1.3	FOOTER_TEXT	173
8.14.1.4	HEADER_SPACING	174
8.14.1.5	PAGE_TOLERANCE	174
8.15	/home/frank/Projects/kitlist/src/printing.hpp File Reference	174
8.15.1	Typedef Documentation	174
8.15.1.1	layout_refptr	174
8.16	/home/frank/Projects/kitlist/src/service.cpp File Reference	175
8.17	/home/frank/Projects/kitlist/src/service.hpp File Reference	175
8.18	/home/frank/Projects/kitlist/src/xmldao.cpp File Reference	175
8.19	/home/frank/Projects/kitlist/src/xmldao.hpp File Reference	175
8.19.1	Macro Definition Documentation	176
8.19.1.1	NYI	176
8.19.1.2	XMLDAO_H	176
8.20	/home/frank/Projects/kitlist/src/yamlconfig.cpp File Reference	176
8.20.1	Variable Documentation	177
8.20.1.1	CONFIG_FILENAME	177
8.20.1.2	CURRENT_FILENAME_CONFIG_KEY	177
8.20.1.3	DEBUG_LOG_FILENAME_CONFIG_KEY	177
8.20.1.4	MAX_RECENT_FILES_CONFIG_KEY	177
8.20.1.5	PAGE_TITLE_CONFIG_KEY	178
8.20.1.6	RECENT_FILES_CONFIG_KEY	178
8.21	/home/frank/Projects/kitlist/src/yamlconfig.hpp File Reference	178
8.21.1	Variable Documentation	178
8.21.1.1	DEFAULT_MAX_RECENT_FILES	179
8.21.1.2	DEFAULT_PAGE_TITLE	179







# Chapter 1

## Kitlist Documentation

### 1.1 Introduction

The kitlist program has been developed to be run on multiple platforms. It is known to run on the following platforms:

- Debian versions 5 (Lenny) through to 10 (Buster)
- macOS 11.3 (Big Sur) and many previous versions via <https://macports.org/>
- Windows XP (but no longer supported)

It can be compiled to use either a PostgreSQL database or XML documents as the data store (Debian Linux only). When compiled to use PostgreSQL, the program can be used from the command line without a GUI.

Where the program is executed without any arguments, the GUI is shown. If there are any arguments, the command line version is executed unless the '-g' option is specified to force running the GUI.

### 1.2 Additional Documentation

See the README in the root of the source code for information on building the application.



# Chapter 2

## Namespace Index

### 2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

[anonymous\\_namespace{kitlistgui.cpp}](#) . . . . . 11



## Chapter 3

# Hierarchical Index

### 3.1 Class Hierarchy

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

Category . . . . .	17
ModelCategory . . . . .	112
CategoryCompareId . . . . .	23
CategoryCompareName . . . . .	24
ColumnRecord	
ModelCategoryColumns . . . . .	118
ModelItemColumns . . . . .	121
GuiState . . . . .	26
ModelCategory . . . . .	112
ModelItem . . . . .	119
Item . . . . .	30
ModelItem . . . . .	119
ItemCompareId . . . . .	34
ItemCompareName . . . . .	35
ItemFunctor . . . . .	36
FilterItem . . . . .	25
TickItem . . . . .	136
KitList . . . . .	37
KitListDao . . . . .	48
XmlDao . . . . .	137
KitListGui . . . . .	58
KitModel . . . . .	91
ModelItemCompareId . . . . .	122
PrintOperation	
KitPrintOperation . . . . .	107
SaxParser	
KitParser . . . . .	100
Service . . . . .	123
YamlConfig . . . . .	150



# Chapter 4

## Class Index

### 4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">Category</a>	Represents a <a href="#">Category</a> . . . . .	17
<a href="#">CategoryCompareId</a>	Comparator used for comparing Categories by id . . . . .	23
<a href="#">CategoryCompareName</a>	Comparator used for sorting Categories by name . . . . .	24
<a href="#">FilterItem</a>	. . . . .	25
<a href="#">GuiState</a>	Class encapsulating state of an object in the data model . . . . .	26
<a href="#">Item</a>	Represents an <a href="#">Item</a> . . . . .	30
<a href="#">ItemCompareId</a>	Comparator used for comparing Items by id . . . . .	34
<a href="#">ItemCompareName</a>	Comparator used for sorting Items by name . . . . .	35
<a href="#">ItemFunctor</a>	Functor for processing items . . . . .	36
<a href="#">KitList</a>	Main application class . . . . .	37
<a href="#">KitListDao</a>	Defines the methods that an implementation of this class must implement . . . . .	48
<a href="#">KitListGui</a>	Encapsulates the methods for the application's GUI front end . . . . .	58
<a href="#">KitModel</a>	Holds a rich graph of objects representing the application's data model . . . . .	91
<a href="#">KitParser</a>	SaxParser implementation for reading the <a href="#">KitModel</a> from an XML document . . . . .	100
<a href="#">KitPrintOperation</a>	Prints the kitlist . . . . .	107
<a href="#">ModelCategory</a>	Represents a <a href="#">Category</a> combined with <a href="#">GuiState</a> attributes . . . . .	112
<a href="#">ModelCategoryColumns</a>	A definition for displaying a <a href="#">ModelCategory</a> in a combo box . . . . .	118
<a href="#">ModelItem</a>	Represents an <a href="#">Item</a> combined with <a href="#">GuiState</a> attributes . . . . .	119

---

<a href="#">ModellItemColumns</a>	A definition for displaying an item in a multi-column list . . . . .	121
<a href="#">ModellItemCompareId</a>	Comparator for comparing items by their unique ID . . . . .	122
<a href="#">Service</a>	Business/service layer implementation . . . . .	123
<a href="#">TickItem</a>	. . . . .	136
<a href="#">XmlDao</a>	Implementation of a <a href="#">KitListDao</a> using XML as the persistence store . . . . .	137
<a href="#">YamlConfig</a>	Maintains the application's configuration parameters in a YAML formatted file . . . . .	150



# Chapter 5

## File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

<a href="#">/home/frank/Projects/kitlist/src/category.cpp</a>	159
<a href="#">/home/frank/Projects/kitlist/src/category.hpp</a>	159
<a href="#">/home/frank/Projects/kitlist/src/item.hpp</a>	160
<a href="#">/home/frank/Projects/kitlist/src/kitlistdao.hpp</a>	161
<a href="#">/home/frank/Projects/kitlist/src/kitlistgui.cpp</a>	162
<a href="#">/home/frank/Projects/kitlist/src/kitlistgui.hpp</a>	165
<a href="#">/home/frank/Projects/kitlist/src/kitlistpgsqldao.cpp</a>	167
<a href="#">/home/frank/Projects/kitlist/src/kitlistpgsqldao.hpp</a>	167
<a href="#">/home/frank/Projects/kitlist/src/kitmodel.cpp</a>	167
<a href="#">/home/frank/Projects/kitlist/src/kitmodel.hpp</a>	167
<a href="#">/home/frank/Projects/kitlist/src/kitparser.cpp</a>	170
<a href="#">/home/frank/Projects/kitlist/src/kitparser.hpp</a>	171
<a href="#">/home/frank/Projects/kitlist/src/main.cpp</a>	171
<a href="#">/home/frank/Projects/kitlist/src/printing.cpp</a>	172
<a href="#">/home/frank/Projects/kitlist/src/printing.hpp</a>	174
<a href="#">/home/frank/Projects/kitlist/src/service.cpp</a>	175
<a href="#">/home/frank/Projects/kitlist/src/service.hpp</a>	175
<a href="#">/home/frank/Projects/kitlist/src/xml dao.cpp</a>	175
<a href="#">/home/frank/Projects/kitlist/src/xml dao.hpp</a>	175
<a href="#">/home/frank/Projects/kitlist/src/yamlconfig.cpp</a>	176
<a href="#">/home/frank/Projects/kitlist/src/yamlconfig.hpp</a>	178



## Chapter 6

# Namespace Documentation

### 6.1 anonymous\_namespace{kitlistgui.cpp} Namespace Reference

#### Variables

- const string `GLADE_APP_FILE` = "kitlist.glade"  
*Resource file name.*
- const guint `SB_ITEM_COUNT` = 1000  
*Status bar message constant for displaying item counts.*
- const guint `SB_SAVE` = `SB_ITEM_COUNT` + 1  
*Status bar message constant for save notifications.*
- const guint `SB_MSG` = `SB_SAVE` + 1  
*Status bar message constant for general messages.*
- const guint `SB_PRINT` = `SB_MSG` + 1  
*Status bar message constant for printer messages.*
- const char `item_target_custom` [] = "kitlistclipboard"  
*Key used for custom clipboard.*
- const char `item_target_text` [] = "text/plain"  
*Mime type for clipboard content.*
- const char `XML_ELEMENT_ID` [] = "id"  
*Tag name for the ID element in the clipboard XML document.*
- const Glib::ustring `DEFAULT_FILENAME_EXTENSION` = ".kit"  
*Default filename extension.*
- const Glib::ustring `PDF_FILENAME_EXTENSION` = ".pdf"  
*PDF filename extension.*
- const Glib::ustring `DEFAULT_FILENAME` = "kitlist" + `DEFAULT_FILENAME_EXTENSION`  
*The default filename.*
- const Glib::ustring `GCONF_KEY` = "/apps/kitlist"  
*The application's root key in the GConf hierarchy.*
- const Glib::ustring `GCONF_KEY_CURRENT_FILENAME` = `GCONF_KEY` + "/current\_filename"  
*GConf entry for the current filename.*
- const Glib::ustring `GCONF_KEY_PAGE_TITLE` = `GCONF_KEY` + "/page\_title"  
*GConf entry for the page title.*
- const Glib::ustring `GCONF_KEY_RECENT_FILES` = `GCONF_KEY` + "/recent\_files"  
*GConf entry for recent files.*
- const Glib::ustring `GCONF_KEY_MAX_RECENT_FILES` = `GCONF_KEY` + "/max\_recent\_files"  
*GConf entry for max recent files.*

## 6.1.1 Variable Documentation

### 6.1.1.1 DEFAULT\_FILENAME

```
const Glib::ustring anonymous_namespace{kitlistgui.cpp}::DEFAULT_FILENAME = "kitlist" + DEFAULT_FILENAME_EXTENSION
```

The default filename.

Definition at line 83 of file kitlistgui.cpp.

Referenced by KitListGui::init().

### 6.1.1.2 DEFAULT\_FILENAME\_EXTENSION

```
const Glib::ustring anonymous_namespace{kitlistgui.cpp}::DEFAULT_FILENAME_EXTENSION = ".kit"
```

Default filename extension.

Definition at line 77 of file kitlistgui.cpp.

Referenced by KitListGui::choose\_filename(), and KitListGui::on\_menu\_file\_open().

### 6.1.1.3 GCONF\_KEY

```
const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY = "/apps/kitlist"
```

The application's root key in the GConf hierarchy.

Definition at line 86 of file kitlistgui.cpp.

Referenced by KitListGui::init().

### 6.1.1.4 GCONF\_KEY\_CURRENT\_FILENAME

```
const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_CURRENT_FILENAME = GCONF_KEY + "/current_filename"
```

GConf entry for the current filename.

Definition at line 89 of file kitlistgui.cpp.

Referenced by KitListGui::init(), KitListGui::on\_menu\_file\_new(), KitListGui::on\_menu\_recent\_file(), KitListGui::on\_menu\_save(), KitListGui::on\_menu\_save\_as(), and KitListGui::open\_file().

#### 6.1.1.5 GCONF\_KEY\_MAX\_RECENT\_FILES

```
const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_MAX_RECENT_FILES = GCONF_KEY +  
"/max_recent_files"
```

GConf entry for max recent files.

Definition at line 105 of file kitlistgui.cpp.

#### 6.1.1.6 GCONF\_KEY\_PAGE\_TITLE

```
const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_PAGE_TITLE = GCONF_KEY +  
"/page_title"
```

GConf entry for the page title.

Definition at line 92 of file kitlistgui.cpp.

Referenced by KitListGui::init(), and KitListGui::set\_page\_title().

#### 6.1.1.7 GCONF\_KEY\_RECENT\_FILES

```
const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_RECENT_FILES = GCONF_KEY +  
"/recent_files"
```

GConf entry for recent files.

Definition at line 102 of file kitlistgui.cpp.

Referenced by KitListGui::update\_recent\_files(), and KitListGui::update\_recent\_files\_menu().

#### 6.1.1.8 GLADE\_APP\_FILE

```
const string anonymous_namespace{kitlistgui.cpp}::GLADE_APP_FILE = "kitlist.glade"
```

Resource file name.

Definition at line 56 of file kitlistgui.cpp.

Referenced by KitListGui::init().

#### 6.1.1.9 item\_target\_custom

```
const char anonymous_namespace{kitlistgui.cpp}::item_target_custom[] = "kitlistclipboard"
```

Key used for custom clipboard.

Definition at line 69 of file kitlistgui.cpp.

Referenced by KitListGui::copy\_selected\_items\_to\_clipboard(), KitListGui::on\_clipboard\_get(), KitListGui::on\_↔clipboard\_received(), and KitListGui::paste\_status\_received().

#### 6.1.1.10 item\_target\_text

```
const char anonymous_namespace{kitlistgui.cpp}::item_target_text[] = "text/plain"
```

Mime type for clipboard content.

Definition at line 71 of file kitlistgui.cpp.

Referenced by KitListGui::copy\_selected\_items\_to\_clipboard(), KitListGui::on\_clipboard\_get(), KitListGui::on\_↔clipboard\_received(), KitListGui::on\_menu\_paste(), and KitListGui::paste\_status\_received().

#### 6.1.1.11 PDF\_FILENAME\_EXTENSION

```
const Glib::ustring anonymous_namespace{kitlistgui.cpp}::PDF_FILENAME_EXTENSION = ".pdf"
```

PDF filename extension.

Definition at line 80 of file kitlistgui.cpp.

Referenced by KitListGui::choose\_pdf\_filename().

#### 6.1.1.12 SB\_ITEM\_COUNT

```
const quint anonymous_namespace{kitlistgui.cpp}::SB_ITEM_COUNT = 1000
```

Status bar message constant for displaying item counts.

Definition at line 59 of file kitlistgui.cpp.

Referenced by KitListGui::update\_item\_count().

#### 6.1.1.13 SB\_MSG

```
const quint anonymous_namespace{kitlistgui.cpp}::SB_MSG = SB_SAVE + 1
```

Status bar message constant for general messages.

Definition at line 63 of file kitlistgui.cpp.

Referenced by KitListGui::on\_menu\_cut(), KitListGui::on\_menu\_delete\_category(), and KitListGui::on\_menu\_rename\_category().

#### 6.1.1.14 SB\_PRINT

```
const quint anonymous_namespace{kitlistgui.cpp}::SB_PRINT = SB_MSG + 1
```

Status bar message constant for printer messages.

Definition at line 66 of file kitlistgui.cpp.

Referenced by KitListGui::on\_printoperation\_status\_changed().

#### 6.1.1.15 SB\_SAVE

```
const quint anonymous_namespace{kitlistgui.cpp}::SB_SAVE = SB_ITEM_COUNT + 1
```

Status bar message constant for save notifications.

Definition at line 61 of file kitlistgui.cpp.

Referenced by KitListGui::on\_menu\_export\_to\_pdf(), KitListGui::on\_menu\_file\_new(), KitListGui::on\_menu\_file\_open(), KitListGui::on\_menu\_save(), KitListGui::on\_menu\_save\_as(), and KitListGui::safe\_open\_file().

#### 6.1.1.16 XML\_ELEMENT\_ID

```
const char anonymous_namespace{kitlistgui.cpp}::XML_ELEMENT_ID[] = "id"
```

Tag name for the ID element in the clipboard XML document.

Definition at line 74 of file kitlistgui.cpp.

Referenced by KitListGui::copy\_selected\_items\_to\_clipboard(), and KitListGui::paste\_from\_xml().





# Chapter 7

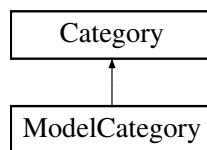
## Class Documentation

### 7.1 Category Class Reference

Represents a [Category](#).

```
#include <category.hpp>
```

Inheritance diagram for Category:



#### Public Member Functions

- [~Category](#) ()
- void [set\\_id](#) (long id)
- long [get\\_id](#) ()
- void [set\\_name](#) (const std::string name)
- std::string [get\\_name](#) ()
- virtual void [add\\_item](#) (Item \*item)  
*Associates the passed item with this [Category](#).*
- virtual void [remove\\_item](#) (Item \*item)  
*Removes the association of the passed item from this [Category](#).*
- virtual size\_t [item\\_count](#) ()  
*Returns the number of items associated with this category.*
- virtual bool [has\\_items](#) ()  
*Returns true if there are any items associated with this category.*
- void [foreach\\_item](#) (const SlotForeachItem &slot)  
*Executes a callback function for each associated item.*
- void [execute](#) (ItemFuncor &funcor)  
*Executes the passed [ItemFuncor](#).*

## Protected Attributes

- long [m\\_id](#)  
*Unique id.*
- std::string [m\\_name](#)  
*The category name.*
- [ItemContainer m\\_items](#)  
*List of associated items.*

## Friends

- class [CategoryCompareName](#)
- class [CategoryCompareId](#)
- class [KitModel](#)

### 7.1.1 Detailed Description

Represents a [Category](#).

Many categories may have one or more items.

Definition at line 37 of file `category.hpp`.

### 7.1.2 Constructor & Destructor Documentation

#### 7.1.2.1 `~Category()`

```
Category::~Category ( ) [inline]
```

Definition at line 43 of file `category.hpp`.

### 7.1.3 Member Function Documentation

#### 7.1.3.1 `add_item()`

```
void Category::add_item (
    Item * item ) [virtual]
```

Associates the passed item with this [Category](#).

Reimplemented in [ModelCategory](#).

Definition at line 29 of file `category.cpp`.

References `m_items`.

Referenced by `ModelCategory::add_item()`, and `get_name()`.

### 7.1.3.2 execute()

```
void Category::execute (
    ItemFuncutor & functor )
```

Executes the passed [ItemFuncutor](#).

The [ItemFuncutor](#)'s override operator() method is called, passing a reference to the item being iterated over. If the called method returns true, the iteration stops and no more calls will be made to the functor.

Definition at line 71 of file category.cpp.

References [m\\_items](#).

Referenced by [has\\_items\(\)](#), and [KitList::tick\\_items\(\)](#).

### 7.1.3.3 foreach\_item()

```
void Category::foreach_item (
    const SlotForeachItem & slot )
```

Executes a callback function for each associated item.

The callback function will be passed a reference to the current item being iterated.

#### Parameters

<i>slot</i>	the callback function.
-------------	------------------------

Definition at line 55 of file category.cpp.

References [m\\_items](#).

Referenced by [XmlDao::add\\_category\\_to\\_dom\(\)](#), [has\\_items\(\)](#), and [KitList::list\\_items\(\)](#).

### 7.1.3.4 get\_id()

```
long Category::get_id ( ) [inline]
```

Definition at line 45 of file category.hpp.

References [m\\_id](#).

Referenced by [KitModel::add\\_category\(\)](#), [XmlDao::add\\_category\\_to\\_dom\(\)](#), [KitListGui::close\\_add\\_category\\_↔ window\(\)](#), [XmlDao::get\\_model\(\)](#), [KitParser::process\\_category\\_item\(\)](#), and [KitListGui::refresh\\_category\\_list\(\)](#).

### 7.1.3.5 get\_name()

```
std::string Category::get_name ( ) [inline]
```

Definition at line 47 of file category.hpp.

References `add_item()`, `m_name`, and `remove_item()`.

Referenced by `XmlDao::add_category_to_dom()`, `KitListGui::on_menu_rename_category()`, and `KitListGui::refresh_category_list()`.

### 7.1.3.6 has\_items()

```
virtual bool Category::has_items ( ) [inline], [virtual]
```

Returns true if there are any items associated with this category.

Definition at line 53 of file category.hpp.

References `execute()`, and `foreach_item()`.

Referenced by `KitList::list_items()`.

### 7.1.3.7 item\_count()

```
virtual size_t Category::item_count ( ) [inline], [virtual]
```

Returns the number of items associated with this category.

Definition at line 51 of file category.hpp.

Referenced by `KitList::list_items()`.

### 7.1.3.8 remove\_item()

```
void Category::remove_item (
    Item * item ) [virtual]
```

Removes the association of the passed item from this [Category](#).

The passed item is not deleted.

#### Parameters

<i>item</i>	the item to un-associate.
-------------	---------------------------

Reimplemented in [ModelCategory](#).

Definition at line 40 of file category.cpp.

References `m_items`.

Referenced by `get_name()`, and `ModelCategory::remove_item()`.

#### 7.1.3.9 `set_id()`

```
void Category::set_id (
    long id ) [inline]
```

Definition at line 44 of file category.hpp.

Referenced by `Service::create_category()`, and `KitParser::process_category()`.

#### 7.1.3.10 `set_name()`

```
void Category::set_name (
    const std::string name ) [inline]
```

Definition at line 46 of file category.hpp.

Referenced by `KitListGui::close_add_category_window()`, and `KitParser::on_end_element()`.

### 7.1.4 Friends And Related Function Documentation

#### 7.1.4.1 `CategoryCompareId`

```
friend class CategoryCompareId [friend]
```

Definition at line 57 of file category.hpp.

#### 7.1.4.2 `CategoryCompareName`

```
friend class CategoryCompareName [friend]
```

Definition at line 56 of file category.hpp.

### 7.1.4.3 KitModel

```
friend class KitModel [friend]
```

Definition at line 58 of file category.hpp.

## 7.1.5 Member Data Documentation

### 7.1.5.1 m\_id

```
long Category::m_id [protected]
```

Unique id.

Definition at line 39 of file category.hpp.

Referenced by `get_id()`, and `CategoryCompareId::operator()`.

### 7.1.5.2 m\_items

```
ItemContainer Category::m_items [protected]
```

List of associated items.

Definition at line 41 of file category.hpp.

Referenced by `add_item()`, `KitModel::copy_items()`, `execute()`, `foreach_item()`, `ModelCategory::get_items()`, `ModelCategory::get_model_items()`, `ModelCategory::purge()`, and `remove_item()`.

### 7.1.5.3 m\_name

```
std::string Category::m_name [protected]
```

The category name.

Definition at line 40 of file category.hpp.

Referenced by `get_name()`, and `CategoryCompareName::operator()`.

The documentation for this class was generated from the following files:

- [/home/frank/Projects/kitlist/src/category.hpp](#)
- [/home/frank/Projects/kitlist/src/category.cpp](#)

## 7.2 CategoryCompareId Class Reference

Comparator used for comparing Categories by id.

```
#include <category.hpp>
```

### Public Member Functions

- [CategoryCompareId](#) ()
- [int operator\(\)](#) ([Category](#) \*c1, [Category](#) \*c2)

### 7.2.1 Detailed Description

Comparator used for comparing Categories by id.

See also

[Category](#)

Definition at line 77 of file category.hpp.

### 7.2.2 Constructor & Destructor Documentation

#### 7.2.2.1 CategoryCompareId()

```
CategoryCompareId::CategoryCompareId ( ) [inline]
```

Definition at line 79 of file category.hpp.

### 7.2.3 Member Function Documentation

#### 7.2.3.1 operator()

```
int CategoryCompareId::operator() (
    Category * c1,
    Category * c2 ) [inline]
```

Definition at line 80 of file category.hpp.

References [Category::m\\_id](#).

The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/category.hpp](#)

## 7.3 CategoryCompareName Class Reference

Comparator used for sorting Categories by name.

```
#include <category.hpp>
```

### Public Member Functions

- [CategoryCompareName](#) ()
- [int operator\(\)](#) ([Category](#) \*c1, [Category](#) \*c2)

#### 7.3.1 Detailed Description

Comparator used for sorting Categories by name.

See also

[Category](#)

Definition at line 66 of file category.hpp.

#### 7.3.2 Constructor & Destructor Documentation

##### 7.3.2.1 CategoryCompareName()

```
CategoryCompareName::CategoryCompareName ( ) [inline]
```

Definition at line 68 of file category.hpp.

#### 7.3.3 Member Function Documentation

##### 7.3.3.1 operator()

```
int CategoryCompareName::operator() (
    Category * c1,
    Category * c2 ) [inline]
```

Definition at line 69 of file category.hpp.

References [Category::m\\_name](#).

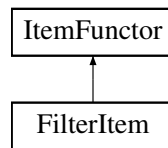
The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/category.hpp](#)



## 7.4 FilterItem Class Reference

Inheritance diagram for FilterItem:



### Public Member Functions

- [FilterItem](#) ([KitModel](#) &model)
- bool [operator\(\)](#) ([Item](#) &item)

### Private Attributes

- [KitModel](#) & [m\\_model](#)

#### 7.4.1 Detailed Description

Definition at line 30 of file `service.cpp`.

#### 7.4.2 Constructor & Destructor Documentation

##### 7.4.2.1 FilterItem()

```
FilterItem::FilterItem (  
    KitModel & model ) [inline]
```

Definition at line 33 of file `service.cpp`.

#### 7.4.3 Member Function Documentation

##### 7.4.3.1 operator()

```
bool FilterItem::operator() (  
    Item & item ) [inline], [virtual]
```

Implements [ItemFunctor](#).

Definition at line 34 of file `service.cpp`.

References [KitModel::filter\(\)](#), and [Item::get\\_checked\(\)](#).

## 7.4.4 Member Data Documentation

### 7.4.4.1 m\_model

```
KitModel& FilterItem::m_model [private]
```

Definition at line 31 of file service.cpp.

The documentation for this class was generated from the following file:

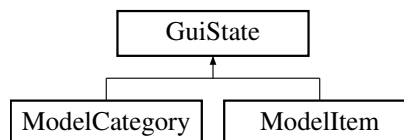
- </home/frank/Projects/kitlist/src/service.cpp>

## 7.5 GuiState Class Reference

Class encapsulating state of an object in the data model.

```
#include <kitmodel.hpp>
```

Inheritance diagram for GuiState:



### Public Member Functions

- [GuiState \(\)](#)
- [bool is\\_dirty \(\)](#)
- [void set\\_dirty \(bool dirty=true\)](#)
- [bool is\\_deleted \(\)](#)
- [void set\\_deleted \(bool deleted\)](#)
- [void set\\_new\\_flag \(bool flag\)](#)
- [bool is\\_new \(\)](#)
- [virtual void reset \(\)](#)

*resets the state of each flag to it's default.*

### Protected Attributes

- [bool m\\_dirty](#)
- [bool m\\_deleted](#)
- [bool m\\_new](#)

### 7.5.1 Detailed Description

Class encapsulating state of an object in the data model.

Provides additional flags to identify whether the object is dirty, has been deleted or is new.

Definition at line 38 of file kitmodel.hpp.

### 7.5.2 Constructor & Destructor Documentation

#### 7.5.2.1 GuiState()

```
GuiState::GuiState ( ) [inline]
```

Definition at line 44 of file kitmodel.hpp.

### 7.5.3 Member Function Documentation

#### 7.5.3.1 is\_deleted()

```
bool GuiState::is_deleted ( ) [inline]
```

Definition at line 47 of file kitmodel.hpp.

References `m_deleted`.

Referenced by `XmlDao::add_category_to_dom()`, `XmlDao::add_item_to_dom()`, `KitModel::get_categories()`, `ModelCategory::get_items()`, `ModelCategory::get_model_items()`, `KitModel::purge()`, `KitListGui::refresh_category_list()`, and `KitModel::reset()`.

#### 7.5.3.2 is\_dirty()

```
bool GuiState::is_dirty ( ) [inline]
```

Definition at line 45 of file kitmodel.hpp.

References `m_dirty`.

### 7.5.3.3 is\_new()

```
bool GuiState::is_new ( ) [inline]
```

Definition at line 50 of file kitmodel.hpp.

References `m_new`, and `reset()`.

### 7.5.3.4 reset()

```
void GuiState::reset ( ) [virtual]
```

resets the state of each flag to it's default.

The dirty, deleted and new flags are set to false.

Reimplemented in [ModelCategory](#).

Definition at line 36 of file kitmodel.cpp.

References `m_deleted`, `m_dirty`, and `m_new`.

Referenced by `ModelCategory::get_added_children()`, `is_new()`, `ModelCategory::reset()`, `KitModel::reset()`, and `KitModel::show_unchecked_only()`.

### 7.5.3.5 set\_deleted()

```
void GuiState::set_deleted (
    bool deleted ) [inline]
```

Definition at line 48 of file kitmodel.hpp.

Referenced by `Service::delete_category()`, and `Service::delete_item()`.

### 7.5.3.6 set\_dirty()

```
void GuiState::set_dirty (
    bool dirty = true ) [inline]
```

Definition at line 46 of file kitmodel.hpp.

Referenced by `KitModel::copy_items()`, `Service::create_category()`, `Service::create_item()`, `Service::delete_category()`, `Service::delete_item()`, `KitListGui::on_menu_cut()`, `ModelItem::set_checked()`, and `Service::update_item()`.

### 7.5.3.7 set\_new\_flag()

```
void GuiState::set_new_flag (  
    bool flag ) [inline]
```

Definition at line 49 of file kitmodel.hpp.

Referenced by Service::create\_category(), and Service::create\_item().

## 7.5.4 Member Data Documentation

### 7.5.4.1 m\_deleted

```
bool GuiState::m_deleted [protected]
```

Definition at line 41 of file kitmodel.hpp.

Referenced by is\_deleted(), and reset().

### 7.5.4.2 m\_dirty

```
bool GuiState::m_dirty [protected]
```

Definition at line 40 of file kitmodel.hpp.

Referenced by is\_dirty(), KitModel::is\_dirty(), and reset().

### 7.5.4.3 m\_new

```
bool GuiState::m_new [protected]
```

Definition at line 42 of file kitmodel.hpp.

Referenced by is\_new(), and reset().

The documentation for this class was generated from the following files:

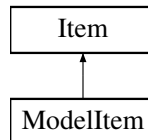
- [/home/frank/Projects/kitlist/src/kitmodel.hpp](#)
- [/home/frank/Projects/kitlist/src/kitmodel.cpp](#)

## 7.6 Item Class Reference

Represents an [Item](#).

```
#include <item.hpp>
```

Inheritance diagram for Item:



### Public Member Functions

- [Item](#) ()
- [Item](#) (const [Item](#) &i)  
*Creates a copy of this item based on the passed item.*
- void [set\\_id](#) (long id)
- long [get\\_id](#) ()
- void [set\\_description](#) (const std::string description)
- std::string [get\\_description](#) ()
- virtual void [set\\_checked](#) (bool checked)
- bool [get\\_checked](#) ()

### Private Attributes

- long [m\\_id](#)  
*Unique ID.*
- std::string [m\\_desc](#)  
*The item's description.*
- bool [m\\_checked](#)  
*Whether checked/ticked or not.*

### Friends

- class [ItemCompareName](#)
- class [ItemCompareId](#)
- std::ostream & [operator<<](#) (std::ostream &os, const [Item](#) &i)

### 7.6.1 Detailed Description

Represents an [Item](#).

Definition at line 37 of file item.hpp.

## 7.6.2 Constructor & Destructor Documentation

### 7.6.2.1 Item() [1/2]

```
Item::Item ( ) [inline]
```

Definition at line 42 of file item.hpp.

### 7.6.2.2 Item() [2/2]

```
Item::Item (
    const Item & i ) [inline]
```

Creates a copy of this item based on the passed item.

Definition at line 44 of file item.hpp.

## 7.6.3 Member Function Documentation

### 7.6.3.1 get\_checked()

```
bool Item::get_checked ( ) [inline]
```

Definition at line 50 of file item.hpp.

References `m_checked`.

Referenced by `XmlDao::add_item_to_dom()`, `FilterItem::operator()()`, and `TickItem::operator()()`.

### 7.6.3.2 get\_description()

```
std::string Item::get_description ( ) [inline]
```

Definition at line 48 of file item.hpp.

References `m_desc`.

Referenced by `XmlDao::add_item_to_dom()`, `KitList::list_item()`, `KitPrintOperation::on_begin_print()`, and `TickItem::operator()()`.

### 7.6.3.3 get\_id()

```
long Item::get_id ( ) [inline]
```

Definition at line 46 of file item.hpp.

References [m\\_id](#).

Referenced by [XmlDao::add\\_category\\_item\\_to\\_dom\(\)](#), [ModelCategory::add\\_item\(\)](#), [KitModel::add\\_item\(\)](#), [XmlDao::add\\_item\\_to\\_dom\(\)](#), [XmlDao::get\\_model\(\)](#), [KitList::list\\_item\(\)](#), [ModellItemCompareId::operator\(\)](#), [TickItem::operator\(\)](#), and [ModelCategory::remove\\_item\(\)](#).

### 7.6.3.4 set\_checked()

```
virtual void Item::set_checked (
    bool checked ) [inline], [virtual]
```

Reimplemented in [ModellItem](#).

Definition at line 49 of file item.hpp.

Referenced by [KitListGui::close\\_add\\_item\\_window\(\)](#), [TickItem::operator\(\)](#), and [ModellItem::set\\_checked\(\)](#).

### 7.6.3.5 set\_description()

```
void Item::set_description (
    const std::string description ) [inline]
```

Definition at line 47 of file item.hpp.

Referenced by [KitListGui::close\\_add\\_item\\_window\(\)](#), [KitParser::on\\_end\\_element\(\)](#), and [Service::update\\_item\(\)](#).

### 7.6.3.6 set\_id()

```
void Item::set_id (
    long id ) [inline]
```

Definition at line 45 of file item.hpp.

Referenced by [Service::create\\_item\(\)](#), and [KitParser::process\\_item\(\)](#).

## 7.6.4 Friends And Related Function Documentation



#### 7.6.4.1 ItemCompareId

```
friend class ItemCompareId [friend]
```

Definition at line 52 of file item.hpp.

#### 7.6.4.2 ItemCompareName

```
friend class ItemCompareName [friend]
```

Definition at line 51 of file item.hpp.

#### 7.6.4.3 operator<<

```
std::ostream& operator<< (  
    std::ostream & os,  
    const Item & i ) [friend]
```

Definition at line 53 of file item.hpp.

### 7.6.5 Member Data Documentation

#### 7.6.5.1 m\_checked

```
bool Item::m_checked [private]
```

Whether checked/ticked or not.

Definition at line 40 of file item.hpp.

Referenced by `get_checked()`.

#### 7.6.5.2 m\_desc

```
std::string Item::m_desc [private]
```

The item's description.

Definition at line 39 of file item.hpp.

Referenced by `get_description()`, and `ItemCompareName::operator()`.

### 7.6.5.3 m\_id

```
long Item::m_id [private]
```

Unique ID.

Definition at line 38 of file item.hpp.

Referenced by `get_id()`, and `ItemCompareId::operator()()`.

The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/item.hpp](#)

## 7.7 ItemCompareId Class Reference

Comparator used for comparing Items by id.

```
#include <item.hpp>
```

### Public Member Functions

- [ItemCompareId \(\)](#)
- [int operator\(\) \(Item \\*i1, Item \\*i2\)](#)

#### 7.7.1 Detailed Description

Comparator used for comparing Items by id.

See also

[Item](#)

Definition at line 72 of file item.hpp.

#### 7.7.2 Constructor & Destructor Documentation

##### 7.7.2.1 ItemCompareId()

```
ItemCompareId::ItemCompareId ( ) [inline]
```

Definition at line 74 of file item.hpp.

### 7.7.3 Member Function Documentation

#### 7.7.3.1 operator()

```
int ItemCompareId::operator() (  
    Item * i1,  
    Item * i2 ) [inline]
```

Definition at line 75 of file item.hpp.

References [Item::m\\_id](#).

The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/item.hpp](#)

## 7.8 ItemCompareName Class Reference

Comparator used for sorting Items by name.

```
#include <item.hpp>
```

### Public Member Functions

- [ItemCompareName](#) ()
- int [operator\(\)](#) ([Item](#) \*i1, [Item](#) \*i2)

#### 7.8.1 Detailed Description

Comparator used for sorting Items by name.

See also

[Item](#)

Definition at line 61 of file item.hpp.

#### 7.8.2 Constructor & Destructor Documentation

### 7.8.2.1 ItemCompareName()

```
ItemCompareName::ItemCompareName ( ) [inline]
```

Definition at line 63 of file item.hpp.

## 7.8.3 Member Function Documentation

### 7.8.3.1 operator()

```
int ItemCompareName::operator() (
    Item * i1,
    Item * i2 ) [inline]
```

Definition at line 64 of file item.hpp.

References Item::m\_desc.

The documentation for this class was generated from the following file:

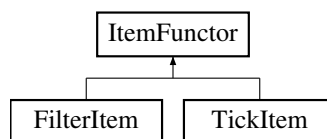
- </home/frank/Projects/kitlist/src/item.hpp>

## 7.9 ItemFunctor Class Reference

Functor for processing items.

```
#include <item.hpp>
```

Inheritance diagram for ItemFunctor:



### Public Member Functions

- virtual bool [operator\(\)](#) (Item &item)=0

### 7.9.1 Detailed Description

Functor for processing items.

Intended to have the operator() overridden by an actual implementation.

Definition at line 85 of file item.hpp.

## 7.9.2 Member Function Documentation

### 7.9.2.1 operator()

```
virtual bool ItemFuncor::operator() (
    Item & item ) [pure virtual]
```

Implemented in [TickItem](#), and [FilterItem](#).

The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/item.hpp](#)

## 7.10 KitList Class Reference

Main application class.

### Public Member Functions

- [KitList](#) (const string dbname, const string user, const string port, bool verbose=false)  
*Constructor specifying Postgresql database connection parameters.*
- [~KitList](#) ()
- [KitListDao](#) \* [get\\_dao](#) ()
- void [add\\_item](#) (const string name)
- void [add\\_item](#) (const string name, long cat\_id)
- void [append\\_items\\_to\\_category](#) (long from\_cat\_id, long to\_cat\_id, [item\\_choice](#) choice)  
*Copies items from one category to another.*
- void [associate\\_item\\_with\\_category](#) (long id, long cat\_id)  
*Associates an existing item with an existing category.*
- void [list\\_items\\_start](#) (bool empty\_list)  
*Called before writing a list of items to STDOUT.*
- void [list\\_item](#) (Item &item)  
*Outputs details of the passed item to STDOUT.*
- void [list\\_items\\_end](#) (bool empty\_list, size\_t count)  
*Called after writing a list of items to STDOUT.*
- void [list\\_items](#) (Category &c)
- bool [on\\_list\\_item](#) (Item &item)
- void [list\\_items](#) (ItemContainer &items)
- void [list\\_items](#) (long cat\_id, [item\\_choice](#) choice)
- void [execute](#) (ItemContainer &items, [ItemFuncor](#) &funcor)
- void [tick\\_items](#) (Category &c)
- void [tick\\_items](#) (ItemContainer &items)
- void [tick\\_items](#) (long cat\_id, [item\\_choice](#) choice)  
*Checks/ticks all items belonging to a category.*
- void [list\\_categories](#) ()  
*Lists details of all categories to STDOUT.*

- void [new\\_category](#) (const string name)
- void [delete\\_category](#) (long id)
- void [delete\\_item](#) (long id)
- void [remove\\_item\\_from\\_category](#) (long id, long cat\_id)
- void [set\\_item\\_flag](#) (long id)
- void [unset\\_item\\_flag](#) (long id)
- void [set\\_category\\_flag](#) (long id)
- void [unset\\_category\\_flag](#) (long id)
- void [set\\_all\\_flags](#) ()
- void [unset\\_all\\_flags](#) ()

## Protected Attributes

- [KitListDao](#) \* [m\\_dao](#)

*Reference to an implementation DAO class.*

## 7.10.1 Detailed Description

Main application class.

Primarily performs application startup and command line parsing. Allows the application to be run either from the command line or as an interactive GUI application.

Definition at line 78 of file main.cpp.

## 7.10.2 Constructor & Destructor Documentation

### 7.10.2.1 KitList()

```
KitList::KitList (
    const string dbname,
    const string user,
    const string port,
    bool verbose = false )
```

Constructor specifying Postgresql database connection parameters.

Note that the PostgreSQL libraries will use default values for these parameters if they are not specified, including examining environment variables. See the PostgreSQL documentation for full details.

#### Parameters

<i>dbname</i>	The name of the database.
<i>user</i>	The database user name to connect with.
<i>port</i>	The database port to connect to.
<i>verbose</i>	Optional parameter indicating whether to include verbose output to STDOUT. Defaults to false.

Definition at line 162 of file main.cpp.

### 7.10.2.2 ~KitList()

```
KitList::~KitList ( )
```

Definition at line 171 of file main.cpp.

## 7.10.3 Member Function Documentation

### 7.10.3.1 add\_item() [1/2]

```
void KitList::add_item (
    const string name )
```

Referenced by main().

### 7.10.3.2 add\_item() [2/2]

```
void KitList::add_item (
    const string name,
    long cat_id )
```

Creates a new item with the specified name and optionally associates it with a category.

#### Parameters

<i>name</i>	The name of the item to create.
<i>cat↔ _id</i>	The ID of the existing category to associate the item with. If the ID is less than zero, the item will not be associated with a category. Otherwise, the category must already exist.

Definition at line 184 of file main.cpp.

### 7.10.3.3 append\_items\_to\_category()

```
void KitList::append_items_to_category (
    long from_cat_id,
    long to_cat_id,
    item_choice choice )
```

Copies items from one category to another.

Optionally, only checked or unchecked items are copied.



## Parameters

<i>from_cat↔ _id</i>	The ID of the source category.
<i>to_cat_id</i>	The ID of the target category.
<i>choice</i>	One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Definition at line 203 of file main.cpp.

Referenced by main().

## 7.10.3.4 associate\_item\_with\_category()

```
void KitList::associate_item_with_category (
    long id,
    long cat_id )
```

Associates an existing item with an existing category.

## Parameters

<i>id</i>	The <a href="#">Item</a> ID
<i>cat↔ _id</i>	The <a href="#">Category</a> ID

Definition at line 214 of file main.cpp.

Referenced by main().

## 7.10.3.5 delete\_category()

```
void KitList::delete_category (
    long id )
```

Deletes the [Category](#) with the specified id.

## Parameters

<i>id</i>	The <a href="#">Category</a> id.
-----------	----------------------------------

Definition at line 435 of file main.cpp.

Referenced by main().

### 7.10.3.6 delete\_item()

```
void KitList::delete_item (
    long id )
```

Deletes the item with the passed id.

Definition at line 413 of file main.cpp.

Referenced by main().

### 7.10.3.7 execute()

```
void KitList::execute (
    ItemContainer & items,
    ItemFuncutor & functor )
```

Executes the passed [ItemFuncutor](#) for each item in the ItemContainer.

See also

[ItemFuncutor](#).

Definition at line 356 of file main.cpp.

### 7.10.3.8 get\_dao()

```
KitListDao* KitList::get_dao ( ) [inline]
```

Definition at line 84 of file main.cpp.

Referenced by main().

### 7.10.3.9 list\_categories()

```
void KitList::list_categories ( )
```

Lists details of all categories to STDOUT.

Renders with HTML if the html\_flag has been set using the '-html' command line option.

Definition at line 447 of file main.cpp.

References [html\\_flag](#).

Referenced by main().

### 7.10.3.10 list\_item()

```
void KitList::list_item (
    Item & item )
```

Outputs details of the passed item to STDOUT.

Details are wrapped in HTML format if the `html_flag` has been set. The `html_flag` is set by the command line option, `'-html'`.

Definition at line 292 of file `main.cpp`.

References `Item::get_description()`, `Item::get_id()`, and `html_flag`.

### 7.10.3.11 list\_items() [1/3]

```
void KitList::list_items (
    Category & c )
```

Lists details of all items in the passed `Category` to STDOUT.

Definition at line 308 of file `main.cpp`.

References `Category::foreach_item()`, `Category::has_items()`, `Category::item_count()`, and `on_list_item()`.

Referenced by `main()`.

### 7.10.3.12 list\_items() [2/3]

```
void KitList::list_items (
    ItemContainer & items )
```

Lists details of all items in the passed `ItemContainer` to STDOUT.

Definition at line 318 of file `main.cpp`.

### 7.10.3.13 list\_items() [3/3]

```
void KitList::list_items (
    long cat_id,
    item_choice choice = ALL_ITEMS )
```

Lists details of items to STDOUT.

## Parameters

<i>cat_id</i>	The ID Of the <a href="#">Category</a> to list. If the value is less than zero, all items are listed.
<i>choice</i>	Optional. One of ALL_ITEMS (default), CHECKED_ITEMS or UNCHECKED_ITEMS.

Definition at line 336 of file main.cpp.

## 7.10.3.14 list\_items\_end()

```
void KitList::list_items_end (
    bool empty_list,
    size_t count )
```

Called after writing a list of items to STDOUT.

Fundamentally outputs a footer, in either HTML or text format, depending on the value of `html_flag`. The `html_flag` is set by the command line option, '-html'.

## Parameters

<i>empty_list</i>	Set to true if the list is empty. If so, a table footer will not be output.
-------------------	---

Definition at line 269 of file main.cpp.

References `html_flag`.

## 7.10.3.15 list\_items\_start()

```
void KitList::list_items_start (
    bool empty_list )
```

Called before writing a list of items to STDOUT.

Fundamentally outputs headings, in either HTML or text format, depending on the value of `html_flag`. The `html_flag` is set by the command line option, '-html'.

## Parameters

<i>empty_list</i>	Set to true if the list is empty. If so, table headings will not be output.
-------------------	---

Definition at line 238 of file main.cpp.

References `html_flag`.

### 7.10.3.16 new\_category()

```
void KitList::new_category (
    const string name )
```

Creates a new category with the passed name.

Definition at line 496 of file main.cpp.

Referenced by main().

### 7.10.3.17 on\_list\_item()

```
bool KitList::on_list_item (
    Item & item )
```

A callback function to output details of the passed [Item](#) to STDOUT.

Definition at line 223 of file main.cpp.

Referenced by list\_items().

### 7.10.3.18 remove\_item\_from\_category()

```
void KitList::remove_item_from_category (
    long id,
    long cat_id )
```

Un-associates the specified item from the specified category.

#### Parameters

<i>id</i>	The <a href="#">Item</a> id.
<i>cat↔ _id</i>	The <a href="#">Category</a> id.

Definition at line 425 of file main.cpp.

Referenced by main().

### 7.10.3.19 set\_all\_flags()

```
void KitList::set_all_flags ( )
```

Checks/ticks all items.

Definition at line 541 of file main.cpp.

Referenced by main().

#### 7.10.3.20 set\_category\_flag()

```
void KitList::set_category_flag (  
    long id )
```

Checks/ticks all items in the specified [Category](#).

Definition at line 523 of file main.cpp.

Referenced by main().

#### 7.10.3.21 set\_item\_flag()

```
void KitList::set_item_flag (  
    long id )
```

Checks/ticks the specified item.

Definition at line 505 of file main.cpp.

Referenced by main().

#### 7.10.3.22 tick\_items() [1/3]

```
void KitList::tick_items (  
    Category & c )
```

Checks/ticks all items in the passed [Category](#).

Definition at line 378 of file main.cpp.

References [Category::execute\(\)](#).

Referenced by main().

### 7.10.3.23 tick\_items() [2/3]

```
void KitList::tick_items (
    ItemContainer & items )
```

Checks/ticks all items in the passed ItemContainer.

Definition at line 367 of file main.cpp.

### 7.10.3.24 tick\_items() [3/3]

```
void KitList::tick_items (
    long cat_id,
    item_choice choice = ALL_ITEMS )
```

Checks/ticks all items belonging to a category.

Acts on all items if the cat\_id is less than zero. Optionally operates on checked or unchecked items, depending on the value of choice.

#### Parameters

<i>choice</i>	One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.
---------------	---

Definition at line 394 of file main.cpp.

### 7.10.3.25 unset\_all\_flags()

```
void KitList::unset_all_flags ( )
```

Unchecks/unticks all items.

Definition at line 550 of file main.cpp.

Referenced by main().

### 7.10.3.26 unset\_category\_flag()

```
void KitList::unset_category_flag (
    long id )
```

Unchecks/unticks all items in the specified [Category](#).

Definition at line 532 of file main.cpp.

Referenced by main().

### 7.10.3.27 unset\_item\_flag()

```
void KitList::unset_item_flag (
    long id )
```

Unchecks/unticks the specified item.

Definition at line 514 of file main.cpp.

Referenced by main().

## 7.10.4 Member Data Documentation

### 7.10.4.1 m\_dao

```
KitListDao* KitList::m_dao [protected]
```

Reference to an implementation DAO class.

Definition at line 80 of file main.cpp.

The documentation for this class was generated from the following file:

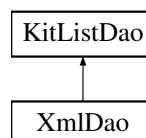
- [/home/frank/Projects/kitlist/src/main.cpp](#)

## 7.11 KitListDao Class Reference

Defines the methods that an implementation of this class must implement.

```
#include <kitlistdao.hpp>
```

Inheritance diagram for KitListDao:





## Public Member Functions

- [KitListDao](#) (int verbose=0)  
*Constructor which will use default database connection parameters.*
- virtual [~KitListDao](#) ()
- virtual [KitModel](#) \* [get\\_model](#) ()=0  
*Loads the data model.*
- virtual void [save\\_model](#) ([KitModel](#) \*model)=0  
*Saves the current data model.*
- void [set\\_verbose](#) (int [verbose\\_flag](#))
- int [is\\_verbose](#) ()
- virtual [Category](#) \* [get\\_category](#) (long cat\_id, [item\\_choice](#) choice=[ALL\\_ITEMS](#))=0  
*Loads a category.*
- virtual [ItemContainer](#) \* [get\\_all\\_items](#) ([item\\_choice](#) choice=[ALL\\_ITEMS](#))=0  
*Returns a list of all items.*
- virtual long [add\\_item](#) (const std::string name)=0
- virtual long [add\\_item](#) (const std::string name, long cat\_id)=0
- virtual void [append\\_items\\_to\\_category](#) (long to\_cat\_id, long from\_cat\_id=-1, [item\\_choice](#) choice=[ALL\\_ITEMS](#))=0  
*Copies items from one category to another.*
- virtual void [associate\\_item\\_with\\_category](#) (long id, long cat\_id)=0  
*Associates an existing item with an existing category.*
- virtual [CategoryContainer](#) [get\\_categories](#) ()=0
- virtual long [new\\_category](#) (const std::string name)=0  
*Creates a new category.*
- virtual void [delete\\_item](#) (long id)=0
- virtual void [update\\_item\\_checked\\_state](#) ([ItemContainer](#) &items)=0  
*Persists the state of the 'checked' flag of each item.*
- virtual void [remove\\_item\\_from\\_category](#) (long id, long cat\_id)=0
- virtual long [get\\_next\\_item\\_id](#) ()=0
- virtual long [get\\_next\\_category\\_id](#) ()=0
- virtual void [delete\\_category](#) (long id)=0
- virtual void [set\\_item\\_flag](#) (long id)=0
- virtual void [unset\\_item\\_flag](#) (long id)=0
- virtual void [set\\_category\\_flag](#) (long id)=0
- virtual void [unset\\_category\\_flag](#) (long id)=0
- virtual void [set\\_all\\_flags](#) ()=0
- virtual void [unset\\_all\\_flags](#) ()=0
- virtual bool [require\\_filename](#) ()  
*Indicates whether the implementation of the data model requires a filename.*

## Protected Attributes

- int [m\\_verbose\\_flag](#)

### 7.11.1 Detailed Description

Defines the methods that an implementation of this class must implement.

Definition at line 46 of file kitlistdao.hpp.

## 7.11.2 Constructor & Destructor Documentation

### 7.11.2.1 KitListDao()

```
KitListDao::KitListDao (
    int verbose = 0 ) [inline]
```

Constructor which will use default database connection parameters.

#### Parameters

<i>verbose</i>	Optional parameter indicating whether to include verbose output to STDOUT. Defaults to false.
----------------	---

Definition at line 59 of file kitlistdao.hpp.

### 7.11.2.2 ~KitListDao()

```
virtual KitListDao::~~KitListDao ( ) [inline], [virtual]
```

Definition at line 61 of file kitlistdao.hpp.

References `get_model()`, and `save_model()`.

## 7.11.3 Member Function Documentation

### 7.11.3.1 add\_item() [1/2]

```
virtual long KitListDao::add_item (
    const std::string name ) [pure virtual]
```

Creates a new item.

#### Parameters

<i>name</i>	The name of the new item.
-------------	---------------------------

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

7.11.3.2 `add_item()` [2/2]

```
virtual long KitListDao::add_item (
    const std::string name,
    long cat_id ) [pure virtual]
```

Creates a new item and associates it with a category.

## Parameters

<i>name</i>	The name of the new item.
-------------	---------------------------

Implemented in [XmlDao](#).

7.11.3.3 `append_items_to_category()`

```
virtual void KitListDao::append_items_to_category (
    long to_cat_id,
    long from_cat_id = -1,
    item_choice choice = ALL_ITEMS ) [pure virtual]
```

Copies items from one category to another.

Optionally, only checked or unchecked items are copied.

## Parameters

<i>from_cat_id</i>	The ID of the source category.
<i>to_cat_id</i>	The ID of the target category.
<i>choice</i>	One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

7.11.3.4 `associate_item_with_category()`

```
virtual void KitListDao::associate_item_with_category (
    long id,
    long cat_id ) [pure virtual]
```

Associates an existing item with an existing category.

## Parameters

<i>id</i>	The <a href="#">Item</a> ID
<i>cat_id</i>	The <a href="#">Category</a> ID

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

#### 7.11.3.5 `delete_category()`

```
virtual void KitListDao::delete_category (
    long id ) [pure virtual]
```

Deletes a category.

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

#### 7.11.3.6 `delete_item()`

```
virtual void KitListDao::delete_item (
    long id ) [pure virtual]
```

Deletes an item by it's ID.

##### Parameters

<i>id</i>	the ID of the item to delete.
-----------	-------------------------------

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

#### 7.11.3.7 `get_all_items()`

```
virtual ItemContainer\* KitListDao::get_all_items (
    item\_choice choice = ALL\_ITEMS ) [pure virtual]
```

Returns a list of all items.

##### Parameters

<i>choice</i>	Which items to load. One of <code>ALL_ITEMS</code> , <code>CHECKED_ITEMS</code> or <code>UNCHECKED_ITEMS</code> .
---------------	---

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

### 7.11.3.8 get\_categories()

```
virtual CategoryContainer KitListDao::get_categories ( ) [pure virtual]
```

Returns a list of all categories.

Implemented in [XmlDao](#).

Referenced by [is\\_verbose\(\)](#).

### 7.11.3.9 get\_category()

```
virtual Category\* KitListDao::get_category (
    long cat_id,
    item\_choice choice = ALL\_ITEMS ) [pure virtual]
```

Loads a category.

#### Parameters

<i>choice</i>	Which items to load for the category. One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.
---------------	---

Implemented in [XmlDao](#).

Referenced by [is\\_verbose\(\)](#).

### 7.11.3.10 get\_model()

```
virtual KitModel\* KitListDao::get_model ( ) [pure virtual]
```

Loads the data model.

The data model holds a rich graph of objects, representing the entire list of categories and items.

#### See also

[KitModel](#)

Implemented in [XmlDao](#).

Referenced by [Service::create\\_default\\_model\(\)](#), [Service::load\\_model\(\)](#), and [~KitListDao\(\)](#).

### 7.11.3.11 `get_next_category_id()`

```
virtual long KitListDao::get_next_category_id ( ) [pure virtual]
```

Returns the next unused unique id for categories.

Implemented in [XmlDao](#).

Referenced by `Service::get_next_category_id()`, and `is_verbose()`.

### 7.11.3.12 `get_next_item_id()`

```
virtual long KitListDao::get_next_item_id ( ) [pure virtual]
```

Returns the next unused unique id for items.

Implemented in [XmlDao](#).

Referenced by `Service::get_next_item_id()`, and `is_verbose()`.

### 7.11.3.13 `is_verbose()`

```
int KitListDao::is_verbose ( ) [inline]
```

Indicates whether verbose output should be written to STDOUT.

Definition at line 92 of file `kitlistdao.hpp`.

References `add_item()`, `ALL_ITEMS`, `append_items_to_category()`, `associate_item_with_category()`, `delete_category()`, `delete_item()`, `get_all_items()`, `get_categories()`, `get_category()`, `get_next_category_id()`, `get_next_item_id()`, `m_verbose_flag`, `new_category()`, `remove_item_from_category()`, `set_all_flags()`, `set_category_flag()`, `set_item_flag()`, `unset_all_flags()`, `unset_category_flag()`, `unset_item_flag()`, and `update_item_checked_state()`.

### 7.11.3.14 `new_category()`

```
virtual long KitListDao::new_category (
    const std::string name ) [pure virtual]
```

Creates a new category.

#### Parameters

<i>name</i>	the name of the new category.
-------------	-------------------------------

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

#### 7.11.3.15 `remove_item_from_category()`

```
virtual void KitListDao::remove_item_from_category (
    long id,
    long cat_id ) [pure virtual]
```

Un-associates the specified item from the specified category.

##### Parameters

<i>id</i>	The <a href="#">Item</a> id.
<i>cat↔ _id</i>	The <a href="#">Category</a> id.

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

#### 7.11.3.16 `require_filename()`

```
virtual bool KitListDao::require_filename ( ) [inline], [virtual]
```

Indicates whether the implementation of the data model requires a filename.

Some persistence models may require a filename to be chosen, others (e.g. database) may be defined through another mechanism. If a filename has not been set, then fire up save-as instead.

##### Returns

Always returns false.

Reimplemented in [XmlDao](#).

Definition at line 225 of file `kitlistdao.hpp`.

Referenced by `Service::require_filename()`.

### 7.11.3.17 save\_model()

```
virtual void KitListDao::save_model (
    KitModel * model ) [pure virtual]
```

Saves the current data model.

Note: The data model will only be saved if it is dirty. Further, only items that are individually flagged as dirty will be saved.

See also

[GuiState](#)

Implemented in [XmlDao](#).

Referenced by [Service::save\(\)](#), and [~KitListDao\(\)](#).

### 7.11.3.18 set\_all\_flags()

```
virtual void KitListDao::set_all_flags ( ) [pure virtual]
```

Checks/ticks all items.

Implemented in [XmlDao](#).

Referenced by [is\\_verbose\(\)](#).

### 7.11.3.19 set\_category\_flag()

```
virtual void KitListDao::set_category_flag (
    long id ) [pure virtual]
```

Checks/ticks all items in the specified [Category](#).

Implemented in [XmlDao](#).

Referenced by [is\\_verbose\(\)](#).

### 7.11.3.20 set\_item\_flag()

```
virtual void KitListDao::set_item_flag (
    long id ) [pure virtual]
```

Sets the checked flag for an item.



## Parameters

<i>id</i>	The id of the item to change.
-----------	-------------------------------

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

### 7.11.3.21 `set_verbose()`

```
void KitListDao::set_verbose (
    int verbose_flag ) [inline]
```

Indicates whether verbose output should be written to STDOUT.

Definition at line 85 of file `kitlistdao.hpp`.

References `verbose_flag`.

### 7.11.3.22 `unset_all_flags()`

```
virtual void KitListDao::unset_all_flags ( ) [pure virtual]
```

Unchecks/unticks all items.

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

### 7.11.3.23 `unset_category_flag()`

```
virtual void KitListDao::unset_category_flag (
    long id ) [pure virtual]
```

Unchecks/unticks all items in the specified [Category](#).

Implemented in [XmlDao](#).

Referenced by `is_verbose()`.

#### 7.11.3.24 unset\_item\_flag()

```
virtual void KitListDao::unset_item_flag (
    long id ) [pure virtual]
```

Clears the checked flag for an item.

Implemented in [XmlDao](#).

Referenced by [is\\_verbose\(\)](#).

#### 7.11.3.25 update\_item\_checked\_state()

```
virtual void KitListDao::update_item_checked_state (
    ItemContainer & items ) [pure virtual]
```

Persists the state of the 'checked' flag of each item.

Implemented in [XmlDao](#).

Referenced by [is\\_verbose\(\)](#).

### 7.11.4 Member Data Documentation

#### 7.11.4.1 m\_verbose\_flag

```
int KitListDao::m_verbose_flag [protected]
```

Optional parameter indicating whether to include verbose output to STDOUT. Defaults to false.

Definition at line 50 of file [kitlistdao.hpp](#).

Referenced by [is\\_verbose\(\)](#).

The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/kitlistdao.hpp](#)

## 7.12 KitListGui Class Reference

Encapsulates the methods for the application's GUI front end.

```
#include <kitlistgui.hpp>
```

## Public Member Functions

- [KitListGui](#) (int argc, char \*\*argv, [Service](#) &service)
- [~KitListGui](#) ()
- virtual void [open\\_file](#) (const Glib::ustring &filename)
 

*Opens an existing XML document.*
- virtual void [raise](#) ()
 

*Make this application topmost.*
- virtual void [safe\\_open\\_file](#) (const Glib::ustring &filename)
 

*Opens an existing XML document, checking for unsaved changes.*
- void [run](#) ()
 

*Starts the GUI application running.*

## Protected Member Functions

- virtual void [init](#) ()
- virtual gint [get\\_max\\_recent\\_files](#) ()
- virtual [ModellItemContainer](#) \* [get\\_selected\\_items](#) ()
 

*Returns a list of items selected in the item list.*
- virtual void [add\\_items](#) (const [ModellItemContainer](#) &items)
 

*Associates the passed list of items with the currently selected category.*
- virtual void [set\\_page\\_title](#) (const Glib::ustring page\_title)
- virtual void [close\\_preferences\\_window](#) ()
- virtual void [cancel\\_preferences\\_window](#) ()
- virtual void [close\\_add\\_item\\_window](#) ()
- virtual void [cancel\\_add\\_item\\_window](#) ()
- virtual void [close\\_add\\_category\\_window](#) ()
 

*Called when the add/rename category dialog is closed using the 'OK' button.*
- virtual void [cancel\\_add\\_category\\_window](#) ()
 

*Called when the add/rename category dialog is closed using the 'Cancel' button.*
- virtual long [get\\_selected\\_category](#) ()
 

*Returns the ID of the currently selected [Category](#).*
- virtual void [init\\_add\\_item\\_window](#) ()
- virtual void [delete\\_selected\\_items](#) ()
 

*Deletes the currently selected items.*
- virtual [ModellItemContainer](#) \* [copy\\_selected\\_items\\_to\\_clipboard](#) ()
- virtual bool [confirm\\_lose\\_changes](#) (const Glib::ustring &message)
 

*Shows a confirmation message.*
- virtual void [update\\_recent\\_files\\_menu](#) ()
 

*Updates the recent files sub menu.*
- virtual void [update\\_recent\\_files](#) (const Glib::ustring &filename)
- virtual bool [on\\_delete\\_event](#) (GdkEventAny \*event)
 

*Called when the application is closed by the user.*
- virtual void [on\\_menu\\_quit](#) ()
 

*Called when the user chooses to quit the application.*
- virtual void [on\\_menu\\_file\\_new](#) ()
 

*Creates a new empty model.*
- virtual void [on\\_menu\\_file\\_open](#) ()
 

*Allows the user to open an existing XML document.*
- virtual void [on\\_menu\\_save](#) ()
 

*Saves the current application state.*

- virtual void [on\\_menu\\_save\\_as](#) ()  
*Saves the current application state in a new document.*
- void [on\\_printoperation\\_done](#) (Gtk::PrintOperationResult result, const Glib::RefPtr< Gtk::PrintOperation > &op)  
*Called when the printer operation is done.*
- void [on\\_printoperation\\_status\\_changed](#) (const Glib::RefPtr< Gtk::PrintOperation > &op)  
*Called when the print status changes.*
- virtual void [on\\_menu\\_print](#) ()  
*Prints the kit list.*
- virtual void [on\\_menu\\_export\\_to\\_pdf](#) ()
- virtual void [on\\_menu\\_recent\\_file](#) (const Glib::ustring &filename)  
*displays the most recent files menu*
- virtual void [on\\_menu\\_preferences](#) ()
- virtual void [on\\_menu\\_add](#) ()
- virtual void [on\\_menu\\_delete](#) ()  
*Called when the user chooses to delete items.*
- virtual void [on\\_menu\\_cut](#) ()
- virtual void [on\\_menu\\_copy](#) ()  
*Called when the use chooses the 'copy' menu option.*
- virtual void [on\\_menu\\_paste](#) ()  
*Called when the user chooses the 'paste' menu option.*
- virtual void [on\\_menu\\_show\\_all](#) ()  
*Causes all items to be displayed.*
- virtual void [on\\_menu\\_show\\_checked](#) ()  
*Causes only checked items to be displayed.*
- virtual void [on\\_menu\\_show\\_unchecked](#) ()  
*Causes only unchecked items to be displayed.*
- virtual void [on\\_menu\\_select\\_all](#) ()  
*Called when the user chooses the 'select all' menu option.*
- virtual void [on\\_menu\\_check\\_selected](#) ()  
*Marks all selected items as checked.*
- virtual void [on\\_menu\\_uncheck\\_selected](#) ()  
*Marks all selected items as unchecked.*
- virtual void [on\\_menu\\_create\\_category](#) ()  
*Called when the user chooses to create a new category.*
- virtual void [on\\_menu\\_delete\\_category](#) ()  
*Called when the user chooses the delete category menu option.*
- virtual void [on\\_menu\\_rename\\_category](#) ()  
*Called when the user chooses to rename a category.*
- virtual void [on\\_menu\\_help\\_about](#) ()
- virtual void [on\\_clipboard\\_get](#) (Gtk::SelectionData &selection\_date, guint)
- virtual void [on\\_clipboard\\_clear](#) ()  
*This method gets called after a second 'copy' operation.*
- virtual void [on\\_clipboard\\_received](#) (const Gtk::SelectionData &selection\_data)
- virtual void [on\\_category\\_change](#) ()
- virtual void [on\\_cell\\_edit](#) (const Glib::ustring s)  
*Callback method called when a cell has been edited in the item list.*
- virtual bool [choose\\_filename](#) (Glib::ustring &filename)  
*Displays a file chooser dialog for a user to choose a filename.*
- virtual bool [choose\\_pdf\\_filename](#) (Glib::ustring &filename)  
*Displays a file chooser dialog for a user to choose a filename for export to PDF.*

- virtual void [update\\_paste\\_status](#) ()  
*Enables or disables the paste menu option.*
- virtual void [paste\\_status\\_received](#) (const Glib::StringArrayHandle &targets\_array)
- virtual void [paste\\_from\\_xml](#) (const Glib::ustring &document)
- virtual void [refresh\\_item\\_list](#) ()
- virtual void [refresh\\_category\\_list](#) (long cat\_id=-2)  
*Refreshes the category combo box list.*
- virtual void [selected\\_row\\_callback](#) (const Gtk::TreeModel::iterator &iter)
- virtual void [set\\_selected](#) (bool checked)
- virtual void [toggle\\_selected](#) ()
- void [on\\_row\\_changed](#) (const Gtk::TreeModel::Path path, const Gtk::TreeModel::iterator iter)  
*Callback method called when an item has been modified in the item list.*
- virtual void [update\\_item\\_count](#) (size\_t n)

## Protected Attributes

- Glib::ustring [m\\_filename](#)  
*The filename currently associated with the loaded model.*
- Glib::ustring [m\\_page\\_title](#)  
*The page title to be used when printing the item list.*
- Glib::ustring [m\\_clipboard\\_items](#)  
*Holder for items pasted to the clipboard.*
- bool [m\\_ignore\\_list\\_events](#)  
*Temporarily ignore events on the item list.*
- Gtk::Main [m\\_kit](#)  
*The main application.*
- Gtk::Window \* [m\\_window](#)  
*The main application window.*
- Gtk::Window \* [m\\_window\\_preferences](#)  
*The 'Preferences' dialog.*
- Gtk::Entry \* [m\\_entry\\_page\\_title](#)  
*the text entry field for the page title*
- Gtk::Window \* [m\\_window\\_add\\_item](#)  
*The 'Add Item' dialog.*
- Gtk::Window \* [m\\_window\\_add\\_category](#)  
*The 'Add Category' dialog.*
- Gtk::Entry \* [m\\_entry\\_add\\_item](#)  
*The text entry field of the 'Add Item' dialog.*
- Gtk::Entry \* [m\\_entry\\_add\\_category](#)  
*The text entry field of the 'Add Category' dialog.*
- Gtk::ImageMenuItem \* [m\\_file\\_save\\_menu\\_item](#)  
*The file save menu item.*
- Gtk::ToolButton \* [m\\_file\\_save\\_tool\\_button](#)  
*The file save toolbar button.*
- Gtk::MenuItem \* [m\\_recent\\_files\\_menu\\_item](#)  
*The recent files menu item.*
- Gtk::ImageMenuItem \* [m\\_paste\\_menu\\_item](#)  
*The menu paste button.*
- Gtk::ToolButton \* [m\\_paste\\_tool\\_button](#)  
*The toolbar paste button.*

- `Gtk::CheckBox * m_checkbutton_add_item`  
*The check button field of the 'Add Item' dialog.*
- `Gtk::ComboBox * m_category_combo`  
*The combo box holding a list of categories.*
- `Glib::RefPtr< Gtk::ListStore > m_ref_category_list_store`  
*The model backing the category combo box.*
- `ModelCategoryColumns m_category_cols`  
*The definition of the category combo box columns.*
- `Gtk::TreeView * m_item_tree_view`  
*The item list view definition.*
- `ModelItemColumns m_item_cols`  
*The definition of the item list's columns.*
- `Service & m_service`  
*The business/service object.*
- `Glib::RefPtr< Gtk::ListStore > m_ref_item_tree_model`  
*The model backing the item list.*
- `Gtk::Statusbar * m_status_bar`  
*The application status bar.*
- `Glib::RefPtr< Gtk::PageSetup > m_ref_page_setup`  
*Printer page setup settings.*
- `Glib::RefPtr< Gtk::PrintSettings > m_ref_printer_settings`  
*Printer settings.*
- `enum gui_state m_state`  
*Indicates whether a category is being created or renamed.*
- `long m_current_cat_id`  
*temporary reference to a category id, usually being renamed*

## Private Attributes

- `YamlConfig m_yaml_config`

### 7.12.1 Detailed Description

Encapsulates the methods for the application's GUI front end.

This is the GTK+ implementation.

Definition at line 99 of file kitlistgui.hpp.

### 7.12.2 Constructor & Destructor Documentation

#### 7.12.2.1 KitListGui()

```
KitListGui::KitListGui (
    int argc,
    char ** argv,
    Service & service )
```

Constructor to create the GUI application.

## Parameters

<i>argc</i>	command line argument count passed to Gtk::Main
<i>argv</i>	command line arguments passed to Gtk::Main
<i>service</i>	a reference to the <a href="#">Service</a> object, providing all business/service methods.

Definition at line 193 of file kitlistgui.cpp.

References [init\(\)](#), [m\\_ref\\_page\\_setup](#), and [m\\_ref\\_printer\\_settings](#).

## 7.12.2.2 ~KitListGui()

```
KitListGui::~KitListGui ( )
```

Definition at line 221 of file kitlistgui.cpp.

## 7.12.3 Member Function Documentation

## 7.12.3.1 add\_items()

```
void KitListGui::add_items (
    const ModelItemContainer & items ) [protected], [virtual]
```

Associates the passed list of items with the currently selected category.

Where items already exist in the [Category](#), then are not duplicated, just silently ignored.

Definition at line 1172 of file kitlistgui.cpp.

References [Service::copy\\_items\(\)](#), [get\\_selected\\_category\(\)](#), [m\\_service](#), and [refresh\\_item\\_list\(\)](#).

Referenced by [paste\\_from\\_xml\(\)](#).

## 7.12.3.2 cancel\_add\_category\_window()

```
void KitListGui::cancel_add_category_window ( ) [protected], [virtual]
```

Called when the add/rename category dialog is closed using the 'Cancel' button.

Definition at line 1022 of file kitlistgui.cpp.

References [m\\_window\\_add\\_category](#).

Referenced by [init\(\)](#).

### 7.12.3.3 `cancel_add_item_window()`

```
void KitListGui::cancel_add_item_window ( ) [protected], [virtual]
```

Called when the user closes the 'Add [Item](#)' dialog using the 'Cancel' button.

Definition at line 1746 of file `kitlistgui.cpp`.

References `m_window_add_item`.

Referenced by `init()`.

### 7.12.3.4 `cancel_preferences_window()`

```
void KitListGui::cancel_preferences_window ( ) [protected], [virtual]
```

Definition at line 1674 of file `kitlistgui.cpp`.

References `m_window_preferences`.

Referenced by `init()`.

### 7.12.3.5 `choose_filename()`

```
bool KitListGui::choose_filename (
    Glib::ustring & filename ) [protected], [virtual]
```

Displays a file chooser dialog for a user to choose a filename.

If the file already exists, the user is asked to confirm whether to overwrite.

#### Parameters

<i>filename</i>	A reference to a string to populate with the chosen filename.
-----------------	---

#### Returns

true if the user selects OK, false otherwise.

Definition at line 1276 of file `kitlistgui.cpp`.

References `anonymous_namespace{kitlistgui.cpp}::DEFAULT_FILENAME_EXTENSION`, `file_exists()`, and `m_↔window`.

Referenced by `on_menu_save_as()`.



## 7.12.3.6 choose\_pdf\_filename()

```
bool KitListGui::choose_pdf_filename (
    Glib::ustring & filename ) [protected], [virtual]
```

Displays a file chooser dialog for a user to choose a filename for export to PDF.

If the file already exists, the user is asked to confirm whether to overwrite.

## Parameters

<i>filename</i>	A reference to a string to populate with the chosen filename.
-----------------	---

## Returns

true if the user selects OK, false otherwise.

Definition at line 1374 of file kitlistgui.cpp.

References `file_exists()`, `m_window`, and `anonymous_namespace{kitlistgui.cpp}::PDF_FILENAME_EXTENSION`.

Referenced by `on_menu_export_to_pdf()`.

## 7.12.3.7 close\_add\_category\_window()

```
void KitListGui::close_add_category_window ( ) [protected], [virtual]
```

Called when the add/rename category dialog is closed using the 'OK' button.

The same dialog is used for both creating a new category and renaming an existing one. `m_state` is used to indicate which operation is relevant (create or rename) and `m_current_cat_id` is used to indicate the category being renamed for the rename operation.

Definition at line 997 of file kitlistgui.cpp.

References `ADD_CATEGORY`, `Service::create_category()`, `Service::find_category()`, `Category::get_id()`, `m_current_cat_id`, `m_entry_add_category`, `m_service`, `m_state`, `m_window_add_category`, `refresh_category_list()`, `refresh_item_list()`, `Service::set_model_dirty()`, and `Category::set_name()`.

Referenced by `init()`.

## 7.12.3.8 close\_add\_item\_window()

```
void KitListGui::close_add_item_window ( ) [protected], [virtual]
```

Called when the user closes the 'Add Item' dialog using the 'OK' button.

Definition at line 1726 of file kitlistgui.cpp.

References `Service::create_item()`, `get_selected_category()`, `m_checkbutton_add_item`, `m_entry_add_item`, `m_service`, `m_window_add_item`, `refresh_item_list()`, `Item::set_checked()`, and `Item::set_description()`.

Referenced by `init()`.

### 7.12.3.9 close\_preferences\_window()

```
void KitListGui::close_preferences_window ( ) [protected], [virtual]
```

Called when the user closes the 'Preferences' dialog using the 'OK' button.

Definition at line 1668 of file kitlistgui.cpp.

References `m_entry_page_title`, `m_window_preferences`, and `set_page_title()`.

Referenced by `init()`.

### 7.12.3.10 confirm\_lose\_changes()

```
bool KitListGui::confirm_lose_changes (
    const Glib::ustring & message ) [protected], [virtual]
```

Shows a confirmation message.

Definition at line 247 of file kitlistgui.cpp.

References `m_filename`, `m_service`, `m_window`, `Service::save()`, `Service::save_as_xml()`, and `Service::set_model_dirty()`.

Referenced by `on_delete_event()`, `on_menu_file_new()`, `on_menu_file_open()`, `on_menu_quit()`, `on_menu_recent_file()`, and `safe_open_file()`.

### 7.12.3.11 copy\_selected\_items\_to\_clipboard()

```
ModelItemContainer * KitListGui::copy_selected_items_to_clipboard ( ) [protected], [virtual]
```

Copies the currently selected items to the clipboard.

Definition at line 878 of file kitlistgui.cpp.

References `get_selected_items()`, `anonymous_namespace{kitlistgui.cpp}::item_target_custom`, `anonymous_namespace{kitlistgui.cpp}::item_target_text`, `m_clipboard_items`, `on_clipboard_clear()`, `on_clipboard_get()`, `update_paste_status()`, and `anonymous_namespace{kitlistgui.cpp}::XML_ELEMENT_ID`.

Referenced by `on_menu_copy()`, and `on_menu_cut()`.

#### 7.12.3.12 delete\_selected\_items()

```
void KitListGui::delete_selected_items ( ) [protected], [virtual]
```

Deletes the currently selected items.

The items are flagged as deleted. When save is called, they will no longer be persisted, i.e. they will be permanently deleted.

Definition at line 823 of file kitlistgui.cpp.

References `Service::delete_item()`, `ModelItemColumns::m_col_num`, `m_item_cols`, `m_item_tree_view`, `m_ref_↔item_tree_model`, `m_service`, `refresh_item_list()`, and `selected_row_callback()`.

Referenced by `on_menu_delete()`.

#### 7.12.3.13 get\_max\_recent\_files()

```
gint KitListGui::get_max_recent_files ( ) [protected], [virtual]
```

Definition at line 240 of file kitlistgui.cpp.

References `DEFAULT_MAX_RECENT_FILES`.

Referenced by `update_recent_files()`.

#### 7.12.3.14 get\_selected\_category()

```
long KitListGui::get_selected_category ( ) [protected], [virtual]
```

Returns the ID of the currently selected [Category](#).

Returns -1 if no [Category](#) is currently selected, or the 'all items' option is selected.

Definition at line 1709 of file kitlistgui.cpp.

References `m_category_cols`, `m_category_combo`, and `ModelCategoryColumns::m_col_num`.

Referenced by `add_items()`, `close_add_item_window()`, `on_menu_cut()`, `on_menu_delete_category()`, `on_menu_↔_export_to_pdf()`, `on_menu_print()`, `on_menu_rename_category()`, `refresh_category_list()`, and `refresh_item_list()`.

### 7.12.3.15 get\_selected\_items()

```
ModelItemContainer * KitListGui::get_selected_items ( ) [protected], [virtual]
```

Returns a list of items selected in the item list.

Definition at line 777 of file kitlistgui.cpp.

References Service::find\_item(), ModellItemColumns::m\_col\_num, m\_item\_cols, m\_item\_tree\_view, m\_ref\_item\_↵  
\_tree\_model, and m\_service.

Referenced by copy\_selected\_items\_to\_clipboard(), set\_selected(), and toggle\_selected().

### 7.12.3.16 init()

```
void KitListGui::init ( ) [protected], [virtual]
```

Initialises all the GUI components prior to the GUI application being run.

Definition at line 1770 of file kitlistgui.cpp.

References cancel\_add\_category\_window(), cancel\_add\_item\_window(), cancel\_preferences\_window(), close\_↵  
add\_category\_window(), close\_add\_item\_window(), close\_preferences\_window(), anonymous\_namespace{kitlistgui.↵  
cpp}::DEFAULT\_FILENAME, DEFAULT\_PAGE\_TITLE, file\_exists(), anonymous\_namespace{kitlistgui.cpp}↵  
::GCONF\_KEY, anonymous\_namespace{kitlistgui.cpp}::GCONF\_KEY\_CURRENT\_FILENAME, anonymous\_↵  
namespace{kitlistgui.cpp}::GCONF\_KEY\_PAGE\_TITLE, YamlConfig::get\_current\_filename(), YamlConfig::get\_↵  
\_debug\_log\_filename(), get\_glade\_ref\_ptr(), Service::get\_items(), YamlConfig::get\_page\_title(), anonymous\_↵  
namespace{kitlistgui.cpp}::GLADE\_APP\_FILE, m\_category\_cols, m\_category\_combo, m\_checkbutton\_add\_item,↵  
ModellItemColumns::m\_col\_checked, ModellItemColumns::m\_col\_num, ModellItemColumns::m\_col\_text, m\_↵  
entry\_add\_category, m\_entry\_add\_item, m\_entry\_page\_title, m\_file\_save\_menu\_item, m\_file\_save\_tool\_button,↵  
m\_filename, m\_ignore\_list\_events, m\_item\_cols, m\_item\_tree\_view, m\_page\_title, m\_paste\_menu\_item, m\_↵  
paste\_tool\_button, m\_recent\_files\_menu\_item, m\_ref\_category\_list store, m\_ref\_item\_tree\_model, m\_service,↵  
m\_status\_bar, m\_window, m\_window\_add\_category, m\_window\_add\_item, m\_window\_preferences, m\_yaml\_↵  
config, on\_category\_change(), on\_delete\_event(), on\_menu\_add(), on\_menu\_check\_selected(), on\_menu\_copy(),↵  
on\_menu\_create\_category(), on\_menu\_cut(), on\_menu\_delete(), on\_menu\_delete\_category(), on\_menu\_export\_↵  
\_to\_pdf(), on\_menu\_file\_new(), on\_menu\_file\_open(), on\_menu\_help\_about(), on\_menu\_paste(), on\_menu\_↵  
preferences(), on\_menu\_print(), on\_menu\_quit(), on\_menu\_rename\_category(), on\_menu\_save(), on\_menu\_↵  
save\_as(), on\_menu\_select\_all(), on\_menu\_show\_all(), on\_menu\_show\_checked(), on\_menu\_show\_unchecked(),↵  
on\_menu\_uncheck\_selected(), on\_row\_changed(), Service::open\_as\_xml(), refresh\_category\_list(), refresh\_↵  
item\_list(), Service::require\_filename(), Service::set\_model\_dirty(), toggle\_selected(), update\_item\_count(), and↵  
update\_recent\_files\_menu().

Referenced by KitListGui().

### 7.12.3.17 init\_add\_item\_window()

```
void KitListGui::init_add_item_window ( ) [protected], [virtual]
```

Initialises the 'Add [Item](#)' dialog prior to it being displayed.

Definition at line 1696 of file kitlistgui.cpp.

References m\_entry\_add\_item.

Referenced by on\_menu\_add().

#### 7.12.3.18 on\_category\_change()

```
void KitListGui::on_category_change ( ) [protected], [virtual]
```

Called after the user has changed the currently selected [Category](#).

Definition at line 1754 of file kitlistgui.cpp.

References `m_ignore_list_events`, and `refresh_item_list()`.

Referenced by `init()`.

#### 7.12.3.19 on\_cell\_edit()

```
void KitListGui::on_cell_edit (
    const Glib::ustring s ) [protected], [virtual]
```

Callback method called when a cell has been edited in the item list.

Flags the model as dirty.

Definition at line 1259 of file kitlistgui.cpp.

References `m_ignore_list_events`, `m_service`, and `Service::set_model_dirty()`.

#### 7.12.3.20 on\_clipboard\_clear()

```
void KitListGui::on_clipboard_clear ( ) [protected], [virtual]
```

This method gets called after a second 'copy' operation.

i.e. if we have previously called `Clipboard::set()` and then called it a second time. However, I think this is intended to only clean up data objects that may have been allocated by a previous call to [on\\_clipboard\\_get\(\)](#) where we might have created an expensive object based on a list of IDs or something. This gives us the opportunity to release the big object.

However, in the way we're using the clipboard, we're holding the expensive object (in this case an XML document). We should probably be holding the list of selected ID's and only creating the XML document when the clipboard contents are requested.

As things are, we don't want to clear the XML document... `m_clipboard_items.clear()`;

Definition at line 1161 of file kitlistgui.cpp.

Referenced by `copy_selected_items_to_clipboard()`.

### 7.12.3.21 on\_clipboard\_get()

```
void KitListGui::on_clipboard_get (
    Gtk::SelectionData & selection_data,
    guint ) [protected], [virtual]
```

Called when the current clipboard contents are required.

Definition at line 1130 of file kitlistgui.cpp.

References `anonymous_namespace{kitlistgui.cpp}::item_target_custom`, `anonymous_namespace{kitlistgui.cpp}↔::item_target_text`, and `m_clipboard_items`.

Referenced by `copy_selected_items_to_clipboard()`.

### 7.12.3.22 on\_clipboard\_received()

```
void KitListGui::on_clipboard_received (
    const Gtk::SelectionData & selection_data ) [protected], [virtual]
```

Callback notification method for capturing clipboard contents.

Definition at line 1220 of file kitlistgui.cpp.

References `anonymous_namespace{kitlistgui.cpp}::item_target_custom`, `anonymous_namespace{kitlistgui.cpp}↔::item_target_text`, and `m_clipboard_items`.

Referenced by `on_menu_paste()`.

### 7.12.3.23 on\_delete\_event()

```
bool KitListGui::on_delete_event (
    GdkEventAny * event ) [protected], [virtual]
```

Called when the application is closed by the user.

Definition at line 356 of file kitlistgui.cpp.

References `confirm_lose_changes()`, `Service::is_model_dirty()`, `m_service`, and `Service::set_model_dirty()`.

Referenced by `init()`.

#### 7.12.3.24 on\_menu\_add()

```
void KitListGui::on_menu_add ( ) [protected], [virtual]
```

Called when the users chooses to create a new [Item](#).

Definition at line 1682 of file kitlistgui.cpp.

References [init\\_add\\_item\\_window\(\)](#), [m\\_entry\\_add\\_item](#), and [m\\_window\\_add\\_item](#).

Referenced by [init\(\)](#).

#### 7.12.3.25 on\_menu\_check\_selected()

```
virtual void KitListGui::on_menu_check_selected ( ) [inline], [protected], [virtual]
```

Marks all selected items as checked.

Definition at line 225 of file kitlistgui.hpp.

Referenced by [init\(\)](#).

#### 7.12.3.26 on\_menu\_copy()

```
void KitListGui::on_menu_copy ( ) [protected], [virtual]
```

Called when the use chooses the 'copy' menu option.

Definition at line 937 of file kitlistgui.cpp.

References [copy\\_selected\\_items\\_to\\_clipboard\(\)](#).

Referenced by [init\(\)](#).

#### 7.12.3.27 on\_menu\_create\_category()

```
void KitListGui::on_menu_create_category ( ) [protected], [virtual]
```

Called when the user chooses to create a new category.

Definition at line 971 of file kitlistgui.cpp.

References [ADD\\_CATEGORY](#), [m\\_current\\_cat\\_id](#), [m\\_entry\\_add\\_category](#), [m\\_state](#), and [m\\_window\\_add\\_category](#).

Referenced by [init\(\)](#).

### 7.12.3.28 on\_menu\_cut()

```
void KitListGui::on_menu_cut ( ) [protected], [virtual]
```

Copies the currently selected items to the clipboard as a 'cut' operation.

Definition at line 911 of file kitlistgui.cpp.

References `copy_selected_items_to_clipboard()`, `Service::find_category()`, `get_selected_category()`, `m_service`, `m_status_bar`, `refresh_item_list()`, `ModelCategory::remove_items()`, `anonymous_namespace{kitlistgui.cpp}::SB_↔MSG`, `GuiState::set_dirty()`, and `Service::set_model_dirty()`.

Referenced by `init()`.

### 7.12.3.29 on\_menu\_delete()

```
void KitListGui::on_menu_delete ( ) [protected], [virtual]
```

Called when the user chooses to delete items.

The user is prompted to confirm deletion, prior to the items being flagged as deleted in the model. Once the model is saved, they will be permanently deleted from the persistence store.

Definition at line 855 of file kitlistgui.cpp.

References `delete_selected_items()`, and `m_window`.

Referenced by `init()`.

### 7.12.3.30 on\_menu\_delete\_category()

```
void KitListGui::on_menu_delete_category ( ) [protected], [virtual]
```

Called when the user chooses the delete category menu option.

Displays a confirmation box before deleting the currently selected category.

Definition at line 1033 of file kitlistgui.cpp.

References `Service::delete_category()`, `get_selected_category()`, `m_service`, `m_status_bar`, `m_window`, `refresh_↔category_list()`, `refresh_item_list()`, and `anonymous_namespace{kitlistgui.cpp}::SB_MSG`.

Referenced by `init()`.



### 7.12.3.31 on\_menu\_export\_to\_pdf()

```
void KitListGui::on_menu_export_to_pdf ( ) [protected], [virtual]
```

Allows the user to choose a file to export the kitlist to a PDF file.

Definition at line 692 of file kitlistgui.cpp.

References `choose_pdf_filename()`, `KitPrintOperation::create()`, `Service::get_filtered_items()`, `get_selected_category()`, `m_page_title`, `m_ref_page_setup`, `m_ref_printer_settings`, `m_service`, `m_status_bar`, and anonymous namespace{kitlistgui.cpp}::SB\_SAVE.

Referenced by `init()`.

### 7.12.3.32 on\_menu\_file\_new()

```
void KitListGui::on_menu_file_new ( ) [protected], [virtual]
```

Creates a new empty model.

Definition at line 387 of file kitlistgui.cpp.

References `confirm_lose_changes()`, `Service::create_default_model()`, anonymous namespace{kitlistgui.cpp}::GCONF\_KEY\_CURRENT\_FILENAME, `Service::is_model_dirty()`, `m_filename`, `m_service`, `m_status_bar`, `m_yaml_config`, `refresh_category_list()`, `refresh_item_list()`, `Service::require_filename()`, anonymous namespace{kitlistgui.cpp}::SB\_SAVE, and `YamlConfig::set_current_filename()`.

Referenced by `init()`.

### 7.12.3.33 on\_menu\_file\_open()

```
void KitListGui::on_menu_file_open ( ) [protected], [virtual]
```

Allows the user to open an existing XML document.

If there are unsaved changes, the user is first prompted to discard them or cancel the operation.

Definition at line 432 of file kitlistgui.cpp.

References `confirm_lose_changes()`, anonymous namespace{kitlistgui.cpp}::DEFAULT\_FILENAME\_EXTENSION, `Service::is_model_dirty()`, `m_service`, `m_status_bar`, `m_window`, `open_file()`, and anonymous namespace{kitlistgui.cpp}::SB\_SAVE.

Referenced by `init()`.

#### 7.12.3.34 on\_menu\_help\_about()

```
void KitListGui::on_menu_help_about ( ) [protected], [virtual]
```

Shows the help about dialog

Definition at line 1093 of file kitlistgui.cpp.

Referenced by init().

#### 7.12.3.35 on\_menu\_paste()

```
void KitListGui::on_menu_paste ( ) [protected], [virtual]
```

Called when the user chooses the 'paste' menu option.

Definition at line 946 of file kitlistgui.cpp.

References anonymous\_namespace{kitlistgui.cpp}::item\_target\_text, m\_clipboard\_items, on\_clipboard\_received(), paste\_from\_xml(), and update\_paste\_status().

Referenced by init().

#### 7.12.3.36 on\_menu\_preferences()

```
void KitListGui::on_menu_preferences ( ) [protected], [virtual]
```

Called when the user chooses the preferences option from the menu.

Definition at line 1633 of file kitlistgui.cpp.

References m\_entry\_page\_title, m\_page\_title, and m\_window\_preferences.

Referenced by init().

#### 7.12.3.37 on\_menu\_print()

```
void KitListGui::on_menu_print ( ) [protected], [virtual]
```

Prints the kit list.

Definition at line 663 of file kitlistgui.cpp.

References KitPrintOperation::create(), Service::get\_filtered\_items(), get\_selected\_category(), m\_page\_title, m\_ref\_page\_setup, m\_ref\_printer\_settings, m\_service, and on\_printoperation\_done().

Referenced by init().

### 7.12.3.38 on\_menu\_quit()

```
void KitListGui::on_menu_quit ( ) [protected], [virtual]
```

Called when the user chooses to quit the application.

Definition at line 370 of file kitlistgui.cpp.

References `confirm_lose_changes()`, `Service::is_model_dirty()`, `m_service`, `m_window`, and `Service::set_model_dirty()`.

Referenced by `init()`.

### 7.12.3.39 on\_menu\_recent\_file()

```
void KitListGui::on_menu_recent_file (
    const Glib::ustring & filename ) [protected], [virtual]
```

displays the most recent files menu

Definition at line 742 of file kitlistgui.cpp.

References `confirm_lose_changes()`, `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_CURRENT_FILE_NAME`, `Service::is_model_dirty()`, `m_filename`, `m_service`, `m_yaml_config`, `Service::open_as_xml()`, `refresh_category_list()`, `refresh_item_list()`, `YamlConfig::set_current_filename()`, and `update_recent_files()`.

Referenced by `update_recent_files_menu()`.

### 7.12.3.40 on\_menu\_rename\_category()

```
void KitListGui::on_menu_rename_category ( ) [protected], [virtual]
```

Called when the user chooses to rename a category.

Definition at line 1066 of file kitlistgui.cpp.

References `Service::find_category()`, `Category::get_name()`, `get_selected_category()`, `m_current_cat_id`, `m_entry_add_category`, `m_service`, `m_state`, `m_status_bar`, `m_window_add_category`, `RENAME_CATEGORY`, and `anonymous_namespace{kitlistgui.cpp}::SB_MSG`.

Referenced by `init()`.

#### 7.12.3.41 on\_menu\_save()

```
void KitListGui::on_menu_save ( ) [protected], [virtual]
```

Saves the current application state.

Definition at line 528 of file kitlistgui.cpp.

References `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_CURRENT_FILENAME`, `Service::is_model_dirty()`, `m_filename`, `m_service`, `m_status_bar`, `m_yaml_config`, `on_menu_save_as()`, `Service::require_filename()`, `Service::save()`, `Service::save_as_xml()`, `anonymous_namespace{kitlistgui.cpp}::SB_SAVE`, `YamlConfig::set_current_filename()`, and `update_recent_files()`.

Referenced by `init()`.

#### 7.12.3.42 on\_menu\_save\_as()

```
void KitListGui::on_menu_save_as ( ) [protected], [virtual]
```

Saves the current application state in a new document.

Definition at line 579 of file kitlistgui.cpp.

References `choose_filename()`, `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_CURRENT_FILENAME`, `m_filename`, `m_service`, `m_status_bar`, `m_yaml_config`, `Service::save_as_xml()`, `anonymous_namespace{kitlistgui.cpp}::SB_SAVE`, `YamlConfig::set_current_filename()`, and `update_recent_files()`.

Referenced by `init()`, and `on_menu_save()`.

#### 7.12.3.43 on\_menu\_select\_all()

```
void KitListGui::on_menu_select_all ( ) [protected], [virtual]
```

Called when the user chooses the 'select all' menu option.

Definition at line 960 of file kitlistgui.cpp.

References `m_item_tree_view`.

Referenced by `init()`.

#### 7.12.3.44 on\_menu\_show\_all()

```
virtual void KitListGui::on_menu_show_all ( ) [inline], [protected], [virtual]
```

Causes all items to be displayed.

Definition at line 218 of file kitlistgui.hpp.

References Service::show\_all().

Referenced by init().

#### 7.12.3.45 on\_menu\_show\_checked()

```
virtual void KitListGui::on_menu_show_checked ( ) [inline], [protected], [virtual]
```

Causes only checked items to be displayed.

Definition at line 220 of file kitlistgui.hpp.

References Service::show\_checked\_only().

Referenced by init().

#### 7.12.3.46 on\_menu\_show\_unchecked()

```
virtual void KitListGui::on_menu_show_unchecked ( ) [inline], [protected], [virtual]
```

Causes only unchecked items to be displayed.

Definition at line 222 of file kitlistgui.hpp.

References Service::show\_unchecked\_only().

Referenced by init().

#### 7.12.3.47 on\_menu\_uncheck\_selected()

```
virtual void KitListGui::on_menu_uncheck_selected ( ) [inline], [protected], [virtual]
```

Marks all selected items as unchecked.

Definition at line 227 of file kitlistgui.hpp.

Referenced by init().

#### 7.12.3.48 on\_printoperation\_done()

```
void KitListGui::on_printoperation_done (
    Gtk::PrintOperationResult result,
    const Glib::RefPtr< Gtk::PrintOperation > & op ) [protected]
```

Called when the printer operation is done.

Definition at line 648 of file kitlistgui.cpp.

References `m_ref_printer_settings`, `m_window`, and `on_printoperation_status_changed()`.

Referenced by `on_menu_print()`.

#### 7.12.3.49 on\_printoperation\_status\_changed()

```
void KitListGui::on_printoperation_status_changed (
    const Glib::RefPtr< Gtk::PrintOperation > & op ) [protected]
```

Called when the print status changes.

Definition at line 633 of file kitlistgui.cpp.

References `m_status_bar`, and `anonymous_namespace{kitlistgui.cpp}::SB_PRINT`.

Referenced by `on_printoperation_done()`.

#### 7.12.3.50 on\_row\_changed()

```
void KitListGui::on_row_changed (
    const Gtk::TreeModel::Path path,
    const Gtk::TreeModel::iterator iter ) [protected]
```

Callback method called when an item has been modified in the item list.

Sets the model as dirty and copies the row details to the appropriate [Item](#) in the model. The item's state is also set to dirty.

Definition at line 1497 of file kitlistgui.cpp.

References `ModelItemColumns::m_col_checked`, `ModelItemColumns::m_col_num`, `ModelItemColumns::m_col_text`, `m_ignore_list_events`, `m_item_cols`, `m_service`, `Service::set_model_dirty()`, and `Service::update_item()`.

Referenced by `init()`.

### 7.12.3.51 open\_file()

```
void KitListGui::open_file (
    const Glib::ustring & filename ) [virtual]
```

Opens an existing XML document.

Definition at line 481 of file kitlistgui.cpp.

References `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_CURRENT_FILENAME`, `m_filename`, `m_service`, `m_yaml_config`, `Service::open_as_xml()`, `refresh_category_list()`, `refresh_item_list()`, `YamlConfig::set_current_filename()`, and `update_recent_files()`.

Referenced by `on_menu_file_open()`, and `safe_open_file()`.

### 7.12.3.52 paste\_from\_xml()

```
void KitListGui::paste_from_xml (
    const Glib::ustring & document ) [protected], [virtual]
```

Performs a paste of items from the clipboard.

Definition at line 1184 of file kitlistgui.cpp.

References `add_items()`, `Service::find_item()`, `m_service`, and `anonymous_namespace{kitlistgui.cpp}::XML_ELEMENT_ID`.

Referenced by `on_menu_paste()`.

### 7.12.3.53 paste\_status\_received()

```
void KitListGui::paste_status_received (
    const Glib::StringArrayHandle & targets_array ) [protected], [virtual]
```

Callback method for enabling or disabling the paste menu option based on the clipboard contents.

See also

[KitListGui::update\\_paste\\_status\(\)](#)

Definition at line 1470 of file kitlistgui.cpp.

References `anonymous_namespace{kitlistgui.cpp}::item_target_custom`, `anonymous_namespace{kitlistgui.cpp}::item_target_text`, `m_paste_menu_item`, and `m_paste_tool_button`.

Referenced by `update_paste_status()`.

#### 7.12.3.54 raise()

```
void KitListGui::raise ( ) [virtual]
```

Make this application topmost.

Definition at line 1762 of file kitlistgui.cpp.

References `m_window`.

#### 7.12.3.55 refresh\_category\_list()

```
void KitListGui::refresh_category_list (
    long cat_id = -2 ) [protected], [virtual]
```

Refreshes the category combo box list.

##### Parameters

<code>cat↔ _id</code>	the id of the category to select in the combo box. If set to -2 the currently selected category is used, otherwise the specified category ID is used. If the category ID does not exist, or if -1 is specified, then no category is selected.
---------------------------	---

Definition at line 1582 of file kitlistgui.cpp.

References `Service::get_categories()`, `Category::get_id()`, `Category::get_name()`, `get_selected_category()`, `GuiState::is_deleted()`, `m_category_cols`, `m_category_combo`, `ModelCategoryColumns::m_col_num`, `Model↔  
CategoryColumns::m_col_text`, `m_ignore_list_events`, `m_ref_category_list_store`, and `m_service`.

Referenced by `close_add_category_window()`, `init()`, `on_menu_delete_category()`, `on_menu_file_new()`, `on_↔  
menu_recent_file()`, and `open_file()`.

#### 7.12.3.56 refresh\_item\_list()

```
void KitListGui::refresh_item_list ( ) [protected], [virtual]
```

Refreshes the item list.

Definition at line 1543 of file kitlistgui.cpp.

References `Service::filter()`, `Service::get_items()`, `get_selected_category()`, `ModelItemColumns::m_col_checked`, `ModelItemColumns::m_col_num`, `ModelItemColumns::m_col_text`, `m_ignore_list_events`, `m_item_cols`, `m_ref_↔  
item_tree_model`, `m_service`, and `update_item_count()`.

Referenced by `add_items()`, `close_add_category_window()`, `close_add_item_window()`, `delete_selected_items()`, `init()`, `on_category_change()`, `on_menu_cut()`, `on_menu_delete_category()`, `on_menu_file_new()`, `on_menu_↔  
recent_file()`, `open_file()`, `set_selected()`, and `toggle_selected()`.



### 7.12.3.57 run()

```
void KitListGui::run ( )
```

Starts the GUI application running.

Definition at line 232 of file kitlistgui.cpp.

References `Service::is_model_dirty()`, `m_kit`, `m_service`, and `m_window`.

### 7.12.3.58 safe\_open\_file()

```
void KitListGui::safe_open_file (
    const Glib::ustring & filename ) [virtual]
```

Opens an existing XML document, checking for unsaved changes.

If there are unsaved changes, the user is first prompted to discard them or cancel the operation.

Definition at line 514 of file kitlistgui.cpp.

References `confirm_lose_changes()`, `Service::is_model_dirty()`, `m_service`, `m_status_bar`, `open_file()`, and `anonymous_namespace{kitlistgui.cpp}::SB_SAVE`.

### 7.12.3.59 selected\_row\_callback()

```
void KitListGui::selected_row_callback (
    const Gtk::TreeModel::iterator & iter ) [protected], [virtual]
```

Called to delete an [Item](#) referenced by a row iterator.

Definition at line 1513 of file kitlistgui.cpp.

References `Service::delete_item()`, `ModelItemColumns::m_col_num`, `m_item_cols`, and `m_service`.

Referenced by `delete_selected_items()`.

### 7.12.3.60 set\_page\_title()

```
void KitListGui::set_page_title (
    const Glib::ustring page_title ) [protected], [virtual]
```

Definition at line 1647 of file kitlistgui.cpp.

References `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_PAGE_TITLE`, `m_page_title`, `m_yaml_config`, and `YamlConfig::set_page_title()`.

Referenced by `close_preferences_window()`.

#### 7.12.3.61 set\_selected()

```
void KitListGui::set_selected (
    bool checked ) [protected], [virtual]
```

Checks or unchecks the currently selected items.

Definition at line 1108 of file kitlistgui.cpp.

References `get_selected_items()`, `m_service`, `refresh_item_list()`, and `Service::select_items()`.

#### 7.12.3.62 toggle\_selected()

```
void KitListGui::toggle_selected ( ) [protected], [virtual]
```

Toggles the checked state of the currently selected items.

Definition at line 1119 of file kitlistgui.cpp.

References `get_selected_items()`, `m_service`, `refresh_item_list()`, and `Service::toggle_selected_items()`.

Referenced by `init()`.

#### 7.12.3.63 update\_item\_count()

```
void KitListGui::update_item_count (
    size_t n ) [protected], [virtual]
```

Writes the count of currently displayed items to the status bar.

Definition at line 1234 of file kitlistgui.cpp.

References `m_status_bar`, and anonymous\_namespace{kitlistgui.cpp}::SB\_ITEM\_COUNT.

Referenced by `init()`, and `refresh_item_list()`.

#### 7.12.3.64 update\_paste\_status()

```
void KitListGui::update_paste_status ( ) [protected], [virtual]
```

Enables or disables the paste menu option.

Definition at line 1458 of file kitlistgui.cpp.

References `paste_status_received()`.

Referenced by `copy_selected_items_to_clipboard()`, and `on_menu_paste()`.

#### 7.12.3.65 update\_recent\_files()

```
void KitListGui::update_recent_files (
    const Glib::ustring & filename ) [protected], [virtual]
```

Updates the list of most recently used files.

## Parameters

<i>filename</i>	The filename to append to the list if it does not already exist.
-----------------	--

Definition at line 324 of file kitlistgui.cpp.

References `YamlConfig::add_recent_filename()`, `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_RECENT_FILES`, `get_max_recent_files()`, `m_yaml_config`, and `update_recent_files_menu()`.

Referenced by `on_menu_recent_file()`, `on_menu_save()`, `on_menu_save_as()`, and `open_file()`.

## 7.12.3.66 update\_recent\_files\_menu()

```
void KitListGui::update_recent_files_menu ( ) [protected], [virtual]
```

Updates the recent files sub menu.

Definition at line 285 of file kitlistgui.cpp.

References `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_RECENT_FILES`, `YamlConfig::get_recent_filenames()`, `m_recent_files_menu_item`, `m_yaml_config`, and `on_menu_recent_file()`.

Referenced by `init()`, and `update_recent_files()`.

## 7.12.4 Member Data Documentation

## 7.12.4.1 m\_category\_cols

```
ModelCategoryColumns KitListGui::m_category_cols [protected]
```

The definition of the category combo box columns.

Definition at line 158 of file kitlistgui.hpp.

Referenced by `get_selected_category()`, `init()`, and `refresh_category_list()`.

## 7.12.4.2 m\_category\_combo

```
Gtk::ComboBox* KitListGui::m_category_combo [protected]
```

The combo box holding a list of categories.

Definition at line 154 of file kitlistgui.hpp.

Referenced by `get_selected_category()`, `init()`, and `refresh_category_list()`.

#### 7.12.4.3 m\_checkbutton\_add\_item

```
Gtk::CheckButton* KitListGui::m_checkbutton_add_item [protected]
```

The check button field of the 'Add [Item](#)' dialog.

Definition at line 152 of file kitlistgui.hpp.

Referenced by `close_add_item_window()`, and `init()`.

#### 7.12.4.4 m\_clipboard\_items

```
Glib::ustring KitListGui::m_clipboard_items [protected]
```

Holder for items pasted to the clipboard.

Definition at line 122 of file kitlistgui.hpp.

Referenced by `copy_selected_items_to_clipboard()`, `on_clipboard_get()`, `on_clipboard_received()`, and `on_menu_↔_paste()`.

#### 7.12.4.5 m\_current\_cat\_id

```
long KitListGui::m_current_cat_id [protected]
```

temporary reference to a category id, usually being renamed

Definition at line 181 of file kitlistgui.hpp.

Referenced by `close_add_category_window()`, `on_menu_create_category()`, and `on_menu_rename_category()`.

#### 7.12.4.6 m\_entry\_add\_category

```
Gtk::Entry* KitListGui::m_entry_add_category [protected]
```

The text entry field of the 'Add [Category](#)' dialog.

Definition at line 140 of file kitlistgui.hpp.

Referenced by `close_add_category_window()`, `init()`, `on_menu_create_category()`, and `on_menu_rename_↔_category()`.

#### 7.12.4.7 m\_entry\_add\_item

Gtk::Entry\* KitListGui::m\_entry\_add\_item [protected]

The text entry field of the 'Add Item' dialog.

Definition at line 138 of file kitlistgui.hpp.

Referenced by close\_add\_item\_window(), init(), init\_add\_item\_window(), and on\_menu\_add().

#### 7.12.4.8 m\_entry\_page\_title

Gtk::Entry\* KitListGui::m\_entry\_page\_title [protected]

the text entry field for the page title

Definition at line 132 of file kitlistgui.hpp.

Referenced by close\_preferences\_window(), init(), and on\_menu\_preferences().

#### 7.12.4.9 m\_file\_save\_menu\_item

Gtk::ImageMenuItem\* KitListGui::m\_file\_save\_menu\_item [protected]

The file save menu item.

Definition at line 142 of file kitlistgui.hpp.

Referenced by init().

#### 7.12.4.10 m\_file\_save\_tool\_button

Gtk::ToolButton\* KitListGui::m\_file\_save\_tool\_button [protected]

The file save toolbar button.

Definition at line 144 of file kitlistgui.hpp.

Referenced by init().

#### 7.12.4.11 m\_filename

```
Glib::ustring KitListGui::m_filename [protected]
```

The filename currently associated with the loaded model.

Should be an empty string if not related to a file.

Definition at line 114 of file kitlistgui.hpp.

Referenced by `confirm_lose_changes()`, `init()`, `on_menu_file_new()`, `on_menu_recent_file()`, `on_menu_save()`, `on_menu_save_as()`, and `open_file()`.

#### 7.12.4.12 m\_ignore\_list\_events

```
bool KitListGui::m_ignore_list_events [protected]
```

Temporarily ignore events on the item list.

Definition at line 124 of file kitlistgui.hpp.

Referenced by `init()`, `on_category_change()`, `on_cell_edit()`, `on_row_changed()`, `refresh_category_list()`, and `refresh_item_list()`.

#### 7.12.4.13 m\_item\_cols

```
ModelItemColumns KitListGui::m_item_cols [protected]
```

The definition of the item list's columns.

Definition at line 162 of file kitlistgui.hpp.

Referenced by `delete_selected_items()`, `get_selected_items()`, `init()`, `on_row_changed()`, `refresh_item_list()`, and `selected_row_callback()`.

#### 7.12.4.14 m\_item\_tree\_view

```
Gtk::TreeView* KitListGui::m_item_tree_view [protected]
```

The item list view definition.

Definition at line 160 of file kitlistgui.hpp.

Referenced by `delete_selected_items()`, `get_selected_items()`, `init()`, and `on_menu_select_all()`.

#### 7.12.4.15 m\_kit

Gtk::Main KitListGui::m\_kit [protected]

The main application.

Definition at line 126 of file kitlistgui.hpp.

Referenced by run().

#### 7.12.4.16 m\_page\_title

Glib::ustring KitListGui::m\_page\_title [protected]

The page title to be used when printing the item list.

Definition at line 116 of file kitlistgui.hpp.

Referenced by init(), on\_menu\_export\_to\_pdf(), on\_menu\_preferences(), on\_menu\_print(), and set\_page\_title().

#### 7.12.4.17 m\_paste\_menu\_item

Gtk::ImageMenuItem\* KitListGui::m\_paste\_menu\_item [protected]

The menu paste button.

Definition at line 148 of file kitlistgui.hpp.

Referenced by init(), and paste\_status\_received().

#### 7.12.4.18 m\_paste\_tool\_button

Gtk::ToolButton\* KitListGui::m\_paste\_tool\_button [protected]

The toolbar paste button.

Definition at line 150 of file kitlistgui.hpp.

Referenced by init(), and paste\_status\_received().

#### 7.12.4.19 m\_recent\_files\_menu\_item

```
Gtk::MenuItem* KitListGui::m_recent_files_menu_item [protected]
```

The recent files menu item.

Definition at line 146 of file kitlistgui.hpp.

Referenced by `init()`, and `update_recent_files_menu()`.

#### 7.12.4.20 m\_ref\_category\_list\_store

```
Glib::RefPtr<Gtk::ListStore> KitListGui::m_ref_category_list_store [protected]
```

The model backing the category combo box.

Definition at line 156 of file kitlistgui.hpp.

Referenced by `init()`, and `refresh_category_list()`.

#### 7.12.4.21 m\_ref\_item\_tree\_model

```
Glib::RefPtr<Gtk::ListStore> KitListGui::m_ref_item_tree_model [protected]
```

The model backing the item list.

Definition at line 166 of file kitlistgui.hpp.

Referenced by `delete_selected_items()`, `get_selected_items()`, `init()`, and `refresh_item_list()`.

#### 7.12.4.22 m\_ref\_page\_setup

```
Glib::RefPtr<Gtk::PageSetup> KitListGui::m_ref_page_setup [protected]
```

Printer page setup settings.

Definition at line 170 of file kitlistgui.hpp.

Referenced by `KitListGui()`, `on_menu_export_to_pdf()`, and `on_menu_print()`.



#### 7.12.4.23 m\_ref\_printer\_settings

```
Glib::RefPtr<Gtk::PrintSettings> KitListGui::m_ref_printer_settings [protected]
```

Printer settings.

Definition at line 172 of file kitlistgui.hpp.

Referenced by KitListGui(), on\_menu\_export\_to\_pdf(), on\_menu\_print(), and on\_printoperation\_done().

#### 7.12.4.24 m\_service

```
Service& KitListGui::m_service [protected]
```

The business/service object.

Definition at line 164 of file kitlistgui.hpp.

Referenced by add\_items(), close\_add\_category\_window(), close\_add\_item\_window(), confirm\_lose\_changes(), delete\_selected\_items(), get\_selected\_items(), init(), on\_cell\_edit(), on\_delete\_event(), on\_menu\_cut(), on\_menu\_delete\_category(), on\_menu\_export\_to\_pdf(), on\_menu\_file\_new(), on\_menu\_file\_open(), on\_menu\_print(), on\_menu\_quit(), on\_menu\_recent\_file(), on\_menu\_rename\_category(), on\_menu\_save(), on\_menu\_save\_as(), on\_row\_changed(), open\_file(), paste\_from\_xml(), refresh\_category\_list(), refresh\_item\_list(), run(), safe\_open\_file(), selected\_row\_callback(), set\_selected(), and toggle\_selected().

#### 7.12.4.25 m\_state

```
enum gui_state KitListGui::m_state [protected]
```

Indicates whether a category is being created or renamed.

Only used whilst the 'Add/Rename dialog is being displayed', or when it has been closed to determine the correct action.

Definition at line 180 of file kitlistgui.hpp.

Referenced by close\_add\_category\_window(), on\_menu\_create\_category(), and on\_menu\_rename\_category().

#### 7.12.4.26 m\_status\_bar

```
Gtk::Statusbar* KitListGui::m_status_bar [protected]
```

The application status bar.

Definition at line 168 of file kitlistgui.hpp.

Referenced by init(), on\_menu\_cut(), on\_menu\_delete\_category(), on\_menu\_export\_to\_pdf(), on\_menu\_file\_new(), on\_menu\_file\_open(), on\_menu\_rename\_category(), on\_menu\_save(), on\_menu\_save\_as(), on\_printoperation\_status\_changed(), safe\_open\_file(), and update\_item\_count().

#### 7.12.4.27 m\_window

```
Gtk::Window* KitListGui::m_window [protected]
```

The main application window.

Definition at line 128 of file kitlistgui.hpp.

Referenced by `choose_filename()`, `choose_pdf_filename()`, `confirm_lose_changes()`, `init()`, `on_menu_delete()`, `on_menu_delete_category()`, `on_menu_file_open()`, `on_menu_quit()`, `on_printoperation_done()`, `raise()`, and `run()`.

#### 7.12.4.28 m\_window\_add\_category

```
Gtk::Window* KitListGui::m_window_add_category [protected]
```

The 'Add [Category](#)' dialog.

Definition at line 136 of file kitlistgui.hpp.

Referenced by `cancel_add_category_window()`, `close_add_category_window()`, `init()`, `on_menu_create_category()`, and `on_menu_rename_category()`.

#### 7.12.4.29 m\_window\_add\_item

```
Gtk::Window* KitListGui::m_window_add_item [protected]
```

The 'Add [Item](#)' dialog.

Definition at line 134 of file kitlistgui.hpp.

Referenced by `cancel_add_item_window()`, `close_add_item_window()`, `init()`, and `on_menu_add()`.

#### 7.12.4.30 m\_window\_preferences

```
Gtk::Window* KitListGui::m_window_preferences [protected]
```

The 'Preferences' dialog.

Definition at line 130 of file kitlistgui.hpp.

Referenced by `cancel_preferences_window()`, `close_preferences_window()`, `init()`, and `on_menu_preferences()`.

## 7.12.4.31 m\_yaml\_config

```
YamlConfig KitListGui::m_yaml_config [private]
```

Definition at line 102 of file kitlistgui.hpp.

Referenced by `init()`, `on_menu_file_new()`, `on_menu_recent_file()`, `on_menu_save()`, `on_menu_save_as()`, `open_file()`, `set_page_title()`, `update_recent_files()`, and `update_recent_files_menu()`.

The documentation for this class was generated from the following files:

- [/home/frank/Projects/kitlist/src/kitlistgui.hpp](#)
- [/home/frank/Projects/kitlist/src/kitlistgui.cpp](#)

## 7.13 KitModel Class Reference

Holds a rich graph of objects representing the application's data model.

```
#include <kitmodel.hpp>
```

### Public Member Functions

- [KitModel](#) ()  
*Creates an empty data model.*
- [~KitModel](#) ()  
*Destructor.*
- void [foreach\\_item\\_iter](#) (const [SlotForeachModellItemIter](#) &slot)  
*Calls the provided slot for each item in the map.*
- void [foreach\\_item](#) (const [SlotForeachModellItem](#) &slot)  
*Calls the provided slot for each item in the map.*
- void [foreach\\_category\\_iter](#) (const [SlotForeachCategoryIter](#) &slot)  
*Calls the provided slot for each category in the map.*
- void [foreach\\_category](#) (const [SlotForeachCategory](#) &slot)  
*Calls the provided slot for each category in the map.*
- virtual [ModelCategory](#) \* [find\\_category](#) (long id)  
*Finds a [Category](#) by it's unique ID.*
- virtual [ModellItem](#) \* [find\\_item](#) (long id)  
*Finds an item by it's unique ID.*
- virtual [CategoryContainer](#) \* [get\\_categories](#) ()  
*Returns a list of all categories.*
- virtual [ItemContainer](#) \* [get\\_all\\_items](#) ()  
*Returns a list of all items.*
- virtual [ItemContainer](#) \* [get\\_all\\_items](#) ([ItemFuncor](#) &funcor)  
*Returns a list of all items, filtered by the passed functor.*
- virtual void [add\\_category](#) ([ModelCategory](#) \*category)  
*Add a category to the model.*
- virtual void [add\\_item](#) ([ModellItem](#) \*item)  
*Adds an item to the model.*
- virtual void [add\\_item](#) ([ModellItem](#) \*item, long cat\_id)

- Adds an item to the model and associates it with the specified category.*

  - virtual void [copy\\_items](#) (const [ModellItemContainer](#) &items, long cat\_id=-1)

*Copies items to the specified category.*
- virtual bool [filter](#) (bool checked)

*Applies the current filter.*
- virtual void [set\\_dirty](#) (bool dirty=true)
  - virtual bool [is\\_dirty](#) ()
  - virtual void [show\\_all](#) ()

*Removes filter. All items are shown.*
- virtual void [show\\_checked\\_only](#) ()

*Sets the filter to show only checked items.*
- virtual void [show\\_unchecked\\_only](#) ()

*Sets the filter to show only unchecked items.*
- virtual void [reset](#) ()

*Resets all contained objects to their default states.*
- virtual void [purge](#) ()

*Purges deleted categories and items from the model.*

## Protected Attributes

- bool [m\\_dirty](#)

*Indicates whether the model needs saving. I.e it's state has changed.*
- [CategoryMap](#) \* [m\\_category\\_map](#)

*Map allowing categories to be located by ID.*
- [ItemMap](#) \* [m\\_item\\_map](#)

*Map allowing items to be located by ID.*
- enum [item\\_filter\\_types](#) [m\\_item\\_filter](#)

*Indicates whether and how items are currently filtered.*

### 7.13.1 Detailed Description

Holds a rich graph of objects representing the application's data model.

Definition at line 135 of file kitmodel.hpp.

### 7.13.2 Constructor & Destructor Documentation

#### 7.13.2.1 KitModel()

```
KitModel::KitModel ( )
```

Creates an empty data model.

Definition at line 187 of file kitmodel.cpp.

References [m\\_category\\_map](#), and [m\\_item\\_map](#).

### 7.13.2.2 ~KitModel()

```
KitModel::~~KitModel ( )
```

Destructor.

Deletes all items and categories belonging to the model.

Definition at line 198 of file kitmodel.cpp.

References `m_category_map`, and `m_item_map`.

## 7.13.3 Member Function Documentation

### 7.13.3.1 add\_category()

```
void KitModel::add_category (
    ModelCategory * category ) [virtual]
```

Add a category to the model.

The category is included in the model's map.

Definition at line 361 of file kitmodel.cpp.

References `Category::get_id()`, and `m_category_map`.

Referenced by `Service::create_category()`, and `KitParser::process_category()`.

### 7.13.3.2 add\_item() [1/2]

```
void KitModel::add_item (
    ModelItem * item ) [virtual]
```

Adds an item to the model.

The item is included in the model's map.

Definition at line 371 of file kitmodel.cpp.

References `Item::get_id()`, and `m_item_map`.

Referenced by `Service::create_item()`, and `KitParser::process_item()`.

### 7.13.3.3 add\_item() [2/2]

```
void KitModel::add_item (
    ModelItem * item,
    long cat_id ) [virtual]
```

Adds an item to the model and associates it with the specified category.

The category must have already been added to the model's map.

Definition at line 382 of file kitmodel.cpp.

References ModelCategory::add\_item(), find\_category(), Item::get\_id(), and m\_item\_map.

### 7.13.3.4 copy\_items()

```
void KitModel::copy_items (
    const ModelItemContainer & items,
    long cat_id = -1 ) [virtual]
```

Copies items to the specified category.

Only copies those items that do not already exist in the target category.

Definition at line 399 of file kitmodel.cpp.

References ModelCategory::add\_item(), find\_category(), Category::m\_items, and GuiState::set\_dirty().

Referenced by Service::copy\_items().

### 7.13.3.5 filter()

```
bool KitModel::filter (
    bool checked ) [virtual]
```

Applies the current filter.

#### Parameters

<i>checked</i>	The checked/ticked state of the item being filtered.
----------------	--

#### Returns

true if the item should be included.

Definition at line 446 of file kitmodel.cpp.

References ALL, CHECKED, m\_item\_filter, and UNCHECKED.

Referenced by Service::filter(), and FilterItem::operator().

#### 7.13.3.6 find\_category()

```
ModelCategory * KitModel::find_category (
    long id ) [virtual]
```

Finds a [Category](#) by it's unique ID.

Definition at line 265 of file kitmodel.cpp.

References m\_category\_map.

Referenced by add\_item(), copy\_items(), Service::delete\_category(), Service::find\_category(), Service::get\_↔ filtered\_items(), and Service::get\_items().

#### 7.13.3.7 find\_item()

```
ModelItem * KitModel::find_item (
    long id ) [virtual]
```

Finds an item by it's unique ID.

Definition at line 278 of file kitmodel.cpp.

References m\_item\_map.

Referenced by Service::delete\_item(), Service::find\_item(), KitParser::process\_category\_item(), and Service↔ ::update\_item().

#### 7.13.3.8 foreach\_category()

```
void KitModel::foreach_category (
    const SlotForeachCategory & slot )
```

Calls the provided slot for each category in the map.

Definition at line 254 of file kitmodel.cpp.

References m\_category\_map.

Referenced by XmlDao::save\_model().

### 7.13.3.9 foreach\_category\_iter()

```
void KitModel::foreach_category_iter (
    const SlotForeachCategoryIter & slot )
```

Calls the provided slot for each category in the map.

Definition at line 242 of file kitmodel.cpp.

References m\_category\_map.

### 7.13.3.10 foreach\_item()

```
void KitModel::foreach_item (
    const SlotForeachModelItem & slot )
```

Calls the provided slot for each item in the map.

Definition at line 230 of file kitmodel.cpp.

References m\_item\_map.

Referenced by XmlDao::save\_model().

### 7.13.3.11 foreach\_item\_iter()

```
void KitModel::foreach_item_iter (
    const SlotForeachModelItemIter & slot )
```

Calls the provided slot for each item in the map.

Definition at line 218 of file kitmodel.cpp.

References m\_item\_map.

### 7.13.3.12 get\_all\_items() [1/2]

```
ItemContainer * KitModel::get_all_items ( ) [virtual]
```

Returns a list of all items.

Excluded items are excluded from the list. The caller is responsible for deleting the returned item list.

Definition at line 327 of file kitmodel.cpp.

References m\_item\_map.

Referenced by Service::get\_filtered\_items(), Service::get\_items(), and XmlDao::get\_model().



#### 7.13.3.13 `get_all_items()` [2/2]

```
ItemContainer * KitModel::get_all_items (
    ItemFunctor & functor ) [virtual]
```

Returns a list of all items, filtered by the passed functor.

Excluded items are excluded from the list. The caller is responsible for deleting the returned item list.

Definition at line 345 of file `kitmodel.cpp`.

References `m_item_map`.

#### 7.13.3.14 `get_categories()`

```
CategoryContainer * KitModel::get_categories ( ) [virtual]
```

Returns a list of all categories.

Definition at line 300 of file `kitmodel.cpp`.

References `GuiState::is_deleted()`, and `m_category_map`.

Referenced by `Service::get_categories()`, and `XmlDao::get_model()`.

#### 7.13.3.15 `is_dirty()`

```
virtual bool KitModel::is_dirty ( ) [inline], [virtual]
```

Definition at line 177 of file `kitmodel.hpp`.

References `GuiState::m_dirty`.

Referenced by `Service::is_model_dirty()`.

#### 7.13.3.16 `purge()`

```
void KitModel::purge ( ) [virtual]
```

Purges deleted categories and items from the model.

Typically, this method is called after a save operation, but before calling `KitModel::reset()`

Definition at line 486 of file `kitmodel.cpp`.

References `GuiState::is_deleted()`, `m_category_map`, `m_item_map`, and `ModelCategory::purge()`.

Referenced by `XmlDao::save_model()`.

### 7.13.3.17 reset()

```
void KitModel::reset ( ) [virtual]
```

Resets all contained objects to their default states.

This method needs to be called after load or save operations to ensure all the dirty, deleted and new flags of each object are reset.

After a save operation, [KitModel::purge\(\)](#) should be called before this method.

Definition at line 466 of file kitmodel.cpp.

References [GuiState::is\\_deleted\(\)](#), [m\\_category\\_map](#), [m\\_dirty](#), [m\\_item\\_map](#), [GuiState::reset\(\)](#), and [ModelCategory::reset\(\)](#).

Referenced by [Service::create\\_default\\_model\(\)](#), [XmlDao::get\\_model\(\)](#), and [XmlDao::save\\_model\(\)](#).

### 7.13.3.18 set\_dirty()

```
virtual void KitModel::set_dirty (
    bool dirty = true ) [inline], [virtual]
```

Definition at line 176 of file kitmodel.hpp.

Referenced by [Service::copy\\_items\(\)](#), [Service::create\\_category\(\)](#), [Service::create\\_item\(\)](#), [Service::delete\\_category\(\)](#), [Service::delete\\_item\(\)](#), [Service::select\\_items\(\)](#), [Service::set\\_model\\_dirty\(\)](#), and [Service::toggle\\_selected\\_items\(\)](#).

### 7.13.3.19 show\_all()

```
virtual void KitModel::show_all ( ) [inline], [virtual]
```

Removes filter. All items are shown.

Definition at line 179 of file kitmodel.hpp.

References ALL.

Referenced by [Service::show\\_all\(\)](#).

#### 7.13.3.20 show\_checked\_only()

```
virtual void KitModel::show_checked_only ( ) [inline], [virtual]
```

Sets the filter to show only checked items.

Definition at line 181 of file kitmodel.hpp.

References CHECKED.

Referenced by Service::show\_checked\_only().

#### 7.13.3.21 show\_unchecked\_only()

```
virtual void KitModel::show_unchecked_only ( ) [inline], [virtual]
```

Sets the filter to show only unchecked items.

Definition at line 183 of file kitmodel.hpp.

References GuiState::reset(), and UNCHECKED.

Referenced by Service::show\_unchecked\_only().

### 7.13.4 Member Data Documentation

#### 7.13.4.1 m\_category\_map

```
CategoryMap* KitModel::m_category_map [protected]
```

Map allowing categories to be located by ID.

The map's key is the category ID, with the second field of the pair containing a pointer to the [ModelCategory](#) instance.

Definition at line 145 of file kitmodel.hpp.

Referenced by add\_category(), find\_category(), foreach\_category(), foreach\_category\_iter(), get\_categories(), KitModel(), purge(), reset(), and ~KitModel().

#### 7.13.4.2 m\_dirty

```
bool KitModel::m_dirty [protected]
```

Indicates whether the model needs saving. I.e it's state has changed.

Definition at line 138 of file kitmodel.hpp.

Referenced by reset().

#### 7.13.4.3 m\_item\_filter

```
enum item_filter_types KitModel::m_item_filter [protected]
```

Indicates whether and how items are currently filtered.

One of ALL, CHECKED or UNCHECKED.

Definition at line 158 of file kitmodel.hpp.

Referenced by filter().

#### 7.13.4.4 m\_item\_map

```
ItemMap* KitModel::m_item_map [protected]
```

Map allowing items to be located by ID.

The map's key is the item ID, with the second field of the pair containing a pointer to the [ModelItem](#) instance.

Definition at line 152 of file kitmodel.hpp.

Referenced by add\_item(), find\_item(), foreach\_item(), foreach\_item\_iter(), get\_all\_items(), KitModel(), purge(), reset(), and ~KitModel().

The documentation for this class was generated from the following files:

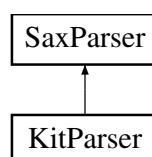
- [/home/frank/Projects/kitlist/src/kitmodel.hpp](#)
- [/home/frank/Projects/kitlist/src/kitmodel.cpp](#)

## 7.14 KitParser Class Reference

SaxParser implementation for reading the [KitModel](#) from an XML document.

```
#include <kitparser.hpp>
```

Inheritance diagram for KitParser:



## Public Member Functions

- [KitParser](#) ([KitModel](#) &model)  
*Constructor taking the rich data model to save the XML document within.*
- virtual [~KitParser](#) ()

## Protected Member Functions

- virtual void [on\\_start\\_document](#) ()  
*Does nothing. Called at the start of a document.*
- virtual void [on\\_end\\_document](#) ()  
*Does nothing. Called at the end of a document.*
- virtual void [on\\_start\\_element](#) (const Glib::ustring &name, const AttributeList &attributes)  
*Called for each element.*
- virtual void [on\\_end\\_element](#) (const Glib::ustring &name)  
*Called at the end of each element.*
- virtual void [on\\_characters](#) (const Glib::ustring &text)  
*Called for each piece of CDATA belonging to an element.*
- virtual void [on\\_comment](#) (const Glib::ustring &text)  
*Does nothing. Called for each comment.*
- virtual void [on\\_warning](#) (const Glib::ustring &text)  
*Outputs any warnings to STDOUT.*
- virtual void [on\\_error](#) (const Glib::ustring &text)  
*Outputs any errors to STDOUT.*
- virtual void [on\\_fatal\\_error](#) (const Glib::ustring &text)  
*Outputs any fatal errors to STDOUT.*

## Protected Attributes

- [KitModel](#) & [m\\_model](#)

## Private Member Functions

- void [process\\_item](#) (const AttributeList &attributes)  
*< The most recently processed CDATA*
- void [process\\_category](#) (const AttributeList &attributes)  
*Reads a category's attributes from a 'category' element.*
- void [process\\_category\\_item](#) (const AttributeList &attributes)  
*Reads details of an item association with a category from a 'category-item' element.*

## Private Attributes

- [ModelCategory](#) \* [m\\_category](#)  
*The most recently processed category element.*
- [ModelItem](#) \* [m\\_item](#)  
*The most recently processed item element.*
- Glib::ustring [m\\_cdata](#)

### 7.14.1 Detailed Description

SaxParser implementation for reading the [KitModel](#) from an XML document.

Definition at line 35 of file kitparser.hpp.

### 7.14.2 Constructor & Destructor Documentation

#### 7.14.2.1 KitParser()

```
KitParser::KitParser (
    KitModel & model ) [inline]
```

Constructor taking the rich data model to save the XML document within.

Definition at line 45 of file kitparser.hpp.

#### 7.14.2.2 ~KitParser()

```
virtual KitParser::~~KitParser ( ) [inline], [virtual]
```

Definition at line 46 of file kitparser.hpp.

### 7.14.3 Member Function Documentation

#### 7.14.3.1 on\_characters()

```
void KitParser::on_characters (
    const Glib::ustring & text ) [protected], [virtual]
```

Called for each piece of CDATA belonging to an element.

This may be called a number of times, each time passing a bit more of the parsed CDATA. The processing of the CDATA is split up by any embedded entities.

The CDATA is held in a private member variable. The CDATA variable is cleared at the start of each new element and appended to at each call to this method. The captured text is then set on the appropriate object during [KitParser::on\\_end\\_element\(\)](#).

Definition at line 164 of file kitparser.cpp.

References [m\\_cdata](#).

### 7.14.3.2 on\_comment()

```
void KitParser::on_comment (
    const Glib::ustring & text ) [protected], [virtual]
```

Does nothing. Called for each comment.

Definition at line 171 of file kitparser.cpp.

### 7.14.3.3 on\_end\_document()

```
void KitParser::on_end_document ( ) [protected], [virtual]
```

Does nothing. Called at the end of a document.

Definition at line 35 of file kitparser.cpp.

### 7.14.3.4 on\_end\_element()

```
void KitParser::on_end_element (
    const Glib::ustring & name ) [protected], [virtual]
```

Called at the end of each element.

If any CDATA existed in the element body, sets the text belonging to the appropriate element object with the previously captured CDATA.

Definition at line 141 of file kitparser.cpp.

References `m_category`, `m_cdata`, `m_item`, `Item::set_description()`, and `Category::set_name()`.

### 7.14.3.5 on\_error()

```
void KitParser::on_error (
    const Glib::ustring & text ) [protected], [virtual]
```

Outputs any errors to STDOUT.

Definition at line 183 of file kitparser.cpp.

#### 7.14.3.6 on\_fatal\_error()

```
void KitParser::on_fatal_error (
    const Glib::ustring & text ) [protected], [virtual]
```

Outputs any fatal errors to STDOUT.

Definition at line 189 of file kitparser.cpp.

#### 7.14.3.7 on\_start\_document()

```
void KitParser::on_start_document ( ) [protected], [virtual]
```

Does nothing. Called at the start of a document.

Definition at line 29 of file kitparser.cpp.

#### 7.14.3.8 on\_start\_element()

```
void KitParser::on_start_element (
    const Glib::ustring & name,
    const AttributeList & attributes ) [protected], [virtual]
```

Called for each element.

Determines which element is being processed and calls appropriate method to process the corresponding attributes.

Definition at line 115 of file kitparser.cpp.

References `m_cdata`, `process_category()`, `process_category_item()`, and `process_item()`.

#### 7.14.3.9 on\_warning()

```
void KitParser::on_warning (
    const Glib::ustring & text ) [protected], [virtual]
```

Outputs any warnings to STDOUT.

Definition at line 177 of file kitparser.cpp.

#### 7.14.3.10 process\_category()

```
void KitParser::process_category (
    const AttributeList & attributes ) [private]
```

Reads a category's attributes from a 'category' element.



**Parameters**

<i>attributes</i>	The list of attributes for the element.
-------------------	---

Definition at line 68 of file kitparser.cpp.

References KitModel::add\_category(), m\_category, m\_model, and Category::set\_id().

Referenced by on\_start\_element().

**7.14.3.11 process\_category\_item()**

```
void KitParser::process_category_item (
    const AttributeList & attributes ) [private]
```

Reads details of an item association with a category from a 'category-item' element.

**Parameters**

<i>attributes</i>	The list of attributes for the element.
-------------------	---

Definition at line 87 of file kitparser.cpp.

References ModelCategory::add\_item(), KitModel::find\_item(), Category::get\_id(), m\_category, and m\_model.

Referenced by on\_start\_element().

**7.14.3.12 process\_item()**

```
void KitParser::process_item (
    const AttributeList & attributes ) [private]
```

<The most recently processed CDATA

Reads an item's attributes from an 'item' element.

**Parameters**

<i>attributes</i>	The list of attributes for the element.
-------------------	---

Definition at line 45 of file kitparser.cpp.

References KitModel::add\_item(), m\_item, m\_model, ModelItem::set\_checked(), and Item::set\_id().

Referenced by on\_start\_element().

## 7.14.4 Member Data Documentation

### 7.14.4.1 m\_category

`ModelCategory*` `KitParser::m_category` [private]

The most recently processed category element.

Definition at line 37 of file `kitparser.hpp`.

Referenced by `on_end_element()`, `process_category()`, and `process_category_item()`.

### 7.14.4.2 m\_cdata

`Glib::ustring` `KitParser::m_cdata` [private]

Definition at line 39 of file `kitparser.hpp`.

Referenced by `on_characters()`, `on_end_element()`, and `on_start_element()`.

### 7.14.4.3 m\_item

`ModelItem*` `KitParser::m_item` [private]

The most recently processed item element.

Definition at line 38 of file `kitparser.hpp`.

Referenced by `on_end_element()`, and `process_item()`.

### 7.14.4.4 m\_model

`KitModel&` `KitParser::m_model` [protected]

Definition at line 48 of file `kitparser.hpp`.

Referenced by `process_category()`, `process_category_item()`, and `process_item()`.

The documentation for this class was generated from the following files:

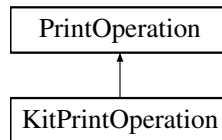
- [/home/frank/Projects/kitlist/src/kitparser.hpp](#)
- [/home/frank/Projects/kitlist/src/kitparser.cpp](#)

## 7.15 KitPrintOperation Class Reference

Prints the kitlist.

```
#include <printing.hpp>
```

Inheritance diagram for KitPrintOperation:



### Public Member Functions

- void [set\\_items](#) ([ItemContainer](#) \*items)  
*Sets the list of items to be printed.*
- void [set\\_page\\_title](#) (const Glib::ustring page\_title)
- [~KitPrintOperation](#) ()  
*Destructor.*

### Static Public Member Functions

- static Glib::RefPtr< [KitPrintOperation](#) > [create](#) ()  
*Factory to create instances.*

### Protected Member Functions

- [layout\\_refptr new\\_header](#) (const Glib::RefPtr< [Gtk::PrintContext](#) > &context)
- [layout\\_refptr new\\_footer](#) (const Glib::RefPtr< [Gtk::PrintContext](#) > &context)
- virtual void [on\\_begin\\_print](#) (const Glib::RefPtr< [Gtk::PrintContext](#) > &context)
- virtual void [on\\_draw\\_page](#) (const Glib::RefPtr< [Gtk::PrintContext](#) > &context, int page\_number)

### Protected Attributes

- [layout\\_refptr m\\_ref\\_layout](#)  
*A layout to hold the body of the entire kitlist to be printed.*
- std::vector< int > [m\\_page\\_breaks](#)  
*A list of line numbers where a page break is required.*
- std::vector< [layout\\_refptr](#) > [m\\_ref\\_headers](#)  
*A list of headers, one for each page.*
- std::vector< [layout\\_refptr](#) > [m\\_ref\\_footers](#)  
*A list of footers, one for each page.*

### Private Attributes

- [ItemContainer](#) \* [m\\_items](#)
- Glib::ustring [m\\_page\\_title](#)

### 7.15.1 Detailed Description

Prints the kitlist.

Definition at line 35 of file printing.hpp.

### 7.15.2 Constructor & Destructor Documentation

#### 7.15.2.1 ~KitPrintOperation()

```
KitPrintOperation::~KitPrintOperation ( )
```

Destructor.

Definition at line 44 of file printing.cpp.

References `m_items`.

Referenced by `set_page_title()`.

### 7.15.3 Member Function Documentation

#### 7.15.3.1 create()

```
Glib::RefPtr< KitPrintOperation > KitPrintOperation::create ( ) [static]
```

Factory to create instances.

Definition at line 51 of file printing.cpp.

Referenced by `KitListGui::on_menu_export_to_pdf()`, and `KitListGui::on_menu_print()`.

#### 7.15.3.2 new\_footer()

```
layout_refptr KitPrintOperation::new_footer (
    const Glib::RefPtr< Gtk::PrintContext > & context ) [protected]
```

Creates a new footer element for a page

## Parameters

<i>context</i>	the print context
----------------	-------------------

Definition at line 76 of file printing.cpp.

References FOOTER\_TEXT, and m\_ref\_footers.

Referenced by on\_begin\_print(), and set\_page\_title().

## 7.15.3.3 new\_header()

```
layout_refptr KitPrintOperation::new_header (
    const Glib::RefPtr< Gtk::PrintContext > & context ) [protected]
```

Creates a new header element for a page

## Parameters

<i>context</i>	the print context
----------------	-------------------

Definition at line 60 of file printing.cpp.

References m\_page\_title, and m\_ref\_headers.

Referenced by on\_begin\_print(), and set\_page\_title().

## 7.15.3.4 on\_begin\_print()

```
void KitPrintOperation::on_begin_print (
    const Glib::RefPtr< Gtk::PrintContext > & context ) [protected], [virtual]
```

Called prior to pages being printed. Calculates what is printed on each page.

## Parameters

<i>context</i>	the print context
----------------	-------------------

Definition at line 93 of file printing.cpp.

References BORDER\_SPACING, FOOTER\_SPACING, FOOTER\_TEXT, Item::get\_description(), HEADER\_SPACING, m\_items, m\_page\_breaks, m\_ref\_footers, m\_ref\_layout, new\_footer(), new\_header(), and PAGE\_TOLERANCE.

Referenced by set\_page\_title().

### 7.15.3.5 on\_draw\_page()

```
void KitPrintOperation::on_draw_page (
    const Glib::RefPtr< Gtk::PrintContext > & context,
    int page_number ) [protected], [virtual]
```

Prints a specified page.

#### Parameters

<i>context</i>	the print context
<i>page_number</i>	the number of the page to be printed

Definition at line 159 of file printing.cpp.

References BORDER\_SPACING, HEADER\_SPACING, m\_page\_breaks, m\_ref\_footers, m\_ref\_headers, and m\_ref\_layout.

Referenced by set\_page\_title().

### 7.15.3.6 set\_items()

```
void KitPrintOperation::set_items (
    ItemContainer * items ) [inline]
```

Sets the list of items to be printed.

Definition at line 41 of file printing.hpp.

### 7.15.3.7 set\_page\_title()

```
void KitPrintOperation::set_page_title (
    const Glib::ustring page_title ) [inline]
```

Definition at line 42 of file printing.hpp.

References new\_footer(), new\_header(), on\_begin\_print(), on\_draw\_page(), and ~KitPrintOperation().

## 7.15.4 Member Data Documentation

#### 7.15.4.1 m\_items

```
ItemContainer* KitPrintOperation::m_items [private]
```

Definition at line 36 of file printing.hpp.

Referenced by on\_begin\_print(), and ~KitPrintOperation().

#### 7.15.4.2 m\_page\_breaks

```
std::vector<int> KitPrintOperation::m_page_breaks [protected]
```

A list of line numbers where a page break is required.

Definition at line 52 of file printing.hpp.

Referenced by on\_begin\_print(), and on\_draw\_page().

#### 7.15.4.3 m\_page\_title

```
Glib::ustring KitPrintOperation::m_page_title [private]
```

Definition at line 37 of file printing.hpp.

Referenced by new\_header().

#### 7.15.4.4 m\_ref\_footers

```
std::vector<layout_refptr> KitPrintOperation::m_ref_footers [protected]
```

A list of footers, one for each page.

Definition at line 56 of file printing.hpp.

Referenced by new\_footer(), on\_begin\_print(), and on\_draw\_page().

#### 7.15.4.5 m\_ref\_headers

```
std::vector<layout_refptr> KitPrintOperation::m_ref_headers [protected]
```

A list of headers, one for each page.

Definition at line 54 of file printing.hpp.

Referenced by new\_header(), and on\_draw\_page().

#### 7.15.4.6 m\_ref\_layout

`layout_refptr` `KitPrintOperation::m_ref_layout` [protected]

A layout to hold the body of the entire kitlist to be printed.

Definition at line 50 of file `printing.hpp`.

Referenced by `on_begin_print()`, and `on_draw_page()`.

The documentation for this class was generated from the following files:

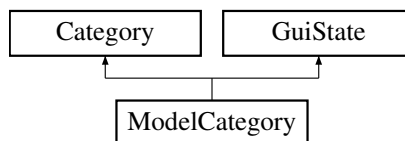
- `/home/frank/Projects/kitlist/src/printing.hpp`
- `/home/frank/Projects/kitlist/src/printing.cpp`

## 7.16 ModelCategory Class Reference

Represents a `Category` combined with `GuiState` attributes.

```
#include <kitmodel.hpp>
```

Inheritance diagram for `ModelCategory`:



### Public Member Functions

- `ModelCategory ()`
- `~ModelCategory ()`
- virtual `ModellItemContainer * get_model_items ()`
- virtual `ItemContainer * get_items ()`  
*Returns the list of items belonging to the `Category`.*
- virtual `ItemContainer * get_items (ItemFunctor &functor)`  
*Returns the list of items belonging to the `Category`, filtered by the passed functor.*
- virtual `ItemMap * get_removed_children ()`  
*Returns a list of items that have been removed from the `Category`.*
- virtual `ItemMap * get_added_children ()`  
*Returns a list of items that have been added to the `Category`.*
- virtual void `add_item (Item *)`  
*Adds the passed item to this `ModelCategory`.*
- virtual void `remove_item (Item *item)`  
*Removes the passed item from this `ModelCategory`.*
- virtual void `remove_items (ModellItemContainer *items)`  
*Removes all the passed items from this `ModelCategory`.*
- virtual void `reset ()`  
*Resets the `Category` state.*
- virtual void `purge ()`



## Protected Attributes

- [ItemMap](#) \* [m\\_removed\\_children](#)  
*List of items removed from the [Category](#).*
- [ItemMap](#) \* [m\\_added\\_children](#)  
*List of items added to the [Category](#).*

## Friends

- class [KitModel](#)

### 7.16.1 Detailed Description

Represents a [Category](#) combined with [GuiState](#) attributes.

Definition at line 94 of file [kitmodel.hpp](#).

### 7.16.2 Constructor & Destructor Documentation

#### 7.16.2.1 ModelCategory()

```
ModelCategory::ModelCategory ( )
```

Definition at line 43 of file [kitmodel.cpp](#).

References [m\\_added\\_children](#), and [m\\_removed\\_children](#).

#### 7.16.2.2 ~ModelCategory()

```
ModelCategory::~~ModelCategory ( )
```

Definition at line 49 of file [kitmodel.cpp](#).

References [m\\_added\\_children](#), and [m\\_removed\\_children](#).

### 7.16.3 Member Function Documentation

### 7.16.3.1 add\_item()

```
void ModelCategory::add_item (
    Item * item ) [virtual]
```

Adds the passed item to this [ModelCategory](#).

Also updates the lists of removed and added items.

Reimplemented from [Category](#).

Definition at line 88 of file kitmodel.cpp.

References [Category::add\\_item\(\)](#), [Item::get\\_id\(\)](#), [m\\_added\\_children](#), and [m\\_removed\\_children](#).

Referenced by [KitModel::add\\_item\(\)](#), [KitModel::copy\\_items\(\)](#), and [KitParser::process\\_category\\_item\(\)](#).

### 7.16.3.2 get\_added\_children()

```
virtual ItemMap* ModelCategory::get_added_children ( ) [inline], [virtual]
```

Returns a list of items that have been added to the [Category](#).

Definition at line 109 of file kitmodel.hpp.

References [GuiState::reset\(\)](#).

### 7.16.3.3 get\_items() [1/2]

```
ItemContainer * ModelCategory::get_items ( ) [virtual]
```

Returns the list of items belonging to the [Category](#).

Items flagged as deleted are excluded.

Definition at line 151 of file kitmodel.cpp.

References [GuiState::is\\_deleted\(\)](#), and [Category::m\\_items](#).

Referenced by [Service::get\\_filtered\\_items\(\)](#), and [Service::get\\_items\(\)](#).

### 7.16.3.4 get\_items() [2/2]

```
ItemContainer * ModelCategory::get_items (
    ItemFuncor & functor ) [virtual]
```

Returns the list of items belonging to the [Category](#), filtered by the passed functor.

Items flagged as deleted are excluded.

## Parameters

<i>functor</i>	if the operator() method returns true the item is included in the returned list.
----------------	--

Definition at line 172 of file kitmodel.cpp.

References [GuiState::is\\_deleted\(\)](#), and [Category::m\\_items](#).

### 7.16.3.5 get\_model\_items()

```
ModelItemContainer * ModelCategory::get_model_items ( ) [virtual]
```

Returns the list of items belonging to the [Category](#).

Items flagged as deleted are excluded.

Definition at line 134 of file kitmodel.cpp.

References [GuiState::is\\_deleted\(\)](#), and [Category::m\\_items](#).

### 7.16.3.6 get\_removed\_children()

```
virtual ItemMap* ModelCategory::get_removed_children ( ) [inline], [virtual]
```

Returns a list of items that have been removed from the [Category](#).

Definition at line 107 of file kitmodel.hpp.

### 7.16.3.7 purge()

```
void ModelCategory::purge ( ) [virtual]
```

Definition at line 68 of file kitmodel.cpp.

References [Category::m\\_items](#).

Referenced by [KitModel::purge\(\)](#).

### 7.16.3.8 remove\_item()

```
void ModelCategory::remove_item (
    Item * item ) [virtual]
```

Removes the passed item from this [ModelCategory](#).

Also updates the lists of removed and added items.

Reimplemented from [Category](#).

Definition at line 106 of file kitmodel.cpp.

References [Item::get\\_id\(\)](#), [m\\_added\\_children](#), [m\\_removed\\_children](#), and [Category::remove\\_item\(\)](#).

Referenced by [remove\\_items\(\)](#).

### 7.16.3.9 remove\_items()

```
void ModelCategory::remove_items (
    ModelItemContainer * items ) [virtual]
```

Removes all the passed items from this [ModelCategory](#).

Also updates the lists of removed and added items.

Definition at line 122 of file kitmodel.cpp.

References [remove\\_item\(\)](#).

Referenced by [KitListGui::on\\_menu\\_cut\(\)](#).

### 7.16.3.10 reset()

```
void ModelCategory::reset ( ) [virtual]
```

Resets the [Category](#) state.

Removes any items flagged as deleted. Sets the [GuiState](#) to it's defaults and clears the lists of added and removed items.

Reimplemented from [GuiState](#).

Definition at line 61 of file kitmodel.cpp.

References [m\\_added\\_children](#), [m\\_removed\\_children](#), and [GuiState::reset\(\)](#).

Referenced by [KitModel::reset\(\)](#).

## 7.16.4 Friends And Related Function Documentation

### 7.16.4.1 KitModel

```
friend class KitModel [friend]
```

Definition at line 115 of file kitmodel.hpp.

## 7.16.5 Member Data Documentation

### 7.16.5.1 m\_added\_children

```
ItemMap* ModelCategory::m_added_children [protected]
```

List of items added to the [Category](#).

Definition at line 99 of file kitmodel.hpp.

Referenced by [add\\_item\(\)](#), [ModelCategory\(\)](#), [remove\\_item\(\)](#), [reset\(\)](#), and [~ModelCategory\(\)](#).

### 7.16.5.2 m\_removed\_children

```
ItemMap* ModelCategory::m_removed_children [protected]
```

List of items removed from the [Category](#).

Definition at line 97 of file kitmodel.hpp.

Referenced by [add\\_item\(\)](#), [ModelCategory\(\)](#), [remove\\_item\(\)](#), [reset\(\)](#), and [~ModelCategory\(\)](#).

The documentation for this class was generated from the following files:

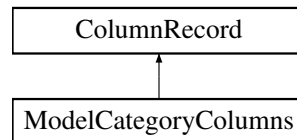
- [/home/frank/Projects/kitlist/src/kitmodel.hpp](#)
- [/home/frank/Projects/kitlist/src/kitmodel.cpp](#)

## 7.17 ModelCategoryColumns Class Reference

A definition for displaying a [ModelCategory](#) in a combo box.

```
#include <kitlistgui.hpp>
```

Inheritance diagram for ModelCategoryColumns:



### Public Member Functions

- [ModelCategoryColumns](#) ()

### Public Attributes

- `Gtk::TreeModelColumn< Glib::ustring >` [m\\_col\\_text](#)
- `Gtk::TreeModelColumn< int >` [m\\_col\\_num](#)

### 7.17.1 Detailed Description

A definition for displaying a [ModelCategory](#) in a combo box.

Definition at line 58 of file kitlistgui.hpp.

### 7.17.2 Constructor & Destructor Documentation

#### 7.17.2.1 ModelCategoryColumns()

```
ModelCategoryColumns::ModelCategoryColumns ( ) [inline]
```

Definition at line 61 of file kitlistgui.hpp.

References [m\\_col\\_num](#), and [m\\_col\\_text](#).

### 7.17.3 Member Data Documentation

## 7.17.3.1 m\_col\_num

```
Gtk::TreeModelColumn<int> ModelCategoryColumns::m_col_num
```

Definition at line 67 of file kitlistgui.hpp.

Referenced by KitListGui::get\_selected\_category(), ModelCategoryColumns(), ModellItemColumns::ModellItemColumns(), and KitListGui::refresh\_category\_list().

## 7.17.3.2 m\_col\_text

```
Gtk::TreeModelColumn<Glib::ustring> ModelCategoryColumns::m_col_text
```

Definition at line 66 of file kitlistgui.hpp.

Referenced by ModelCategoryColumns(), ModellItemColumns::ModellItemColumns(), and KitListGui::refresh\_category\_list().

The documentation for this class was generated from the following file:

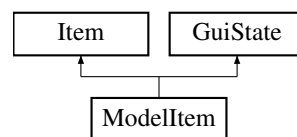
- </home/frank/Projects/kitlist/src/kitlistgui.hpp>

## 7.18 ModellItem Class Reference

Represents an [Item](#) combined with [GuiState](#) attributes.

```
#include <kitmodel.hpp>
```

Inheritance diagram for ModellItem:



### Public Member Functions

- [ModellItem](#) ()
- virtual void [set\\_checked](#) (bool checked)

### Friends

- class [ModellItemCompareId](#)

## Additional Inherited Members

### 7.18.1 Detailed Description

Represents an [Item](#) combined with [GuiState](#) attributes.

Definition at line 60 of file `kitmodel.hpp`.

### 7.18.2 Constructor & Destructor Documentation

#### 7.18.2.1 ModelItem()

```
ModelItem::ModelItem ( ) [inline]
```

Definition at line 62 of file `kitmodel.hpp`.

### 7.18.3 Member Function Documentation

#### 7.18.3.1 set\_checked()

```
virtual void ModelItem::set_checked (
    bool checked ) [inline], [virtual]
```

Reimplemented from [Item](#).

Definition at line 64 of file `kitmodel.hpp`.

References [Item::set\\_checked\(\)](#), and [GuiState::set\\_dirty\(\)](#).

Referenced by [KitParser::process\\_item\(\)](#), and [Service::update\\_item\(\)](#).

### 7.18.4 Friends And Related Function Documentation

#### 7.18.4.1 ModelItemCompareId

```
friend class ModelItemCompareId [friend]
```

Definition at line 63 of file `kitmodel.hpp`.

The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/kitmodel.hpp](#)

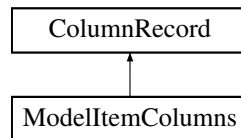


## 7.19 ModelItemColumns Class Reference

A definition for displaying an item in a multi-column list.

```
#include <kitlistgui.hpp>
```

Inheritance diagram for ModelItemColumns:



### Public Member Functions

- [ModelItemColumns \(\)](#)

### Public Attributes

- [Gtk::TreeModelColumn< Glib::ustring > m\\_col\\_text](#)
- [Gtk::TreeModelColumn< bool > m\\_col\\_checked](#)
- [Gtk::TreeModelColumn< int > m\\_col\\_num](#)

### 7.19.1 Detailed Description

A definition for displaying an item in a multi-column list.

Definition at line 77 of file kitlistgui.hpp.

### 7.19.2 Constructor & Destructor Documentation

#### 7.19.2.1 ModelItemColumns()

```
ModelItemColumns::ModelItemColumns ( ) [inline]
```

Definition at line 80 of file kitlistgui.hpp.

References [ModelCategoryColumns::m\\_col\\_num](#), and [ModelCategoryColumns::m\\_col\\_text](#).

### 7.19.3 Member Data Documentation

### 7.19.3.1 m\_col\_checked

```
Gtk::TreeModelColumn<bool> ModelItemColumns::m_col_checked
```

Definition at line 87 of file kitlistgui.hpp.

Referenced by KitListGui::init(), KitListGui::on\_row\_changed(), and KitListGui::refresh\_item\_list().

### 7.19.3.2 m\_col\_num

```
Gtk::TreeModelColumn<int> ModelItemColumns::m_col_num
```

Definition at line 88 of file kitlistgui.hpp.

Referenced by KitListGui::delete\_selected\_items(), KitListGui::get\_selected\_items(), KitListGui::init(), KitListGui::on\_row\_changed(), KitListGui::refresh\_item\_list(), and KitListGui::selected\_row\_callback().

### 7.19.3.3 m\_col\_text

```
Gtk::TreeModelColumn<Glib::ustring> ModelItemColumns::m_col_text
```

Definition at line 86 of file kitlistgui.hpp.

Referenced by KitListGui::init(), KitListGui::on\_row\_changed(), and KitListGui::refresh\_item\_list().

The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/kitlistgui.hpp](#)

## 7.20 ModellItemCompareId Class Reference

Comparator for comparing items by their unique ID.

```
#include <kitmodel.hpp>
```

### Public Member Functions

- [ModellItemCompareId\(\)](#)
- [int operator\(\) \(ModellItem \\*i1, ModellItem \\*i2\)](#)

### 7.20.1 Detailed Description

Comparator for comparing items by their unique ID.

See also

[Item](#)

Definition at line 72 of file kitmodel.hpp.

### 7.20.2 Constructor & Destructor Documentation

#### 7.20.2.1 ModelItemCompareId()

```
ModelItemCompareId::ModelItemCompareId ( ) [inline]
```

Definition at line 74 of file kitmodel.hpp.

### 7.20.3 Member Function Documentation

#### 7.20.3.1 operator()

```
int ModelItemCompareId::operator() (
    ModelItem * i1,
    ModelItem * i2 ) [inline]
```

Definition at line 75 of file kitmodel.hpp.

References [Item::get\\_id\(\)](#).

The documentation for this class was generated from the following file:

- [/home/frank/Projects/kitlist/src/kitmodel.hpp](#)

## 7.21 Service Class Reference

Business/service layer implementation.

```
#include <service.hpp>
```

## Public Member Functions

- [Service](#) ([KitListDao](#) &dao)
  - Loads the data model from the persistence store.*
- [~Service](#) ()
- [ModellItem](#) \* [find\\_item](#) (long id)
- [ModelCategory](#) \* [find\\_category](#) (long cat\_id)
- void [copy\\_items](#) (const [ModellItemContainer](#) &items, long cat\_id)
  - Copies items to the specified category.*
- [Item](#) \* [create\\_item](#) (long cat\_id)
  - Creates a new [ModellItem](#) and associates it with the specified category.*
- bool [delete\\_item](#) (long id)
  - Flags an item as deleted.*
- bool [delete\\_category](#) (long cat\_id)
  - Flags a category as deleted.*
- [Category](#) \* [create\\_category](#) ()
  - Creates a new category.*
- bool [is\\_model\\_dirty](#) ()
- void [set\\_model\\_dirty](#) (bool flag=true)
- virtual bool [filter](#) (bool checked)
  - Applies the current filter.*
- void [create\\_default\\_model](#) ()
  - Creates a default model.*
- void [open\\_as\\_xml](#) (const Glib::ustring &filename)
  - Loads a new data model from the named XML document.*
- void [save](#) ()
- void [save\\_as\\_xml](#) (const Glib::ustring &filename)
  - Saves the model's state to an XML document.*
- bool [update\\_item](#) (long id, const std::string description, bool checked)
  - Updates the attributes of an item.*
- [ItemContainer](#) \* [get\\_items](#) (long cat\_id=-1)
  - Returns a list of items.*
- [ItemContainer](#) \* [get\\_filtered\\_items](#) (long cat\_id=-1)
  - Returns a list of items, applying the current filter.*
- [CategoryContainer](#) \* [get\\_categories](#) ()
  - Returns a list of all categories.*
- virtual void [show\\_all](#) ()
  - Removes filter. All items are shown.*
- virtual void [show\\_checked\\_only](#) ()
  - Sets the filter to show only checked items.*
- virtual void [show\\_unchecked\\_only](#) ()
  - Sets the filter to show only unchecked items.*
- virtual void [select\\_items](#) ([ModellItemContainer](#) \*items, bool checked=true)
  - Checks or unchecks all the passed items.*
- virtual void [toggle\\_selected\\_items](#) ([ModellItemContainer](#) \*items)
  - Toggles the checked state of all the passed items.*
- virtual bool [require\\_filename](#) ()

## Protected Member Functions

- `KitModel * load_model ()`  
*Loads the data model from the persistence store.*
- `long get_next_item_id ()`
- `long get_next_category_id ()`

## Protected Attributes

- `KitListDao & m_dao`  
*Reference to the persistence data access object.*
- `KitModel * m_model`  
*The application's data model.*

### 7.21.1 Detailed Description

Business/service layer implementation.

Implements the service layer of the application, such that a different front-end can re-use the provided business methods.

Definition at line 38 of file `service.hpp`.

### 7.21.2 Constructor & Destructor Documentation

#### 7.21.2.1 Service()

```
Service::Service (  
    KitListDao & dao )
```

Loads the data model from the persistence store.

Definition at line 44 of file `service.cpp`.

References `load_model()`, and `m_model`.

#### 7.21.2.2 ~Service()

```
Service::~Service ( )
```

Definition at line 49 of file `service.cpp`.

References `m_model`.

### 7.21.3 Member Function Documentation

#### 7.21.3.1 copy\_items()

```
void Service::copy_items (
    const ModelItemContainer & items,
    long cat_id )
```

Copies items to the specified category.

Only copies those items that do not already exist in the target category.

Definition at line 144 of file service.cpp.

References `KitModel::copy_items()`, `m_model`, and `KitModel::set_dirty()`.

Referenced by `KitListGui::add_items()`.

#### 7.21.3.2 create\_category()

```
Category * Service::create_category ( )
```

Creates a new category.

A new `ModelCategory` is created, assigned a new unique ID and added to the model.

Definition at line 220 of file service.cpp.

References `KitModel::add_category()`, `get_next_category_id()`, `m_model`, `GuiState::set_dirty()`, `KitModel::set_dirty()`, `Category::set_id()`, and `GuiState::set_new_flag()`.

Referenced by `KitListGui::close_add_category_window()`.

#### 7.21.3.3 create\_default\_model()

```
void Service::create_default_model ( )
```

Creates a default model.

By default, a new empty `XmlDao` data model is created.

Definition at line 83 of file service.cpp.

References `KitListDao::get_model()`, `m_dao`, `m_model`, and `KitModel::reset()`.

Referenced by `filter()`, and `KitListGui::on_menu_file_new()`.

#### 7.21.3.4 create\_item()

```
Item * Service::create_item (
    long cat_id )
```

Creates a new [ModellItem](#) and associates it with the specified category.

A new item is created and assigned a new unique Id. The item is added to the data model and associated with a category if the `cat_id` is greater than or equal to zero. Otherwise the item is added to the model without being associated with a category.

Definition at line 159 of file `service.cpp`.

References `KitModel::add_item()`, `get_next_item_id()`, `m_model`, `GuiState::set_dirty()`, `KitModel::set_dirty()`, `Item::set_id()`, and `GuiState::set_new_flag()`.

Referenced by `KitListGui::close_add_item_window()`.

#### 7.21.3.5 delete\_category()

```
bool Service::delete_category (
    long cat_id )
```

Flags a category as deleted.

Finds the category with the specified ID and flags it as deleted.

##### Returns

false if the category does not exist.

Definition at line 200 of file `service.cpp`.

References `KitModel::find_category()`, `m_model`, `GuiState::set_deleted()`, `GuiState::set_dirty()`, and `KitModel::set_dirty()`.

Referenced by `KitListGui::on_menu_delete_category()`.

#### 7.21.3.6 delete\_item()

```
bool Service::delete_item (
    long id )
```

Flags an item as deleted.

Finds the item with the specified ID and flags it as deleted.

##### Returns

false if the item does not exist.

Definition at line 180 of file `service.cpp`.

References `KitModel::find_item()`, `m_model`, `GuiState::set_deleted()`, `GuiState::set_dirty()`, and `KitModel::set_dirty()`.

Referenced by `KitListGui::delete_selected_items()`, and `KitListGui::selected_row_callback()`.

### 7.21.3.7 filter()

```
virtual bool Service::filter (  
    bool checked ) [inline], [virtual]
```

Applies the current filter.



**Parameters**

<i>checked</i>	The checked/ticked state of the item being filtered.
----------------	--

**Returns**

true if the item should be included.

Definition at line 63 of file service.hpp.

References `create_default_model()`, `KitModel::filter()`, `get_categories()`, `get_filtered_items()`, `get_items()`, `open_as_xml()`, `save()`, `save_as_xml()`, and `update_item()`.

Referenced by `KitListGui::refresh_item_list()`.

**7.21.3.8 find\_category()**

```
ModelCategory * Service::find_category (
    long cat_id )
```

Returns the category with the specified unique ID.

Definition at line 133 of file service.cpp.

References `KitModel::find_category()`, and `m_model`.

Referenced by `KitListGui::close_add_category_window()`, `KitListGui::on_menu_cut()`, and `KitListGui::on_menu_rename_category()`.

**7.21.3.9 find\_item()**

```
ModelItem * Service::find_item (
    long id )
```

Returns the item with the specified unique ID.

Definition at line 125 of file service.cpp.

References `KitModel::find_item()`, and `m_model`.

Referenced by `KitListGui::get_selected_items()`, and `KitListGui::paste_from_xml()`.

### 7.21.3.10 get\_categories()

```
CategoryContainer * Service::get_categories ( )
```

Returns a list of all categories.

Categories flagged as deleted are excluded.

Definition at line 354 of file service.cpp.

References KitModel::get\_categories(), and m\_model.

Referenced by filter(), and KitListGui::refresh\_category\_list().

### 7.21.3.11 get\_filtered\_items()

```
ItemContainer * Service::get_filtered_items (
    long cat_id = -1 )
```

Returns a list of items, applying the current filter.

The returned list is also sorted by item name.

If cat\_id is less than zero, returns all items, otherwise only those items associated with the specified category ID.

#### Parameters

<i>cat</i> ↔ <i>_id</i>	the unique ID of a category or '-1' to indicate all items are to be retrieved.
----------------------------	--

Definition at line 331 of file service.cpp.

References KitModel::find\_category(), KitModel::get\_all\_items(), ModelCategory::get\_items(), and m\_model.

Referenced by filter(), KitListGui::on\_menu\_export\_to\_pdf(), and KitListGui::on\_menu\_print().

### 7.21.3.12 get\_items()

```
ItemContainer * Service::get_items (
    long cat_id = -1 )
```

Returns a list of items.

If cat\_id is less than zero, returns all items, otherwise only those items associated with the specified category ID.

## Parameters

<code>cat↔ _id</code>	the unique ID of a category or '-1' to indicate all items are to be retrieved.
---------------------------	--

Definition at line 305 of file service.cpp.

References `KitModel::find_category()`, `KitModel::get_all_items()`, `ModelCategory::get_items()`, and `m_model`.

Referenced by `filter()`, `KitListGui::init()`, and `KitListGui::refresh_item_list()`.

### 7.21.3.13 `get_next_category_id()`

```
long Service::get_next_category_id ( ) [protected]
```

Returns an unused unique id for a [Category](#).

Definition at line 372 of file service.cpp.

References `KitListDao::get_next_category_id()`, and `m_dao`.

Referenced by `create_category()`.

### 7.21.3.14 `get_next_item_id()`

```
long Service::get_next_item_id ( ) [protected]
```

Returns an unused unique id for an [Item](#).

Definition at line 364 of file service.cpp.

References `KitListDao::get_next_item_id()`, and `m_dao`.

Referenced by `create_item()`.

### 7.21.3.15 `is_model_dirty()`

```
bool Service::is_model_dirty ( ) [inline]
```

Definition at line 55 of file service.hpp.

References `KitModel::is_dirty()`, and `set_model_dirty()`.

Referenced by `KitListGui::on_delete_event()`, `KitListGui::on_menu_file_new()`, `KitListGui::on_menu_file_open()`, `KitListGui::on_menu_quit()`, `KitListGui::on_menu_recent_file()`, `KitListGui::on_menu_save()`, `KitListGui::run()`, and `KitListGui::safe_open_file()`.

#### 7.21.3.16 load\_model()

```
KitModel * Service::load_model ( ) [protected]
```

Loads the data model from the persistence store.

Definition at line 58 of file service.cpp.

References KitListDao::get\_model(), and m\_dao.

Referenced by Service().

#### 7.21.3.17 open\_as\_xml()

```
void Service::open_as_xml (
    const Glib::ustring & filename )
```

Loads a new data model from the named XML document.

Creates a new data model based on the passed filename. The existing data model is destroyed after successfully loading the new one.

Definition at line 70 of file service.cpp.

References m\_dao, and m\_model.

Referenced by filter(), KitListGui::init(), KitListGui::on\_menu\_recent\_file(), and KitListGui::open\_file().

#### 7.21.3.18 require\_filename()

```
virtual bool Service::require_filename ( ) [inline], [virtual]
```

Definition at line 80 of file service.hpp.

References KitListDao::require\_filename().

Referenced by KitListGui::init(), KitListGui::on\_menu\_file\_new(), and KitListGui::on\_menu\_save().

#### 7.21.3.19 save()

```
void Service::save ( )
```

Saves the model's state to the persistence store.

Definition at line 98 of file service.cpp.

References m\_dao, m\_model, and KitListDao::save\_model().

Referenced by KitListGui::confirm\_lose\_changes(), filter(), and KitListGui::on\_menu\_save().

#### 7.21.3.20 save\_as\_xml()

```
void Service::save_as_xml (
    const Glib::ustring & filename )
```

Saves the model's state to an XML document.

This is primarily intended to support switching from one dao implementation to the [XmlDao](#) implementation.

**Parameters**

<i>filename</i>	The full pathname and filename to save the file to.
-----------------	---

Definition at line 111 of file service.cpp.

References `m_dao`, `m_model`, and `XmlDao::save_model()`.

Referenced by `KitListGui::confirm_lose_changes()`, `filter()`, `KitListGui::on_menu_save()`, and `KitListGui::on_menu_save_as()`.

**7.21.3.21 select\_items()**

```
void Service::select_items (
    ModelItemContainer * items,
    bool checked = true ) [virtual]
```

Checks or unchecks all the passed items.

**Parameters**

<i>items</i>	The items to change. the state to change the to.
--------------	--

Definition at line 238 of file service.cpp.

References `m_model`, and `KitModel::set_dirty()`.

Referenced by `KitListGui::set_selected()`, and `show_unchecked_only()`.

**7.21.3.22 set\_model\_dirty()**

```
void Service::set_model_dirty (
    bool flag = true )
```

Flags the model as being dirty.

Definition at line 264 of file service.cpp.

References `m_model`, and `KitModel::set_dirty()`.

Referenced by `KitListGui::close_add_category_window()`, `KitListGui::confirm_lose_changes()`, `KitListGui::init()`, `is_model_dirty()`, `KitListGui::on_cell_edit()`, `KitListGui::on_delete_event()`, `KitListGui::on_menu_cut()`, `KitListGui::on_menu_quit()`, and `KitListGui::on_row_changed()`.

### 7.21.3.23 show\_all()

```
virtual void Service::show_all ( ) [inline], [virtual]
```

Removes filter. All items are shown.

Definition at line 73 of file service.hpp.

References `KitModel::show_all()`.

Referenced by `KitListGui::on_menu_show_all()`.

### 7.21.3.24 show\_checked\_only()

```
virtual void Service::show_checked_only ( ) [inline], [virtual]
```

Sets the filter to show only checked items.

Definition at line 75 of file service.hpp.

References `KitModel::show_checked_only()`.

Referenced by `KitListGui::on_menu_show_checked()`.

### 7.21.3.25 show\_unchecked\_only()

```
virtual void Service::show_unchecked_only ( ) [inline], [virtual]
```

Sets the filter to show only unchecked items.

Definition at line 77 of file service.hpp.

References `select_items()`, `KitModel::show_unchecked_only()`, and `toggle_selected_items()`.

Referenced by `KitListGui::on_menu_show_unchecked()`.

### 7.21.3.26 toggle\_selected\_items()

```
void Service::toggle_selected_items (
    ModelItemContainer * items ) [virtual]
```

Toggles the checked state of all the passed items.

**Parameters**

<i>items</i>	The items to change.
--------------	----------------------

Definition at line 252 of file service.cpp.

References `m_model`, and `KitModel::set_dirty()`.

Referenced by `show_unchecked_only()`, and `KitListGui::toggle_selected()`.

**7.21.3.27 update\_item()**

```
bool Service::update_item (
    long id,
    const std::string description,
    bool checked )
```

Updates the attributes of an item.

Locates the item using the supplied unique ID, then assigns the passed values.

**Parameters**

<i>id</i>	The unique ID of the item to update.
<i>description</i>	The item's descriptive text.
<i>checked</i>	The checked/ticked state of the item.

**Returns**

Returns true if the item exists, false otherwise.

Definition at line 282 of file service.cpp.

References `KitModel::find_item()`, `m_model`, `ModelItem::set_checked()`, `Item::set_description()`, and `GuiState::set_dirty()`.

Referenced by `filter()`, and `KitListGui::on_row_changed()`.

**7.21.4 Member Data Documentation****7.21.4.1 m\_dao**

```
KitListDao& Service::m_dao [protected]
```

Reference to the persistence data access object.

Definition at line 40 of file service.hpp.

Referenced by `create_default_model()`, `get_next_category_id()`, `get_next_item_id()`, `load_model()`, `open_as_xml()`, `save()`, and `save_as_xml()`.

#### 7.21.4.2 m\_model

```
KitModel* Service::m_model [protected]
```

The application's data model.

Definition at line 41 of file service.hpp.

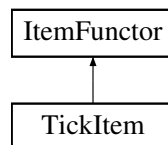
Referenced by `copy_items()`, `create_category()`, `create_default_model()`, `create_item()`, `delete_category()`, `delete_item()`, `find_category()`, `find_item()`, `get_categories()`, `get_filtered_items()`, `get_items()`, `open_as_xml()`, `save()`, `save_as_xml()`, `select_items()`, `Service()`, `set_model_dirty()`, `toggle_selected_items()`, `update_item()`, and `~Service()`.

The documentation for this class was generated from the following files:

- [/home/frank/Projects/kitlist/src/service.hpp](#)
- [/home/frank/Projects/kitlist/src/service.cpp](#)

## 7.22 TickItem Class Reference

Inheritance diagram for TickItem:



### Public Member Functions

- [TickItem](#) ([ItemContainer](#) &changed)
- [bool operator\(\)](#) ([Item](#) &item)

### Private Attributes

- [ItemContainer](#) & [m\\_changed](#)

#### 7.22.1 Detailed Description

Definition at line 114 of file main.cpp.

#### 7.22.2 Constructor & Destructor Documentation



### 7.22.2.1 TickItem()

```
TickItem::TickItem (
    ItemContainer & changed ) [inline]
```

Definition at line 117 of file main.cpp.

## 7.22.3 Member Function Documentation

### 7.22.3.1 operator>()

```
bool TickItem::operator() (
    Item & item ) [inline], [virtual]
```

Implements [ItemFunctor](#).

Definition at line 118 of file main.cpp.

References [Item::get\\_checked\(\)](#), [Item::get\\_description\(\)](#), [Item::get\\_id\(\)](#), and [Item::set\\_checked\(\)](#).

## 7.22.4 Member Data Documentation

### 7.22.4.1 m\_changed

```
ItemContainer& TickItem::m_changed [private]
```

Definition at line 115 of file main.cpp.

The documentation for this class was generated from the following file:

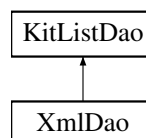
- </home/frank/Projects/kitlist/src/main.cpp>

## 7.23 XmlDao Class Reference

Implementation of a [KitListDao](#) using XML as the persistence store.

```
#include <xmldao.hpp>
```

Inheritance diagram for XmlDao:



## Public Member Functions

- [XmlDao](#) (int verbose=0)
- [KitModel](#) \* [get\\_model](#) ()  
*Loads the data model from the previously set filename.*
- [KitModel](#) \* [get\\_model](#) (Glib::ustring filename)
- void [save\\_model](#) ([KitModel](#) \*model)  
*Saves the model as an XML document.*
- void [save\\_model](#) ([KitModel](#) \*model, Glib::ustring filename)  
*Saves the model as an XML document.*
- [Category](#) \* [get\\_category](#) (long cat\_id, [item\\_choice](#) choice)  
*Loads a category.*
- [ItemContainer](#) \* [get\\_all\\_items](#) ([item\\_choice](#) choice)  
*Returns a list of all items.*
- long [add\\_item](#) (const std::string name)
- long [add\\_item](#) (const std::string name, long cat\_id)
- void [append\\_items\\_to\\_category](#) (long to\_cat\_id, long from\_cat\_id, [item\\_choice](#) choice)  
*Copies items from one category to another.*
- void [associate\\_item\\_with\\_category](#) (long id, long cat\_id)  
*Associates an existing item with an existing category.*
- [CategoryContainer](#) [get\\_categories](#) ()
- long [new\\_category](#) (const std::string name)  
*Creates a new category.*
- void [delete\\_item](#) (long id)
- void [update\\_item\\_checked\\_state](#) ([ItemContainer](#) &items)  
*Persists the state of the 'checked' flag of each item.*
- void [remove\\_item\\_from\\_category](#) (long id, long cat\_id)
- long [get\\_next\\_item\\_id](#) ()  
*Returns the next unused unique id for items.*
- long [get\\_next\\_category\\_id](#) ()
- void [delete\\_category](#) (long id)
- void [set\\_item\\_flag](#) (long id)
- void [unset\\_item\\_flag](#) (long id)
- void [set\\_category\\_flag](#) (long id)
- void [unset\\_category\\_flag](#) (long id)
- void [set\\_all\\_flags](#) ()
- void [unset\\_all\\_flags](#) ()
- void [set\\_filename](#) (Glib::ustring filename)
- virtual bool [require\\_filename](#) ()  
*Indicates that this implementation requires a filename.*

## Protected Attributes

- Glib::ustring [m\\_filename](#)  
*The filename to load or save the XML document from.*
- xmlpp::Element \* [m\\_items\\_node](#)  
*Temporary reference to the items' node.*
- xmlpp::Element \* [m\\_categories\\_node](#)  
*Temporary reference to the categories' node.*
- xmlpp::Element \* [m\\_cat\\_items\\_node](#)  
*Temporary reference to the categories' items' node.*
- long [m\\_max\\_item\\_id](#)  
*The last used ID for items.*
- long [m\\_max\\_category\\_id](#)  
*The last used ID for categories.*

## Private Member Functions

- bool [add\\_item\\_to\\_dom](#) ([ModelItem](#) &item)  
*Adds the passed item to the current items' node.*
- bool [add\\_category\\_item\\_to\\_dom](#) ([Item](#) &item)  
*Adds the passed item to the current category's node.*
- bool [add\\_category\\_to\\_dom](#) ([ModelCategory](#) &item)  
*Adds the passed item to the current categories' node.*

### 7.23.1 Detailed Description

Implementation of a [KitListDao](#) using XML as the persistence store.

Definition at line 42 of file `xml dao.hpp`.

### 7.23.2 Constructor & Destructor Documentation

#### 7.23.2.1 XmlDao()

```
XmlDao::XmlDao (  
    int verbose = 0 ) [inline]
```

Definition at line 67 of file `xml dao.hpp`.

References [get\\_model\(\)](#).

### 7.23.3 Member Function Documentation

#### 7.23.3.1 add\_category\_item\_to\_dom()

```
bool XmlDao::add_category_item_to_dom (  
    Item & item ) [private]
```

Adds the passed item to the current category's node.

#### See also

[XmlDao::save\\_model\(KitModel&\)](#)

Definition at line 106 of file `xml dao.cpp`.

References [Item::get\\_id\(\)](#).

Referenced by [add\\_category\\_to\\_dom\(\)](#).

### 7.23.3.2 add\_category\_to\_dom()

```
bool XmlDao::add_category_to_dom (
    ModelCategory & category ) [private]
```

Adds the passed item to the current categories' node.

#### See also

`XmlDao::save_model(KitModel&)`

Definition at line 123 of file `xmldao.cpp`.

References `add_category_item_to_dom()`, `Category::foreach_item()`, `Category::get_id()`, `Category::get_name()`, and `GuiState::is_deleted()`.

Referenced by `save_model()`.

### 7.23.3.3 add\_item() [1/2]

```
long XmlDao::add_item (
    const std::string name ) [inline], [virtual]
```

Creates a new item.

#### Parameters

<i>name</i>	The name of the new item.
-------------	---------------------------

Implements [KitListDao](#).

Definition at line 92 of file `xmldao.hpp`.

References NYI.

### 7.23.3.4 add\_item() [2/2]

```
long XmlDao::add_item (
    const std::string name,
    long cat_id ) [inline], [virtual]
```

Creates a new item and associates it with a category.

#### Parameters

<i>name</i>	The name of the new item.
-------------	---------------------------

Implements [KitListDao](#).

Definition at line 94 of file `xml dao.hpp`.

References NYI.

### 7.23.3.5 `add_item_to_dom()`

```
bool XmlDao::add_item_to_dom (
    ModelItem & item ) [private]
```

Adds the passed item to the current items' node.

See also

`XmlDao::save_model(KitModel&)`

Definition at line 87 of file `xml dao.cpp`.

References `Item::get_checked()`, `Item::get_description()`, `Item::get_id()`, and `GuiState::is_deleted()`.

Referenced by `save_model()`.

### 7.23.3.6 `append_items_to_category()`

```
void XmlDao::append_items_to_category (
    long to_cat_id,
    long from_cat_id,
    item_choice choice ) [inline], [virtual]
```

Copies items from one category to another.

Optionally, only checked or unchecked items are copied.

Parameters

<i>from_cat_id</i>	The ID of the source category.
<i>to_cat_id</i>	The ID of the target category.
<i>choice</i>	One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Implements [KitListDao](#).

Definition at line 96 of file `xml dao.hpp`.

References NYI.

### 7.23.3.7 associate\_item\_with\_category()

```
void XmlDao::associate_item_with_category (
    long id,
    long cat_id ) [inline], [virtual]
```

Associates an existing item with an existing category.

#### Parameters

<i>id</i>	The <a href="#">Item</a> ID
<i>cat↔ _id</i>	The <a href="#">Category</a> ID

Implements [KitListDao](#).

Definition at line 98 of file `xmldao.hpp`.

References NYI.

### 7.23.3.8 delete\_category()

```
void XmlDao::delete_category (
    long id ) [inline], [virtual]
```

Deletes a category.

Implements [KitListDao](#).

Definition at line 114 of file `xmldao.hpp`.

References NYI.

### 7.23.3.9 delete\_item()

```
void XmlDao::delete_item (
    long id ) [inline], [virtual]
```

Deletes an item by it's ID.

#### Parameters

<i>id</i>	the ID of the item to delete.
-----------	-------------------------------

Implements [KitListDao](#).

Definition at line 104 of file `xmldao.hpp`.

References NYI.

#### 7.23.3.10 `get_all_items()`

```
ItemContainer* XmlDao::get_all_items (
    item_choice choice ) [inline], [virtual]
```

Returns a list of all items.

##### Parameters

<i>choice</i>	Which items to load. One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.
---------------	--

Implements [KitListDao](#).

Definition at line 90 of file `xml dao.hpp`.

References NYI.

#### 7.23.3.11 `get_categories()`

```
CategoryContainer XmlDao::get_categories ( ) [inline], [virtual]
```

Returns a list of all categories.

Implements [KitListDao](#).

Definition at line 100 of file `xml dao.hpp`.

References NYI.

#### 7.23.3.12 `get_category()`

```
Category* XmlDao::get_category (
    long cat_id,
    item_choice choice ) [inline], [virtual]
```

Loads a category.

##### Parameters

<i>choice</i>	Which items to load for the category. One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.
---------------	---

Implements [KitListDao](#).

Definition at line 88 of file xmldao.hpp.

References NYI.

### 7.23.3.13 `get_model()` [1/2]

```
KitModel * XmlDao::get_model ( ) [virtual]
```

Loads the data model from the previously set filename.

The data model holds a rich graph of objects, representing the entire list of categories and items.

The filename must be set before calling this method by calling `XmlDao::set_filename()`, otherwise an empty model is returned.

#### Return values

<i>Returns</i>	a newly created data model. The caller is responsible for destroying the model.
----------------	---

#### See also

[KitModel](#)

Implements [KitListDao](#).

Definition at line 46 of file xmldao.cpp.

References `KitModel::get_all_items()`, `KitModel::get_categories()`, `Category::get_id()`, `Item::get_id()`, and `KitModel::reset()`.

Referenced by `XmlDao()`.

### 7.23.3.14 `get_model()` [2/2]

```
KitModel* XmlDao::get_model (
    Glib::ustring filename ) [inline]
```

Definition at line 76 of file xmldao.hpp.

References `get_model()`, `save_model()`, and `set_filename()`.

Referenced by `get_model()`.



#### 7.23.3.15 `get_next_category_id()`

```
long XmlDao::get_next_category_id ( ) [virtual]
```

Returns the next unused unique id for categories.

Implements [KitListDao](#).

Definition at line 150 of file `xmldao.cpp`.

Referenced by `remove_item_from_category()`.

#### 7.23.3.16 `get_next_item_id()`

```
long XmlDao::get_next_item_id ( ) [virtual]
```

Returns the next unused unique id for items.

Implements [KitListDao](#).

Definition at line 142 of file `xmldao.cpp`.

Referenced by `remove_item_from_category()`.

#### 7.23.3.17 `new_category()`

```
long XmlDao::new_category (
    const std::string name ) [inline], [virtual]
```

Creates a new category.

##### Parameters

<i>name</i>	the name of the new category.
-------------	-------------------------------

Implements [KitListDao](#).

Definition at line 102 of file `xmldao.hpp`.

References NYI.

#### 7.23.3.18 `remove_item_from_category()`

```
void XmlDao::remove_item_from_category (
    long id,
    long cat_id ) [inline], [virtual]
```

Un-associates the specified item from the specified category.

## Parameters

<i>id</i>	The <a href="#">Item</a> id.
<i>cat↔ _id</i>	The <a href="#">Category</a> id.

Implements [KitListDao](#).

Definition at line 108 of file `xmldao.hpp`.

References `get_next_category_id()`, `get_next_item_id()`, and `NYI`.

7.23.3.19 `require_filename()`

```
virtual bool XmlDao::require_filename ( ) [inline], [virtual]
```

Indicates that this implementation requires a filename.

This persistence model requires a filename to be chosen before the data can be persisted.

## Returns

Always returns true.

Reimplemented from [KitListDao](#).

Definition at line 138 of file `xmldao.hpp`.

7.23.3.20 `save_model()` [1/2]

```
void XmlDao::save_model (
    KitModel * model ) [virtual]
```

Saves the model as an XML document.

The filename must be set before calling this method by calling `XmlDao::set_filename()`;

## Parameters

<i>model</i>	The model to save.
--------------	--------------------

Implements [KitListDao](#).

Definition at line 163 of file `xmldao.cpp`.

References `add_category_to_dom()`, `add_item_to_dom()`, `KitModel::foreach_category()`, `KitModel::foreach_item()`, `KitModel::purge()`, and `KitModel::reset()`.

Referenced by `get_model()`, and `Service::save_as_xml()`.

#### 7.23.3.21 `save_model()` [2/2]

```
void XmlDao::save_model (
    KitModel * model,
    Glib::ustring filename ) [inline]
```

Saves the model as an XML document.

##### Parameters

<i>model</i>	The model to save.
<i>filename</i>	The name of the file to save to.

Definition at line 86 of file `xml dao.hpp`.

References `save_model()`, and `set_filename()`.

Referenced by `save_model()`.

#### 7.23.3.22 `set_all_flags()`

```
void XmlDao::set_all_flags ( ) [inline], [virtual]
```

Checks/ticks all items.

Implements [KitListDao](#).

Definition at line 124 of file `xml dao.hpp`.

References NYI.

#### 7.23.3.23 `set_category_flag()`

```
void XmlDao::set_category_flag (
    long id ) [inline], [virtual]
```

Checks/ticks all items in the specified [Category](#).

Implements [KitListDao](#).

Definition at line 120 of file `xml dao.hpp`.

References NYI.

#### 7.23.3.24 set\_filename()

```
void XmlDao::set_filename (
    Glib::ustring filename ) [inline]
```

Definition at line 128 of file xmldao.hpp.

Referenced by `get_model()`, and `save_model()`.

#### 7.23.3.25 set\_item\_flag()

```
void XmlDao::set_item_flag (
    long id ) [inline], [virtual]
```

Sets the checked flag for an item.

##### Parameters

<i>id</i>	The id of the item to change.
-----------	-------------------------------

Implements [KitListDao](#).

Definition at line 116 of file xmldao.hpp.

References NYI.

#### 7.23.3.26 unset\_all\_flags()

```
void XmlDao::unset_all_flags ( ) [inline], [virtual]
```

Unchecks/unticks all items.

Implements [KitListDao](#).

Definition at line 126 of file xmldao.hpp.

References NYI.

#### 7.23.3.27 unset\_category\_flag()

```
void XmlDao::unset_category_flag (
    long id ) [inline], [virtual]
```

Unchecks/unticks all items in the specified [Category](#).

Implements [KitListDao](#).

Definition at line 122 of file xmldao.hpp.

References NYI.

### 7.23.3.28 unset\_item\_flag()

```
void XmlDao::unset_item_flag (
    long id ) [inline], [virtual]
```

Clears the checked flag for an item.

Implements [KitListDao](#).

Definition at line 118 of file xmldao.hpp.

References NYI.

### 7.23.3.29 update\_item\_checked\_state()

```
void XmlDao::update_item_checked_state (
    ItemContainer & items ) [inline], [virtual]
```

Persists the state of the 'checked' flag of each item.

Implements [KitListDao](#).

Definition at line 106 of file xmldao.hpp.

References NYI.

## 7.23.4 Member Data Documentation

### 7.23.4.1 m\_cat\_items\_node

```
xmlpp::Element* XmlDao::m_cat_items_node [protected]
```

Temporary reference to the categories' items' node.

Definition at line 60 of file xmldao.hpp.

### 7.23.4.2 m\_categories\_node

```
xmlpp::Element* XmlDao::m_categories_node [protected]
```

Temporary reference to the categories' node.

Definition at line 58 of file xmldao.hpp.

#### 7.23.4.3 m\_filename

```
Glib::ustring XmlDao::m_filename [protected]
```

The filename to load or save the XML document from.

Definition at line 54 of file xmldao.hpp.

#### 7.23.4.4 m\_items\_node

```
xmlpp::Element* XmlDao::m_items_node [protected]
```

Temporary reference to the items' node.

Definition at line 56 of file xmldao.hpp.

#### 7.23.4.5 m\_max\_category\_id

```
long XmlDao::m_max_category_id [protected]
```

The last used ID for categories.

Definition at line 64 of file xmldao.hpp.

#### 7.23.4.6 m\_max\_item\_id

```
long XmlDao::m_max_item_id [protected]
```

The last used ID for items.

Definition at line 62 of file xmldao.hpp.

The documentation for this class was generated from the following files:

- [/home/frank/Projects/kitlist/src/xmldao.hpp](#)
- [/home/frank/Projects/kitlist/src/xmldao.cpp](#)

## 7.24 YamlConfig Class Reference

Maintains the application's configuration parameters in a YAML formatted file.

```
#include <yamlconfig.hpp>
```

## Public Member Functions

- [YamlConfig](#) ()
- [~YamlConfig](#) ()
  - Also saves the configuration on program exit.*
- void [load](#) ()
  - Loads the configuration file.*
- void [save](#) ()
  - Saves the current configuration.*
- Glib::ustring [get\\_current\\_filename](#) ()
  - The current filename.*
- void [set\\_current\\_filename](#) (Glib::ustring filename)
  - Sets the current filename.*
- Glib::ustring [get\\_page\\_title](#) ()
  - The page title to use for printing or creating a PDF.*
- void [set\\_page\\_title](#) (Glib::ustring title)
  - Set the page title.*
- void [add\\_recent\\_filename](#) (Glib::ustring filename)
  - Adds a filename to the list of recently used filenames.*
- std::deque< Glib::ustring > [get\\_recent\\_filenames](#) ()
  - The history list of recent filenames.*
- Glib::ustring [get\\_debug\\_log\\_filename](#) ()
  - the name of the debug log file.*

## Private Member Functions

- Glib::ustring [get\\_config\\_filename](#) ()

## Private Attributes

- gint [m\\_max\\_recent\\_files](#)
  - The list of recently used files for the recent files menu.*
- Glib::ustring [m\\_current\\_filename](#)
  - The current filename.*
- Glib::ustring [m\\_debug\\_log\\_filename](#)
  - The name of the log file to write debug information to.*
- Glib::ustring [m\\_page\\_title](#)
  - The title to use when printing a page or creating a PDF.*
- std::deque< Glib::ustring > [m\\_mru\\_file\\_history](#)
  - The history list of most recently used filenames.*

### 7.24.1 Detailed Description

Maintains the application's configuration parameters in a YAML formatted file.

See <https://yaml.org/> for information on the YAML file format.

The application uses the C++ implementation of a YAML parser from <https://github.com/jbeder/yaml-cpp/>

The location of the configuration file follows the XDG Base Directory Specification. By default this is `~/.config/kitlist` but can be overridden by specifying the name of the configuration directory by defining the `XDG_CONFIG_HOME` environment variable.

See: <https://specifications.freedesktop.org/basedir-spec/basedir-spec-latest.html>

Definition at line 55 of file `yamlconfig.hpp`.

## 7.24.2 Constructor & Destructor Documentation

### 7.24.2.1 YamlConfig()

```
YamlConfig::YamlConfig ( )
```

Definition at line 51 of file yamlconfig.cpp.

References [load\(\)](#).

### 7.24.2.2 ~YamlConfig()

```
YamlConfig::~YamlConfig ( ) [inline]
```

Also saves the configuration on program exit.

Definition at line 85 of file yamlconfig.hpp.

References [load\(\)](#), and [save\(\)](#).

## 7.24.3 Member Function Documentation

### 7.24.3.1 add\_recent\_filename()

```
void YamlConfig::add_recent_filename (
    Glib::ustring filename )
```

Adds a filename to the list of recently used filenames.

#### Parameters

<i>filename</i>	the filename to set.
-----------------	----------------------

Definition at line 141 of file yamlconfig.cpp.

References [m\\_max\\_recent\\_files](#), and [m\\_mru\\_file\\_history](#).

Referenced by [set\\_page\\_title\(\)](#), and [KitListGui::update\\_recent\\_files\(\)](#).



### 7.24.3.2 get\_config\_filename()

```
Glib::ustring YamlConfig::get_config_filename ( ) [private]
```

Returns the path to the user's configuration file for this application. e.g. "/home/user/.config/kitlist"

Definition at line 129 of file yamlconfig.cpp.

References CONFIG\_FILENAME.

Referenced by load(), and save().

### 7.24.3.3 get\_current\_filename()

```
Glib::ustring YamlConfig::get_current_filename ( ) [inline]
```

The current filename.

#### Returns

the current filename

Definition at line 93 of file yamlconfig.hpp.

References m\_current\_filename.

Referenced by KitListGui::init().

### 7.24.3.4 get\_debug\_log\_filename()

```
Glib::ustring YamlConfig::get_debug_log_filename ( ) [inline]
```

the name of the debug log file.

#### Returns

the filename.

Definition at line 120 of file yamlconfig.hpp.

References m\_debug\_log\_filename.

Referenced by KitListGui::init().

#### 7.24.3.5 get\_page\_title()

```
Glib::ustring YamlConfig::get_page_title ( ) [inline]
```

The page title to use for printing or creating a PDF.

##### Returns

the page title.

Definition at line 102 of file yamlconfig.hpp.

References `m_page_title`.

Referenced by `KitListGui::init()`.

#### 7.24.3.6 get\_recent\_filenames()

```
std::deque<Glib::ustring> YamlConfig::get_recent_filenames ( ) [inline]
```

The history list of recent filenames.

##### Returns

a list of the most recently used filenames.

Definition at line 116 of file yamlconfig.hpp.

References `m_mru_file_history`.

Referenced by `KitListGui::update_recent_files_menu()`.

#### 7.24.3.7 load()

```
void YamlConfig::load ( )
```

Loads the configuration file.

Definition at line 60 of file yamlconfig.cpp.

References `CURRENT_FILENAME_CONFIG_KEY`, `DEBUG_LOG_FILENAME_CONFIG_KEY`, `DEFAULT_PAGE_TITLE`, `get_config_filename()`, `m_current_filename`, `m_debug_log_filename`, `m_max_recent_files`, `m_mru_file_history`, `m_page_title`, `MAX_RECENT_FILES_CONFIG_KEY`, `PAGE_TITLE_CONFIG_KEY`, and `RECENT_FILES_CONFIG_KEY`.

Referenced by `YamlConfig()`, and `~YamlConfig()`.

### 7.24.3.8 save()

```
void YamlConfig::save ( )
```

Saves the current configuration.

Definition at line 104 of file `yamlconfig.cpp`.

References `CURRENT_FILENAME_CONFIG_KEY`, `DEBUG_LOG_FILENAME_CONFIG_KEY`, `get_config_↔filename()`, `m_current_filename`, `m_debug_log_filename`, `m_max_recent_files`, `m_mru_file_history`, `m_page_title`, `MAX_RECENT_FILES_CONFIG_KEY`, `PAGE_TITLE_CONFIG_KEY`, and `RECENT_FILES_CONFIG_KEY`.

Referenced by `~YamlConfig()`.

### 7.24.3.9 set\_current\_filename()

```
void YamlConfig::set_current_filename (
    Glib::ustring filename ) [inline]
```

Sets the current filename.

#### Parameters

<i>filename</i>	the name of the file to set.
-----------------	------------------------------

Definition at line 98 of file `yamlconfig.hpp`.

Referenced by `KitListGui::on_menu_file_new()`, `KitListGui::on_menu_recent_file()`, `KitListGui::on_menu_save()`, `KitListGui::on_menu_save_as()`, and `KitListGui::open_file()`.

### 7.24.3.10 set\_page\_title()

```
void YamlConfig::set_page_title (
    Glib::ustring title ) [inline]
```

Set the page title.

Sets the page title used for printing or creating a PDF.

#### Parameters

<i>title</i>	the page title to set.
--------------	------------------------

Definition at line 109 of file `yamlconfig.hpp`.

References `add_recent_filename()`.

Referenced by `KitListGui::set_page_title()`.

## 7.24.4 Member Data Documentation

### 7.24.4.1 m\_current\_filename

```
Glib::ustring YamlConfig::m_current_filename [private]
```

The current filename.

Used to reload the same file on the next occasion the application is launched.

Definition at line 65 of file yamlconfig.hpp.

Referenced by `get_current_filename()`, `load()`, and `save()`.

### 7.24.4.2 m\_debug\_log\_filename

```
Glib::ustring YamlConfig::m_debug_log_filename [private]
```

The name of the log file to write debug information to.

Note: This is only used if the application is compiled with `KITLIST_DEBUG`. See the distribution README for more information.

Definition at line 72 of file yamlconfig.hpp.

Referenced by `get_debug_log_filename()`, `load()`, and `save()`.

### 7.24.4.3 m\_max\_recent\_files

```
gint YamlConfig::m_max_recent_files [private]
```

The list of recently used files for the recent files menu.

Definition at line 59 of file yamlconfig.hpp.

Referenced by `add_recent_filename()`, `load()`, and `save()`.

### 7.24.4.4 m\_mru\_file\_history

```
std::deque<Glib::ustring> YamlConfig::m_mru_file_history [private]
```

The history list of most recently used filenames.

Definition at line 80 of file yamlconfig.hpp.

Referenced by `add_recent_filename()`, `get_recent_filenames()`, `load()`, and `save()`.

#### 7.24.4.5 m\_page\_title

```
Glib::ustring YamlConfig::m_page_title [private]
```

The title to use when printing a page or creating a PDF.

Definition at line 75 of file `yamlconfig.hpp`.

Referenced by `get_page_title()`, `load()`, and `save()`.

The documentation for this class was generated from the following files:

- [/home/frank/Projects/kitlist/src/yamlconfig.hpp](#)
- [/home/frank/Projects/kitlist/src/yamlconfig.cpp](#)



# Chapter 8

## File Documentation

### 8.1 /home/frank/Projects/kitlist/src/category.cpp File Reference

```
#include <algorithm>
#include "category.hpp"
```

### 8.2 /home/frank/Projects/kitlist/src/category.hpp File Reference

```
#include <iostream>
#include <vector>
#include "item.hpp"
```

#### Classes

- class [Category](#)  
*Represents a [Category](#).*
- class [CategoryCompareName](#)  
*Comparator used for sorting [Categories](#) by name.*
- class [CategoryCompareId](#)  
*Comparator used for comparing [Categories](#) by id.*

#### Typedefs

- typedef std::vector< [Category](#) \* > [CategoryContainer](#)
- typedef [CategoryContainer](#)::iterator [CategoryIter](#)

#### 8.2.1 Typedef Documentation

### 8.2.1.1 CategoryContainer

```
typedef std::vector<Category*> CategoryContainer
```

Definition at line 84 of file category.hpp.

### 8.2.1.2 CategoryIter

```
typedef CategoryContainer::iterator CategoryIter
```

Definition at line 85 of file category.hpp.

## 8.3 /home/frank/Projects/kitlist/src/item.hpp File Reference

```
#include <iostream>
#include <list>
#include <string>
#include <vector>
#include <sigc++/signal.h>
```

### Classes

- class [Item](#)  
*Represents an [Item](#).*
- class [ItemCompareName](#)  
*Comparator used for sorting [Items](#) by name.*
- class [ItemCompareId](#)  
*Comparator used for comparing [Items](#) by id.*
- class [ItemFunctor](#)  
*Functor for processing [items](#).*

### Typedefs

- typedef std::vector< [Item](#) \* > [ItemContainer](#)
- typedef ItemContainer::iterator [ItemIter](#)
- typedef std::list< [Item](#) > [ItemList](#)
- typedef ItemList::iterator [ItemListIter](#)
- typedef sigc::slot< bool, [Item](#) & > [SlotForeachItem](#)

### 8.3.1 Typedef Documentation



### 8.3.1.1 ItemContainer

```
typedef std::vector<Item*> ItemContainer
```

Definition at line 91 of file item.hpp.

### 8.3.1.2 ItemIter

```
typedef ItemContainer::iterator ItemIter
```

Definition at line 92 of file item.hpp.

### 8.3.1.3 ItemList

```
typedef std::list<Item> ItemList
```

Definition at line 94 of file item.hpp.

### 8.3.1.4 ItemListIter

```
typedef ItemList::iterator ItemListIter
```

Definition at line 95 of file item.hpp.

### 8.3.1.5 SlotForeachItem

```
typedef sigc::slot<bool, Item&> SlotForeachItem
```

Definition at line 97 of file item.hpp.

## 8.4 /home/frank/Projects/kitlist/src/kitlistdao.hpp File Reference

```
#include "category.hpp"  
#include "item.hpp"  
#include "kitmodel.hpp"  
#include <string>
```

## Classes

- class [KitListDao](#)

*Defines the methods that an implementation of this class must implement.*

## Enumerations

- enum [item\\_choice](#) { [ALL\\_ITEMS](#), [CHECKED\\_ITEMS](#), [UNCHECKED\\_ITEMS](#) }

### 8.4.1 Enumeration Type Documentation

#### 8.4.1.1 item\_choice

```
enum item_choice
```

Options for selecting all items, only checked items, or only unchecked items.

#### Enumerator

ALL_ITEMS	
CHECKED_ITEMS	
UNCHECKED_ITEMS	

Definition at line 36 of file kitlistdao.hpp.

## 8.5 /home/frank/Projects/kitlist/src/kitlistgui.cpp File Reference

```
#include "kitlistgui.hpp"
#include "printing.hpp"
#include "yamlconfig.hpp"
#include <cassert>
#include <glibmm/i18n.h>
#include <glibmm/refptr.h>
#include <glibmm/uststring.h>
#include <gtkmm/aboutdialog.h>
#include <gtkmm/cellrenderertoggle.h>
#include <gtkmm/clipboard.h>
#include <gtkmm/filechooserdialog.h>
#include <gtkmm/menuitem.h>
#include <gtkmm/messagedialog.h>
#include <gtkmm/stock.h>
#include <gtkmm/targetentry.h>
#include <gtkmm/window.h>
#include <libglademmm/xml.h>
#include <libxml++/libxml++.h>
#include <string>
```

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <sys/stat.h>
#include <config.h>
```

## Namespaces

- [anonymous\\_namespace{kitlistgui.cpp}](#)

## Typedefs

- typedef Gtk::TreeModel::Children [type\\_children](#)

## Functions

- const bool [file\\_exists](#) (const Glib::ustring &filename)  
*Returns true if the passed file exists.*
- const string [load\\_resource\\_glade\\_file](#) (const Glib::ustring &filename)  
*Locate the Glade resource file from a list of potential locations.*
- Glib::RefPtr< Gnome::Glade::Xml > [get\\_glade\\_ref\\_ptr](#) (const string &filename, const Glib::ustring &root=Glib::ustring(), const Glib::ustring &domain=Glib::ustring())  
*Returns a reference to the Glade resource file.*

## Variables

- const string [anonymous\\_namespace{kitlistgui.cpp}::GLADE\\_APP\\_FILE](#) = "kitlist.glade"  
*Resource file name.*
- const guint [anonymous\\_namespace{kitlistgui.cpp}::SB\\_ITEM\\_COUNT](#) = 1000  
*Status bar message constant for displaying item counts.*
- const guint [anonymous\\_namespace{kitlistgui.cpp}::SB\\_SAVE](#) = SB\_ITEM\_COUNT + 1  
*Status bar message constant for save notifications.*
- const guint [anonymous\\_namespace{kitlistgui.cpp}::SB\\_MSG](#) = SB\_SAVE + 1  
*Status bar message constant for general messages.*
- const guint [anonymous\\_namespace{kitlistgui.cpp}::SB\\_PRINT](#) = SB\_MSG + 1  
*Status bar message constant for printer messages.*
- const char [anonymous\\_namespace{kitlistgui.cpp}::item\\_target\\_custom](#) [] = "kitlistclipboard"  
*Key used for custom clipboard.*
- const char [anonymous\\_namespace{kitlistgui.cpp}::item\\_target\\_text](#) [] = "text/plain"  
*Mime type for clipboard content.*
- const char [anonymous\\_namespace{kitlistgui.cpp}::XML\\_ELEMENT\\_ID](#) [] = "id"  
*Tag name for the ID element in the clipboard XML document.*
- const Glib::ustring [anonymous\\_namespace{kitlistgui.cpp}::DEFAULT\\_FILENAME\\_EXTENSION](#) = ".kit"  
*Default filename extension.*
- const Glib::ustring [anonymous\\_namespace{kitlistgui.cpp}::PDF\\_FILENAME\\_EXTENSION](#) = ".pdf"  
*PDF filename extension.*
- const Glib::ustring [anonymous\\_namespace{kitlistgui.cpp}::DEFAULT\\_FILENAME](#) = "kitlist" + DEFAULT\_FILENAME\_EXTENSION

*The default filename.*

- `const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY = "/apps/kitlist"`

*The application's root key in the GConf hierarchy.*

- `const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_CURRENT_FILENAME = GCONF_KEY + "/current_filename"`

*GConf entry for the current filename.*

- `const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_PAGE_TITLE = GCONF_KEY + "/page_title"`

*GConf entry for the page title.*

- `const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_RECENT_FILES = GCONF_KEY + "/recent_files"`

*GConf entry for recent files.*

- `const Glib::ustring anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_MAX_RECENT_FILES = GCONF_KEY + "/max_recent_files"`

*GConf entry for max recent files.*

## 8.5.1 Typedef Documentation

### 8.5.1.1 type\_children

```
typedef Gtk::TreeModel::Children type_children
```

Definition at line 124 of file kitlistgui.cpp.

## 8.5.2 Function Documentation

### 8.5.2.1 file\_exists()

```
const bool file_exists (
    const Glib::ustring & filename )
```

Returns true if the passed file exists.

Note: We do not test for the file type, just it's existence regardless of whether it's a directory etc.

#### Parameters

<i>filename</i>	The file to test for existence.
-----------------	---------------------------------

#### Returns

true if the file exists.

Definition at line 135 of file kitlistgui.cpp.

Referenced by KitListGui::choose\_filename(), KitListGui::choose\_pdf\_filename(), KitListGui::init(), and load\_resource\_glade\_file().

### 8.5.2.2 get\_glade\_ref\_ptr()

```
Glib::RefPtr<Gnome::Glade::Xml> get_glade_ref_ptr (
    const string & filename,
    const Glib::ustring & root = Glib::ustring(),
    const Glib::ustring & domain = Glib::ustring() )
```

Returns a reference to the Glade resource file.

Attempts to locate the external Glade resource file from a number of common locations.

Definition at line 164 of file kitlistgui.cpp.

References load\_resource\_glade\_file().

Referenced by KitListGui::init().

### 8.5.2.3 load\_resource\_glade\_file()

```
const string load_resource_glade_file (
    const Glib::ustring & filename )
```

Locate the Glade resource file from a list of potential locations.

Definition at line 145 of file kitlistgui.cpp.

References file\_exists().

Referenced by get\_glade\_ref\_ptr().

## 8.6 /home/frank/Projects/kitlist/src/kitlistgui.hpp File Reference

```
#include "service.hpp"
#include "yamlconfig.hpp"
#include <iostream>
#include <gtkmm/button.h>
#include <gtkmm/checkbutton.h>
#include <gtkmm/combobox.h>
#include <gtkmm/entry.h>
#include <gtkmm/imagemenuitem.h>
#include <gtkmm/main.h>
#include <gtkmm/statusbar.h>
#include <gtkmm/toolbutton.h>
#include <gtkmm/treemodelcolumn.h>
#include <gtkmm/liststore.h>
#include <gtkmm/pagesetup.h>
#include <gtkmm/printsettings.h>
#include <gtkmm/printoperation.h>
```

## Classes

- class [ModelCategoryColumns](#)  
*A definition for displaying a [ModelCategory](#) in a combo box.*
- class [ModellItemColumns](#)  
*A definition for displaying an item in a multi-column list.*
- class [KitListGui](#)  
*Encapsulates the methods for the application's GUI front end.*

## Enumerations

- enum [gui\\_state](#) { [ADD\\_CATEGORY](#), [RENAME\\_CATEGORY](#) }

## Variables

- const int [CHECKED\\_COL\\_POSITION](#) = 0  
*The position in the list of the tick box column.*

### 8.6.1 Enumeration Type Documentation

#### 8.6.1.1 [gui\\_state](#)

```
enum gui\_state
```

##### Enumerator

<a href="#">ADD_CATEGORY</a>	
<a href="#">RENAME_CATEGORY</a>	

Definition at line 53 of file [kitlistgui.hpp](#).

### 8.6.2 Variable Documentation

#### 8.6.2.1 [CHECKED\\_COL\\_POSITION](#)

```
const int CHECKED\_COL\_POSITION = 0
```

The position in the list of the tick box column.

Definition at line 72 of file [kitlistgui.hpp](#).

## 8.7 /home/frank/Projects/kitlist/src/kitlistpgsqldao.cpp File Reference

```
#include <cassert>
#include <iostream>
#include <sstream>
#include "kitlistpgsqldao.hpp"
```

## 8.8 /home/frank/Projects/kitlist/src/kitlistpgsqldao.hpp File Reference

```
#include <config.h>
```

## 8.9 /home/frank/Projects/kitlist/src/kitmodel.cpp File Reference

```
#include "kitmodel.hpp"
#include <algorithm>
#include <cassert>
#include <glib.h>
#include <iostream>
#include <iterator>
```

## 8.10 /home/frank/Projects/kitlist/src/kitmodel.hpp File Reference

```
#include "item.hpp"
#include "category.hpp"
#include <map>
#include <sigc++/signal.h>
```

### Classes

- class [GuiState](#)  
*Class encapsulating state of an object in the data model.*
- class [ModellItem](#)  
*Represents an [Item](#) combined with [GuiState](#) attributes.*
- class [ModellItemCompareId](#)  
*Comparator for comparing items by their unique ID.*
- class [ModelCategory](#)  
*Represents a [Category](#) combined with [GuiState](#) attributes.*
- class [KitModel](#)  
*Holds a rich graph of objects representing the application's data model.*

## Typedefs

- typedef std::vector< [ModellItem](#) \* > [ModellItemContainer](#)
- typedef ModellItemContainer::iterator [ModellItemIter](#)
- typedef std::map< long, [ModellItem](#) \* > [ItemMap](#)
- typedef ItemMap::iterator [ItemMapIter](#)
- typedef sigc::slot< bool, const ItemMap::iterator & > [SlotForeachModellItemIter](#)
- typedef sigc::slot< bool, [ModellItem](#) & > [SlotForeachModellItem](#)
- typedef std::vector< [ModelCategory](#) \* > [ModelCategoryContainer](#)
- typedef ModelCategoryContainer::iterator [ModelCategoryIter](#)
- typedef std::map< long, [ModelCategory](#) \* > [CategoryMap](#)
- typedef CategoryMap::iterator [CategoryMapIter](#)
- typedef sigc::slot< bool, const CategoryMap::iterator & > [SlotForeachCategoryIter](#)
- typedef sigc::slot< bool, [ModelCategory](#) & > [SlotForeachCategory](#)

## Enumerations

- enum [item\\_filter\\_types](#) { [ALL](#), [CHECKED](#), [UNCHECKED](#) }

### 8.10.1 Typedef Documentation

#### 8.10.1.1 CategoryMap

```
typedef std::map<long, ModelCategory\*> CategoryMap
```

Definition at line 122 of file kitmodel.hpp.

#### 8.10.1.2 CategoryMapIter

```
typedef CategoryMap::iterator CategoryMapIter
```

Definition at line 123 of file kitmodel.hpp.

#### 8.10.1.3 ItemMap

```
typedef std::map<long, ModelItem\*> ItemMap
```

Definition at line 84 of file kitmodel.hpp.



#### 8.10.1.4 ItemMapIter

```
typedef ItemMap::iterator ItemMapIter
```

Definition at line 85 of file kitmodel.hpp.

#### 8.10.1.5 ModelCategoryContainer

```
typedef std::vector<ModelCategory*> ModelCategoryContainer
```

Definition at line 119 of file kitmodel.hpp.

#### 8.10.1.6 ModelCategoryIter

```
typedef ModelCategoryContainer::iterator ModelCategoryIter
```

Definition at line 120 of file kitmodel.hpp.

#### 8.10.1.7 ModelItemContainer

```
typedef std::vector<ModelItem*> ModelItemContainer
```

Definition at line 81 of file kitmodel.hpp.

#### 8.10.1.8 ModelItemIter

```
typedef ModelItemContainer::iterator ModelItemIter
```

Definition at line 82 of file kitmodel.hpp.

#### 8.10.1.9 SlotForeachCategory

```
typedef sigc::slot<bool, ModelCategory*> SlotForeachCategory
```

Definition at line 126 of file kitmodel.hpp.

### 8.10.1.10 SlotForeachCategoryIter

```
typedef sigc::slot<bool, const CategoryMap::iterator&> SlotForeachCategoryIter
```

Definition at line 125 of file kitmodel.hpp.

### 8.10.1.11 SlotForeachModelItem

```
typedef sigc::slot<bool, ModelItem&> SlotForeachModelItem
```

Definition at line 88 of file kitmodel.hpp.

### 8.10.1.12 SlotForeachModelItemIter

```
typedef sigc::slot<bool, const ItemMap::iterator&> SlotForeachModelItemIter
```

Definition at line 87 of file kitmodel.hpp.

## 8.10.2 Enumeration Type Documentation

### 8.10.2.1 item\_filter\_types

```
enum item_filter_types
```

#### Enumerator

ALL	
CHECKED	
UNCHECKED	

Definition at line 128 of file kitmodel.hpp.

## 8.11 /home/frank/Projects/kitlist/src/kitparser.cpp File Reference

```
#include "kitparser.hpp"
#include <algorithm>
#include <iostream>
```

## 8.12 /home/frank/Projects/kitlist/src/kitparser.hpp File Reference

```
#include <libxml++/libxml++.h>
#include "kitmodel.hpp"
```

### Classes

- class [KitParser](#)  
*SaxParser implementation for reading the [KitModel](#) from an XML document.*

## 8.13 /home/frank/Projects/kitlist/src/main.cpp File Reference

```
#include <config.h>
#include <locale.h>
#include <iostream>
#include <getopt.h>
#include <libintl.h>
#include "kitlistgui.hpp"
#include "kitlistpgsdao.hpp"
#include "xmldao.hpp"
```

### Classes

- class [KitList](#)  
*Main application class.*
- class [TickItem](#)

### Functions

- int [main](#) (int argc, char \*\*argv)

### Variables

- static int [verbose\\_flag](#) = 0  
*Flag set by '-verbose'.*
- static int [html\\_flag](#) = 0  
*Flag set by '-html'.*

### 8.13.1 Function Documentation

### 8.13.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

Parses the command line and executes the main application.

Definition at line 559 of file main.cpp.

References `KitList::add_item()`, `ALL_ITEMS`, `KitList::append_items_to_category()`, `KitList::associate_item_with_category()`, `CHECKED_ITEMS`, `KitList::delete_category()`, `KitList::delete_item()`, `KitList::get_dao()`, `html_flag`, `KitList::list_categories()`, `KitList::list_items()`, `KitList::new_category()`, `KitList::remove_item_from_category()`, `KitList::set_all_flags()`, `KitList::set_category_flag()`, `KitList::set_item_flag()`, `KitList::tick_items()`, `UNCHECKED_ITEMS`, `KitList::unset_all_flags()`, `KitList::unset_category_flag()`, `KitList::unset_item_flag()`, and `verbose_flag`.

## 8.13.2 Variable Documentation

### 8.13.2.1 html\_flag

```
int html_flag = 0 [static]
```

Flag set by `'-html'`.

Definition at line 69 of file main.cpp.

Referenced by `KitList::list_categories()`, `KitList::list_item()`, `KitList::list_items_end()`, `KitList::list_items_start()`, and `main()`.

### 8.13.2.2 verbose\_flag

```
int verbose_flag = 0 [static]
```

Flag set by `'-verbose'`.

Definition at line 66 of file main.cpp.

Referenced by `main()`, and `KitListDao::set_verbose()`.

## 8.14 /home/frank/Projects/kitlist/src/printing.cpp File Reference

```
#include "printing.hpp"
#include <sstream>
#include <glibmm/i18n.h>
#include <config.h>
```

## Variables

- const int `PAGE_TOLERANCE` = 40  
*Seems calculating page body height is inaccurate.*
- const int `HEADER_SPACING` = 10  
*The spacing between the header and the body.*
- const int `FOOTER_SPACING` = 10  
*The spacing between the footer and the body.*
- const int `BORDER_SPACING` = 10  
*The space above and below the horizontal lines at the top and bottom of each page.*
- const Glib::ustring `FOOTER_TEXT` = \_("Page %1 of %2")  
*Text used to print page number in footer - Page x of n.*

### 8.14.1 Variable Documentation

#### 8.14.1.1 BORDER\_SPACING

```
const int BORDER_SPACING = 10
```

The space above and below the horizontal lines at the top and bottom of each page.

Definition at line 38 of file printing.cpp.

Referenced by `KitPrintOperation::on_begin_print()`, and `KitPrintOperation::on_draw_page()`.

#### 8.14.1.2 FOOTER\_SPACING

```
const int FOOTER_SPACING = 10
```

The spacing between the footer and the body.

Definition at line 35 of file printing.cpp.

Referenced by `KitPrintOperation::on_begin_print()`.

#### 8.14.1.3 FOOTER\_TEXT

```
const Glib::ustring FOOTER_TEXT = _("Page %1 of %2")
```

Text used to print page number in footer - Page x of n.

Definition at line 41 of file printing.cpp.

Referenced by `KitPrintOperation::new_footer()`, and `KitPrintOperation::on_begin_print()`.

#### 8.14.1.4 HEADER\_SPACING

```
const int HEADER_SPACING = 10
```

The spacing between the header and the body.

Definition at line 32 of file printing.cpp.

Referenced by `KitPrintOperation::on_begin_print()`, and `KitPrintOperation::on_draw_page()`.

#### 8.14.1.5 PAGE\_TOLERANCE

```
const int PAGE_TOLERANCE = 40
```

Seems calculating page body height is inaccurate.

Definition at line 29 of file printing.cpp.

Referenced by `KitPrintOperation::on_begin_print()`.

### 8.15 /home/frank/Projects/kitlist/src/printing.hpp File Reference

```
#include "item.hpp"  
#include <gtkmm/printcontext.h>  
#include <gtkmm/printoperation.h>
```

#### Classes

- class [KitPrintOperation](#)  
*Prints the kitlist.*

#### Typedefs

- typedef `Glib::RefPtr<Pango::Layout>` [layout\\_refptr](#)

#### 8.15.1 Typedef Documentation

##### 8.15.1.1 layout\_refptr

```
typedef Glib::RefPtr<Pango::Layout> layout\_refptr
```

Definition at line 30 of file printing.hpp.

## 8.16 /home/frank/Projects/kitlist/src/service.cpp File Reference

```
#include "service.hpp"  
#include <algorithm>  
#include <cassert>  
#include <iostream>
```

### Classes

- class [FilterItem](#)

## 8.17 /home/frank/Projects/kitlist/src/service.hpp File Reference

```
#include "kitlistdao.hpp"  
#include "kitmodel.hpp"  
#include "xmldao.hpp"  
#include <glibmm/ustring.h>
```

### Classes

- class [Service](#)  
*Business/service layer implementation.*

## 8.18 /home/frank/Projects/kitlist/src/xmldao.cpp File Reference

```
#include <algorithm>  
#include <iostream>  
#include <sstream>  
#include "xmldao.hpp"  
#include "kitparser.hpp"
```

## 8.19 /home/frank/Projects/kitlist/src/xmldao.hpp File Reference

```
#include <cassert>  
#include <libxml++/libxml++.h>  
#include "kitlistdao.hpp"  
#include "kitmodel.hpp"
```

### Classes

- class [XmlDao](#)  
*Implementation of a [KitListDao](#) using XML as the persistence store.*

## Macros

- `#define XMLDAO_H 1`
- `#define NYI assert(false/* == "Method not implemented"*/)`

### 8.19.1 Macro Definition Documentation

#### 8.19.1.1 NYI

```
#define NYI assert(false/* == "Method not implemented"*/)
```

Definition at line 35 of file `xml dao.hpp`.

Referenced by `XmlDao::add_item()`, `XmlDao::append_items_to_category()`, `XmlDao::associate_item_with_category()`, `XmlDao::delete_category()`, `XmlDao::delete_item()`, `XmlDao::get_all_items()`, `XmlDao::get_categories()`, `XmlDao::get_category()`, `XmlDao::new_category()`, `XmlDao::remove_item_from_category()`, `XmlDao::set_all_flags()`, `XmlDao::set_category_flag()`, `XmlDao::set_item_flag()`, `XmlDao::unset_all_flags()`, `XmlDao::unset_category_flag()`, `XmlDao::unset_item_flag()`, and `XmlDao::update_item_checked_state()`.

#### 8.19.1.2 XMLDAO\_H

```
#define XMLDAO_H 1
```

Definition at line 28 of file `xml dao.hpp`.

## 8.20 /home/frank/Projects/kitlist/src/yamlconfig.cpp File Reference

```
#include "yamlconfig.hpp"
#include <config.h>
#include <glibmm/miscutils.h>
#include <fstream>
#include <iostream>
#include <yaml-cpp/yaml.h>
```

### Variables

- `const std::string CONFIG_FILENAME = "/kitlist"`  
*The filename used to store the application's configuration.*
- `const std::string PAGE_TITLE_CONFIG_KEY = "Printed page title"`  
*The key used to store the value of the page title in the configuration file.*
- `const std::string CURRENT_FILENAME_CONFIG_KEY = "current filename"`  
*The key used to store the value of the current filename in the configuration file.*
- `const std::string RECENT_FILES_CONFIG_KEY = "file history list"`  
*The key used to store the value of the current filename in the configuration file.*
- `const std::string MAX_RECENT_FILES_CONFIG_KEY = "max recent files"`  
*The key for the maximum number of files to maintain in the recent files menu.*
- `const std::string DEBUG_LOG_FILENAME_CONFIG_KEY = "debug_log_filename"`  
*GConf entry for the page title.*



## 8.20.1 Variable Documentation

### 8.20.1.1 CONFIG\_FILENAME

```
const std::string CONFIG_FILENAME = "/kitlist"
```

The filename used to store the application's configuration.

Definition at line 34 of file yamlconfig.cpp.

Referenced by `YamlConfig::get_config_filename()`.

### 8.20.1.2 CURRENT\_FILENAME\_CONFIG\_KEY

```
const std::string CURRENT_FILENAME_CONFIG_KEY = "current filename"
```

The key used to store the value of the current filename in the configuration file.

Definition at line 40 of file yamlconfig.cpp.

Referenced by `YamlConfig::load()`, and `YamlConfig::save()`.

### 8.20.1.3 DEBUG\_LOG\_FILENAME\_CONFIG\_KEY

```
const std::string DEBUG_LOG_FILENAME_CONFIG_KEY = "debug_log_filename"
```

GConf entry for the page title.

Definition at line 49 of file yamlconfig.cpp.

Referenced by `YamlConfig::load()`, and `YamlConfig::save()`.

### 8.20.1.4 MAX\_RECENT\_FILES\_CONFIG\_KEY

```
const std::string MAX_RECENT_FILES_CONFIG_KEY = "max recent files"
```

The key for the maximum number of files to maintain in the recent files menu.

Definition at line 46 of file yamlconfig.cpp.

Referenced by `YamlConfig::load()`, and `YamlConfig::save()`.

### 8.20.1.5 PAGE\_TITLE\_CONFIG\_KEY

```
const std::string PAGE_TITLE_CONFIG_KEY = "Printed page title"
```

The key used to store the value of the page title in the configuration file.

Definition at line 37 of file yamlconfig.cpp.

Referenced by `YamlConfig::load()`, and `YamlConfig::save()`.

### 8.20.1.6 RECENT\_FILES\_CONFIG\_KEY

```
const std::string RECENT_FILES_CONFIG_KEY = "file history list"
```

The key used to store the value of the current filename in the configuration file.

Definition at line 43 of file yamlconfig.cpp.

Referenced by `YamlConfig::load()`, and `YamlConfig::save()`.

## 8.21 /home/frank/Projects/kitlist/src/yamlconfig.hpp File Reference

```
#include <deque>
#include <glibmm/i18n.h>
#include <glibmm/ustring.h>
#include <iostream>
#include <string>
#include <config.h>
```

### Classes

- class [YamlConfig](#)  
*Maintains the application's configuration parameters in a YAML formatted file.*

### Variables

- const std::string [DEFAULT\\_PAGE\\_TITLE](#) = \_("Kitlist")  
*The default page title for printing.*
- const gint [DEFAULT\\_MAX\\_RECENT\\_FILES](#) = 4  
*The maximum number of recent files to maintain.*

### 8.21.1 Variable Documentation

### 8.21.1.1 DEFAULT\_MAX\_RECENT\_FILES

```
const gint DEFAULT_MAX_RECENT_FILES = 4
```

The maximum number of recent files to maintain.

Definition at line 35 of file yamlconfig.hpp.

Referenced by KitListGui::get\_max\_recent\_files().

### 8.21.1.2 DEFAULT\_PAGE\_TITLE

```
const std::string DEFAULT_PAGE_TITLE = _("Kitlist")
```

The default page title for printing.

Definition at line 32 of file yamlconfig.hpp.

Referenced by KitListGui::init(), and YamlConfig::load().



# Index

- [/home/frank/Projects/kitlist/src/category.cpp](#), 159
- [/home/frank/Projects/kitlist/src/category.hpp](#), 159
- [/home/frank/Projects/kitlist/src/item.hpp](#), 160
- [/home/frank/Projects/kitlist/src/kitlistdao.hpp](#), 161
- [/home/frank/Projects/kitlist/src/kitlistgui.cpp](#), 162
- [/home/frank/Projects/kitlist/src/kitlistgui.hpp](#), 165
- [/home/frank/Projects/kitlist/src/kitlistpgsqldao.cpp](#), 167
- [/home/frank/Projects/kitlist/src/kitlistpgsqldao.hpp](#), 167
- [/home/frank/Projects/kitlist/src/kitmodel.cpp](#), 167
- [/home/frank/Projects/kitlist/src/kitmodel.hpp](#), 167
- [/home/frank/Projects/kitlist/src/kitparser.cpp](#), 170
- [/home/frank/Projects/kitlist/src/kitparser.hpp](#), 171
- [/home/frank/Projects/kitlist/src/main.cpp](#), 171
- [/home/frank/Projects/kitlist/src/printing.cpp](#), 172
- [/home/frank/Projects/kitlist/src/printing.hpp](#), 174
- [/home/frank/Projects/kitlist/src/service.cpp](#), 175
- [/home/frank/Projects/kitlist/src/service.hpp](#), 175
- [/home/frank/Projects/kitlist/src/xmldao.cpp](#), 175
- [/home/frank/Projects/kitlist/src/xmldao.hpp](#), 175
- [/home/frank/Projects/kitlist/src/yamlconfig.cpp](#), 176
- [/home/frank/Projects/kitlist/src/yamlconfig.hpp](#), 178
- ~Category
  - [Category](#), 18
- ~KitList
  - [KitList](#), 39
- ~KitListDao
  - [KitListDao](#), 50
- ~KitListGui
  - [KitListGui](#), 63
- ~KitModel
  - [KitModel](#), 92
- ~KitParser
  - [KitParser](#), 102
- ~KitPrintOperation
  - [KitPrintOperation](#), 108
- ~ModelCategory
  - [ModelCategory](#), 113
- ~Service
  - [Service](#), 125
- ~YamlConfig
  - [YamlConfig](#), 152
- [add\\_category](#)
  - [KitModel](#), 93
- [add\\_category\\_item\\_to\\_dom](#)
  - [XmlDao](#), 139
- [add\\_category\\_to\\_dom](#)
  - [XmlDao](#), 139
- [add\\_item](#)
  - [Category](#), 18
  - [KitList](#), 39
  - [KitListDao](#), 50
  - [KitModel](#), 93
  - [ModelCategory](#), 113
  - [XmlDao](#), 140
- [add\\_item\\_to\\_dom](#)
  - [XmlDao](#), 141
- [add\\_items](#)
  - [KitListGui](#), 63
- [add\\_recent\\_filename](#)
  - [YamlConfig](#), 152
- [anonymous\\_namespace{kitlistgui.cpp}](#), 11
  - [DEFAULT\\_FILENAME\\_EXTENSION](#), 12
  - [DEFAULT\\_FILENAME](#), 12
  - [GCONF\\_KEY\\_CURRENT\\_FILENAME](#), 12
  - [GCONF\\_KEY\\_MAX\\_RECENT\\_FILES](#), 12
  - [GCONF\\_KEY\\_PAGE\\_TITLE](#), 13
  - [GCONF\\_KEY\\_RECENT\\_FILES](#), 13
  - [GCONF\\_KEY](#), 12
  - [GLADE\\_APP\\_FILE](#), 13
  - [item\\_target\\_custom](#), 13
  - [item\\_target\\_text](#), 14
  - [PDF\\_FILENAME\\_EXTENSION](#), 14
  - [SB\\_ITEM\\_COUNT](#), 14
  - [SB\\_MSG](#), 14
  - [SB\\_PRINT](#), 15
  - [SB\\_SAVE](#), 15
  - [XML\\_ELEMENT\\_ID](#), 15
- [append\\_items\\_to\\_category](#)
  - [KitList](#), 39
  - [KitListDao](#), 51
  - [XmlDao](#), 141
- [associate\\_item\\_with\\_category](#)
  - [KitList](#), 41
  - [KitListDao](#), 51
  - [XmlDao](#), 141
- [BORDER\\_SPACING](#)
  - [printing.cpp](#), 173
- [CHECKED\\_COL\\_POSITION](#)
  - [kitlistgui.hpp](#), 166
- [CONFIG\\_FILENAME](#)
  - [yamlconfig.cpp](#), 177
- [CURRENT\\_FILENAME\\_CONFIG\\_KEY](#)
  - [yamlconfig.cpp](#), 177
- [cancel\\_add\\_category\\_window](#)
  - [KitListGui](#), 63
- [cancel\\_add\\_item\\_window](#)
  - [KitListGui](#), 63

- cancel\_preferences\_window
  - KitListGui, [64](#)
- Category, [17](#)
  - ~Category, [18](#)
  - add\_item, [18](#)
  - CategoryCompareId, [21](#)
  - CategoryCompareName, [21](#)
  - execute, [18](#)
  - foreach\_item, [19](#)
  - get\_id, [19](#)
  - get\_name, [19](#)
  - has\_items, [20](#)
  - item\_count, [20](#)
  - KitModel, [21](#)
  - m\_id, [22](#)
  - m\_items, [22](#)
  - m\_name, [22](#)
  - remove\_item, [20](#)
  - set\_id, [21](#)
  - set\_name, [21](#)
- category.hpp
  - CategoryContainer, [159](#)
  - CategoryItr, [160](#)
- CategoryCompareId, [23](#)
  - Category, [21](#)
  - CategoryCompareId, [23](#)
  - operator(), [23](#)
- CategoryCompareName, [24](#)
  - Category, [21](#)
  - CategoryCompareName, [24](#)
  - operator(), [24](#)
- CategoryContainer
  - category.hpp, [159](#)
- CategoryItr
  - category.hpp, [160](#)
- CategoryMap
  - kitmodel.hpp, [168](#)
- CategoryMapItr
  - kitmodel.hpp, [168](#)
- choose\_filename
  - KitListGui, [64](#)
- choose\_pdf\_filename
  - KitListGui, [64](#)
- close\_add\_category\_window
  - KitListGui, [65](#)
- close\_add\_item\_window
  - KitListGui, [65](#)
- close\_preferences\_window
  - KitListGui, [65](#)
- confirm\_lose\_changes
  - KitListGui, [66](#)
- copy\_items
  - KitModel, [94](#)
  - Service, [126](#)
- copy\_selected\_items\_to\_clipboard
  - KitListGui, [66](#)
- create
  - KitPrintOperation, [108](#)
- create\_category
  - Service, [126](#)
- create\_default\_model
  - Service, [126](#)
- create\_item
  - Service, [126](#)
- DEBUG\_LOG\_FILENAME\_CONFIG\_KEY
  - yamlconfig.cpp, [177](#)
- DEFAULT\_FILENAME\_EXTENSION
  - anonymous\_namespace{kitlistgui.cpp}, [12](#)
- DEFAULT\_FILENAME
  - anonymous\_namespace{kitlistgui.cpp}, [12](#)
- DEFAULT\_MAX\_RECENT\_FILES
  - yamlconfig.hpp, [178](#)
- DEFAULT\_PAGE\_TITLE
  - yamlconfig.hpp, [179](#)
- delete\_category
  - KitList, [41](#)
  - KitListDao, [52](#)
  - Service, [127](#)
  - XmlDao, [142](#)
- delete\_item
  - KitList, [41](#)
  - KitListDao, [52](#)
  - Service, [127](#)
  - XmlDao, [142](#)
- delete\_selected\_items
  - KitListGui, [66](#)
- execute
  - Category, [18](#)
  - KitList, [42](#)
- FOOTER\_SPACING
  - printing.cpp, [173](#)
- FOOTER\_TEXT
  - printing.cpp, [173](#)
- file\_exists
  - kitlistgui.cpp, [164](#)
- filter
  - KitModel, [94](#)
  - Service, [127](#)
- FilterItem, [25](#)
  - FilterItem, [25](#)
  - m\_model, [26](#)
  - operator(), [25](#)
- find\_category
  - KitModel, [95](#)
  - Service, [129](#)
- find\_item
  - KitModel, [95](#)
  - Service, [129](#)
- foreach\_category
  - KitModel, [95](#)
- foreach\_category\_iter
  - KitModel, [95](#)
- foreach\_item
  - Category, [19](#)

- KitModel, 96
- foreach\_item\_iter
  - KitModel, 96
- GCONF\_KEY\_CURRENT\_FILENAME
  - anonymous\_namespace{kitlistgui.cpp}, 12
- GCONF\_KEY\_MAX\_RECENT\_FILES
  - anonymous\_namespace{kitlistgui.cpp}, 12
- GCONF\_KEY\_PAGE\_TITLE
  - anonymous\_namespace{kitlistgui.cpp}, 13
- GCONF\_KEY\_RECENT\_FILES
  - anonymous\_namespace{kitlistgui.cpp}, 13
- GCONF\_KEY
  - anonymous\_namespace{kitlistgui.cpp}, 12
- GLADE\_APP\_FILE
  - anonymous\_namespace{kitlistgui.cpp}, 13
- get\_added\_children
  - ModelCategory, 114
- get\_all\_items
  - KitListDao, 52
  - KitModel, 96
  - XmlDao, 143
- get\_categories
  - KitListDao, 53
  - KitModel, 97
  - Service, 129
  - XmlDao, 143
- get\_category
  - KitListDao, 53
  - XmlDao, 143
- get\_checked
  - Item, 31
- get\_config\_filename
  - YamlConfig, 152
- get\_current\_filename
  - YamlConfig, 153
- get\_dao
  - KitList, 42
- get\_debug\_log\_filename
  - YamlConfig, 153
- get\_description
  - Item, 31
- get\_filtered\_items
  - Service, 130
- get\_glade\_ref\_ptr
  - kitlistgui.cpp, 165
- get\_id
  - Category, 19
  - Item, 31
- get\_items
  - ModelCategory, 114
  - Service, 130
- get\_max\_recent\_files
  - KitListGui, 67
- get\_model
  - KitListDao, 53
  - XmlDao, 144
- get\_model\_items
  - ModelCategory, 115
- get\_name
  - Category, 19
- get\_next\_category\_id
  - KitListDao, 53
  - Service, 131
  - XmlDao, 144
- get\_next\_item\_id
  - KitListDao, 54
  - Service, 131
  - XmlDao, 145
- get\_page\_title
  - YamlConfig, 153
- get\_recent\_filenames
  - YamlConfig, 154
- get\_removed\_children
  - ModelCategory, 115
- get\_selected\_category
  - KitListGui, 67
- get\_selected\_items
  - KitListGui, 67
- gui\_state
  - kitlistgui.hpp, 166
- GuiState, 26
  - GuiState, 27
  - is\_deleted, 27
  - is\_dirty, 27
  - is\_new, 27
  - m\_deleted, 29
  - m\_dirty, 29
  - m\_new, 29
  - reset, 28
  - set\_deleted, 28
  - set\_dirty, 28
  - set\_new\_flag, 28
- HEADER\_SPACING
  - printing.cpp, 173
- has\_items
  - Category, 20
- html\_flag
  - main.cpp, 172
- init
  - KitListGui, 68
- init\_add\_item\_window
  - KitListGui, 68
- is\_deleted
  - GuiState, 27
- is\_dirty
  - GuiState, 27
  - KitModel, 97
- is\_model\_dirty
  - Service, 131
- is\_new
  - GuiState, 27
- is\_verbose
  - KitListDao, 54
- Item, 30
  - get\_checked, 31

- get\_description, 31
- get\_id, 31
- Item, 31
- ItemCompareId, 32
- ItemCompareName, 33
- m\_checked, 33
- m\_desc, 33
- m\_id, 33
- operator<<, 33
- set\_checked, 32
- set\_description, 32
- set\_id, 32
- item.hpp
  - ItemContainer, 160
  - ItemIter, 161
  - ItemList, 161
  - ItemListIter, 161
  - SlotForEachItem, 161
- item\_choice
  - kitlistdao.hpp, 162
- item\_count
  - Category, 20
- item\_filter\_types
  - kitmodel.hpp, 170
- item\_target\_custom
  - anonymous\_namespace{kitlistgui.cpp}, 13
- item\_target\_text
  - anonymous\_namespace{kitlistgui.cpp}, 14
- ItemCompareId, 34
  - Item, 32
  - ItemCompareId, 34
  - operator(), 35
- ItemCompareName, 35
  - Item, 33
  - ItemCompareName, 35
  - operator(), 36
- ItemContainer
  - item.hpp, 160
- ItemFunctor, 36
  - operator(), 37
- ItemIter
  - item.hpp, 161
- ItemList
  - item.hpp, 161
- ItemListIter
  - item.hpp, 161
- ItemMap
  - kitmodel.hpp, 168
- ItemMapIter
  - kitmodel.hpp, 168
- KitList, 37
  - ~KitList, 39
  - add\_item, 39
  - append\_items\_to\_category, 39
  - associate\_item\_with\_category, 41
  - delete\_category, 41
  - delete\_item, 41
  - execute, 42
  - get\_dao, 42
  - KitList, 38
  - list\_categories, 42
  - list\_item, 42
  - list\_items, 43
  - list\_items\_end, 44
  - list\_items\_start, 44
  - m\_dao, 48
  - new\_category, 44
  - on\_list\_item, 45
  - remove\_item\_from\_category, 45
  - set\_all\_flags, 45
  - set\_category\_flag, 46
  - set\_item\_flag, 46
  - tick\_items, 46, 47
  - unset\_all\_flags, 47
  - unset\_category\_flag, 47
  - unset\_item\_flag, 47
- KitListDao, 48
  - ~KitListDao, 50
  - add\_item, 50
  - append\_items\_to\_category, 51
  - associate\_item\_with\_category, 51
  - delete\_category, 52
  - delete\_item, 52
  - get\_all\_items, 52
  - get\_categories, 53
  - get\_category, 53
  - get\_model, 53
  - get\_next\_category\_id, 53
  - get\_next\_item\_id, 54
  - is\_verbose, 54
  - KitListDao, 50
  - m\_verbose\_flag, 58
  - new\_category, 54
  - remove\_item\_from\_category, 55
  - require\_filename, 55
  - save\_model, 55
  - set\_all\_flags, 56
  - set\_category\_flag, 56
  - set\_item\_flag, 56
  - set\_verbose, 57
  - unset\_all\_flags, 57
  - unset\_category\_flag, 57
  - unset\_item\_flag, 57
  - update\_item\_checked\_state, 58
- KitListGui, 58
  - ~KitListGui, 63
  - add\_items, 63
  - cancel\_add\_category\_window, 63
  - cancel\_add\_item\_window, 63
  - cancel\_preferences\_window, 64
  - choose\_filename, 64
  - choose\_pdf\_filename, 64
  - close\_add\_category\_window, 65
  - close\_add\_item\_window, 65
  - close\_preferences\_window, 65
  - confirm\_lose\_changes, 66



copy\_selected\_items\_to\_clipboard, 66  
 delete\_selected\_items, 66  
 get\_max\_recent\_files, 67  
 get\_selected\_category, 67  
 get\_selected\_items, 67  
 init, 68  
 init\_add\_item\_window, 68  
 KitListGui, 62  
 m\_category\_cols, 83  
 m\_category\_combo, 83  
 m\_checkbutton\_add\_item, 83  
 m\_clipboard\_items, 84  
 m\_current\_cat\_id, 84  
 m\_entry\_add\_category, 84  
 m\_entry\_add\_item, 84  
 m\_entry\_page\_title, 85  
 m\_file\_save\_menu\_item, 85  
 m\_file\_save\_tool\_button, 85  
 m\_filename, 85  
 m\_ignore\_list\_events, 86  
 m\_item\_cols, 86  
 m\_item\_tree\_view, 86  
 m\_kit, 86  
 m\_page\_title, 87  
 m\_paste\_menu\_item, 87  
 m\_paste\_tool\_button, 87  
 m\_recent\_files\_menu\_item, 87  
 m\_ref\_category\_list\_store, 88  
 m\_ref\_item\_tree\_model, 88  
 m\_ref\_page\_setup, 88  
 m\_ref\_printer\_settings, 88  
 m\_service, 89  
 m\_state, 89  
 m\_status\_bar, 89  
 m\_window, 89  
 m\_window\_add\_category, 90  
 m\_window\_add\_item, 90  
 m\_window\_preferences, 90  
 m\_yaml\_config, 90  
 on\_category\_change, 68  
 on\_cell\_edit, 69  
 on\_clipboard\_clear, 69  
 on\_clipboard\_get, 69  
 on\_clipboard\_received, 70  
 on\_delete\_event, 70  
 on\_menu\_add, 70  
 on\_menu\_check\_selected, 71  
 on\_menu\_copy, 71  
 on\_menu\_create\_category, 71  
 on\_menu\_cut, 71  
 on\_menu\_delete, 72  
 on\_menu\_delete\_category, 72  
 on\_menu\_export\_to\_pdf, 72  
 on\_menu\_file\_new, 73  
 on\_menu\_file\_open, 73  
 on\_menu\_help\_about, 73  
 on\_menu\_paste, 74  
 on\_menu\_preferences, 74  
 on\_menu\_print, 74  
 on\_menu\_quit, 74  
 on\_menu\_recent\_file, 75  
 on\_menu\_rename\_category, 75  
 on\_menu\_save, 75  
 on\_menu\_save\_as, 76  
 on\_menu\_select\_all, 76  
 on\_menu\_show\_all, 76  
 on\_menu\_show\_checked, 77  
 on\_menu\_show\_unchecked, 77  
 on\_menu\_uncheck\_selected, 77  
 on\_printoperation\_done, 77  
 on\_printoperation\_status\_changed, 78  
 on\_row\_changed, 78  
 open\_file, 78  
 paste\_from\_xml, 79  
 paste\_status\_received, 79  
 raise, 79  
 refresh\_category\_list, 80  
 refresh\_item\_list, 80  
 run, 80  
 safe\_open\_file, 81  
 selected\_row\_callback, 81  
 set\_page\_title, 81  
 set\_selected, 81  
 toggle\_selected, 82  
 update\_item\_count, 82  
 update\_paste\_status, 82  
 update\_recent\_files, 82  
 update\_recent\_files\_menu, 83  
 KitModel, 91  
   ~KitModel, 92  
   add\_category, 93  
   add\_item, 93  
   Category, 21  
   copy\_items, 94  
   filter, 94  
   find\_category, 95  
   find\_item, 95  
   foreach\_category, 95  
   foreach\_category\_iter, 95  
   foreach\_item, 96  
   foreach\_item\_iter, 96  
   get\_all\_items, 96  
   get\_categories, 97  
   is\_dirty, 97  
   KitModel, 92  
   m\_category\_map, 99  
   m\_dirty, 99  
   m\_item\_filter, 100  
   m\_item\_map, 100  
   ModelCategory, 117  
   purge, 97  
   reset, 97  
   set\_dirty, 98  
   show\_all, 98  
   show\_checked\_only, 98  
   show\_unchecked\_only, 99

- KitParser, 100
  - ~KitParser, 102
  - KitParser, 102
  - m\_category, 106
  - m\_cdata, 106
  - m\_item, 106
  - m\_model, 106
  - on\_characters, 102
  - on\_comment, 102
  - on\_end\_document, 103
  - on\_end\_element, 103
  - on\_error, 103
  - on\_fatal\_error, 103
  - on\_start\_document, 104
  - on\_start\_element, 104
  - on\_warning, 104
  - process\_category, 104
  - process\_category\_item, 105
  - process\_item, 105
- KitPrintOperation, 107
  - ~KitPrintOperation, 108
  - create, 108
  - m\_items, 110
  - m\_page\_breaks, 111
  - m\_page\_title, 111
  - m\_ref\_footers, 111
  - m\_ref\_headers, 111
  - m\_ref\_layout, 111
  - new\_footer, 108
  - new\_header, 109
  - on\_begin\_print, 109
  - on\_draw\_page, 109
  - set\_items, 110
  - set\_page\_title, 110
- kitlistdao.hpp
  - item\_choice, 162
- kitlistgui.cpp
  - file\_exists, 164
  - get\_glade\_ref\_ptr, 165
  - load\_resource\_glade\_file, 165
  - type\_children, 164
- kitlistgui.hpp
  - CHECKED\_COL\_POSITION, 166
  - gui\_state, 166
- kitmodel.hpp
  - CategoryMap, 168
  - CategoryMapItr, 168
  - item\_filter\_types, 170
  - ItemMap, 168
  - ItemMapItr, 168
  - ModelCategoryContainer, 169
  - ModelCategoryItr, 169
  - ModelItemContainer, 169
  - ModelItemItr, 169
  - SlotForeachCategory, 169
  - SlotForeachCategoryItr, 169
  - SlotForeachModelItem, 170
  - SlotForeachModelItemItr, 170
- layout\_refptr
  - printing.hpp, 174
- list\_categories
  - KitList, 42
- list\_item
  - KitList, 42
- list\_items
  - KitList, 43
- list\_items\_end
  - KitList, 44
- list\_items\_start
  - KitList, 44
- load
  - YamlConfig, 154
- load\_model
  - Service, 131
- load\_resource\_glade\_file
  - kitlistgui.cpp, 165
- m\_added\_children
  - ModelCategory, 117
- m\_cat\_items\_node
  - XmlDao, 149
- m\_categories\_node
  - XmlDao, 149
- m\_category
  - KitParser, 106
- m\_category\_cols
  - KitListGui, 83
- m\_category\_combo
  - KitListGui, 83
- m\_category\_map
  - KitModel, 99
- m\_cdata
  - KitParser, 106
- m\_changed
  - TickItem, 137
- m\_checkbutton\_add\_item
  - KitListGui, 83
- m\_checked
  - Item, 33
- m\_clipboard\_items
  - KitListGui, 84
- m\_col\_checked
  - ModelItemColumns, 121
- m\_col\_num
  - ModelCategoryColumns, 118
  - ModelItemColumns, 122
- m\_col\_text
  - ModelCategoryColumns, 119
  - ModelItemColumns, 122
- m\_current\_cat\_id
  - KitListGui, 84
- m\_current\_filename
  - YamlConfig, 156
- m\_dao
  - KitList, 48
  - Service, 135
- m\_debug\_log\_filename

- YamlConfig, 156
- m\_deleted
  - GuiState, 29
- m\_desc
  - Item, 33
- m\_dirty
  - GuiState, 29
  - KitModel, 99
- m\_entry\_add\_category
  - KitListGui, 84
- m\_entry\_add\_item
  - KitListGui, 84
- m\_entry\_page\_title
  - KitListGui, 85
- m\_file\_save\_menu\_item
  - KitListGui, 85
- m\_file\_save\_tool\_button
  - KitListGui, 85
- m\_filename
  - KitListGui, 85
  - XmlDao, 149
- m\_id
  - Category, 22
  - Item, 33
- m\_ignore\_list\_events
  - KitListGui, 86
- m\_item
  - KitParser, 106
- m\_item\_cols
  - KitListGui, 86
- m\_item\_filter
  - KitModel, 100
- m\_item\_map
  - KitModel, 100
- m\_item\_tree\_view
  - KitListGui, 86
- m\_items
  - Category, 22
  - KitPrintOperation, 110
- m\_items\_node
  - XmlDao, 150
- m\_kit
  - KitListGui, 86
- m\_max\_category\_id
  - XmlDao, 150
- m\_max\_item\_id
  - XmlDao, 150
- m\_max\_recent\_files
  - YamlConfig, 156
- m\_model
  - FilterItem, 26
  - KitParser, 106
  - Service, 135
- m\_mru\_file\_history
  - YamlConfig, 156
- m\_name
  - Category, 22
- m\_new
  - GuiState, 29
- m\_page\_breaks
  - KitPrintOperation, 111
- m\_page\_title
  - KitListGui, 87
  - KitPrintOperation, 111
  - YamlConfig, 156
- m\_paste\_menu\_item
  - KitListGui, 87
- m\_paste\_tool\_button
  - KitListGui, 87
- m\_recent\_files\_menu\_item
  - KitListGui, 87
- m\_ref\_category\_list\_store
  - KitListGui, 88
- m\_ref\_footers
  - KitPrintOperation, 111
- m\_ref\_headers
  - KitPrintOperation, 111
- m\_ref\_item\_tree\_model
  - KitListGui, 88
- m\_ref\_layout
  - KitPrintOperation, 111
- m\_ref\_page\_setup
  - KitListGui, 88
- m\_ref\_printer\_settings
  - KitListGui, 88
- m\_removed\_children
  - ModelCategory, 117
- m\_service
  - KitListGui, 89
- m\_state
  - KitListGui, 89
- m\_status\_bar
  - KitListGui, 89
- m\_verbose\_flag
  - KitListDao, 58
- m\_window
  - KitListGui, 89
- m\_window\_add\_category
  - KitListGui, 90
- m\_window\_add\_item
  - KitListGui, 90
- m\_window\_preferences
  - KitListGui, 90
- m\_yaml\_config
  - KitListGui, 90
- MAX\_RECENT\_FILES\_CONFIG\_KEY
  - yamlconfig.cpp, 177
- main
  - main.cpp, 171
- main.cpp
  - html\_flag, 172
  - main, 171
  - verbose\_flag, 172
- ModelCategory, 112
  - ~ModelCategory, 113
  - add\_item, 113

- get\_added\_children, 114
- get\_items, 114
- get\_model\_items, 115
- get\_removed\_children, 115
- KitModel, 117
- m\_added\_children, 117
- m\_removed\_children, 117
- ModelCategory, 113
- purge, 115
- remove\_item, 115
- remove\_items, 116
- reset, 116
- ModelCategoryColumns, 118
  - m\_col\_num, 118
  - m\_col\_text, 119
  - ModelCategoryColumns, 118
- ModelCategoryContainer
  - kitmodel.hpp, 169
- ModelCategoryIter
  - kitmodel.hpp, 169
- ModelItem, 119
  - ModelItem, 120
  - ModelItemCompareId, 120
  - set\_checked, 120
- ModelItemColumns, 121
  - m\_col\_checked, 121
  - m\_col\_num, 122
  - m\_col\_text, 122
  - ModelItemColumns, 121
- ModelItemCompareId, 122
  - ModelItem, 120
  - ModelItemCompareId, 123
  - operator(), 123
- ModelItemContainer
  - kitmodel.hpp, 169
- ModelItemIter
  - kitmodel.hpp, 169
- NYI
  - xmldao.hpp, 176
- new\_category
  - KitList, 44
  - KitListDao, 54
  - XmlDao, 145
- new\_footer
  - KitPrintOperation, 108
- new\_header
  - KitPrintOperation, 109
- on\_begin\_print
  - KitPrintOperation, 109
- on\_category\_change
  - KitListGui, 68
- on\_cell\_edit
  - KitListGui, 69
- on\_characters
  - KitParser, 102
- on\_clipboard\_clear
  - KitListGui, 69
- on\_clipboard\_get
  - KitListGui, 69
- on\_clipboard\_received
  - KitListGui, 70
- on\_comment
  - KitParser, 102
- on\_delete\_event
  - KitListGui, 70
- on\_draw\_page
  - KitPrintOperation, 109
- on\_end\_document
  - KitParser, 103
- on\_end\_element
  - KitParser, 103
- on\_error
  - KitParser, 103
- on\_fatal\_error
  - KitParser, 103
- on\_list\_item
  - KitList, 45
- on\_menu\_add
  - KitListGui, 70
- on\_menu\_check\_selected
  - KitListGui, 71
- on\_menu\_copy
  - KitListGui, 71
- on\_menu\_create\_category
  - KitListGui, 71
- on\_menu\_cut
  - KitListGui, 71
- on\_menu\_delete
  - KitListGui, 72
- on\_menu\_delete\_category
  - KitListGui, 72
- on\_menu\_export\_to\_pdf
  - KitListGui, 72
- on\_menu\_file\_new
  - KitListGui, 73
- on\_menu\_file\_open
  - KitListGui, 73
- on\_menu\_help\_about
  - KitListGui, 73
- on\_menu\_paste
  - KitListGui, 74
- on\_menu\_preferences
  - KitListGui, 74
- on\_menu\_print
  - KitListGui, 74
- on\_menu\_quit
  - KitListGui, 74
- on\_menu\_recent\_file
  - KitListGui, 75
- on\_menu\_rename\_category
  - KitListGui, 75
- on\_menu\_save
  - KitListGui, 75
- on\_menu\_save\_as
  - KitListGui, 76

- on\_menu\_select\_all
  - KitListGui, 76
- on\_menu\_show\_all
  - KitListGui, 76
- on\_menu\_show\_checked
  - KitListGui, 77
- on\_menu\_show\_unchecked
  - KitListGui, 77
- on\_menu\_uncheck\_selected
  - KitListGui, 77
- on\_printoperation\_done
  - KitListGui, 77
- on\_printoperation\_status\_changed
  - KitListGui, 78
- on\_row\_changed
  - KitListGui, 78
- on\_start\_document
  - KitParser, 104
- on\_start\_element
  - KitParser, 104
- on\_warning
  - KitParser, 104
- open\_as\_xml
  - Service, 132
- open\_file
  - KitListGui, 78
- operator<<
  - Item, 33
- operator()
  - CategoryCompareId, 23
  - CategoryCompareName, 24
  - FilterItem, 25
  - ItemCompareId, 35
  - ItemCompareName, 36
  - ItemFunctor, 37
  - ModelItemCompareId, 123
  - TickItem, 137
- PAGE\_TITLE\_CONFIG\_KEY
  - ymlconfig.cpp, 177
- PAGE\_TOLERANCE
  - printing.cpp, 174
- PDF\_FILENAME\_EXTENSION
  - anonymous\_namespace{kitlistgui.cpp}, 14
- paste\_from\_xml
  - KitListGui, 79
- paste\_status\_received
  - KitListGui, 79
- printing.cpp
  - BORDER\_SPACING, 173
  - FOOTER\_SPACING, 173
  - FOOTER\_TEXT, 173
  - HEADER\_SPACING, 173
  - PAGE\_TOLERANCE, 174
- printing.hpp
  - layout\_refptr, 174
- process\_category
  - KitParser, 104
- process\_category\_item
  - KitParser, 105
- process\_item
  - KitParser, 105
- purge
  - KitModel, 97
  - ModelCategory, 115
- RECENT\_FILES\_CONFIG\_KEY
  - ymlconfig.cpp, 178
- raise
  - KitListGui, 79
- refresh\_category\_list
  - KitListGui, 80
- refresh\_item\_list
  - KitListGui, 80
- remove\_item
  - Category, 20
  - ModelCategory, 115
- remove\_item\_from\_category
  - KitList, 45
  - KitListDao, 55
  - XmlDao, 145
- remove\_items
  - ModelCategory, 116
- require\_filename
  - KitListDao, 55
  - Service, 132
  - XmlDao, 146
- reset
  - GuiState, 28
  - KitModel, 97
  - ModelCategory, 116
- run
  - KitListGui, 80
- SB\_ITEM\_COUNT
  - anonymous\_namespace{kitlistgui.cpp}, 14
- SB\_MSG
  - anonymous\_namespace{kitlistgui.cpp}, 14
- SB\_PRINT
  - anonymous\_namespace{kitlistgui.cpp}, 15
- SB\_SAVE
  - anonymous\_namespace{kitlistgui.cpp}, 15
- safe\_open\_file
  - KitListGui, 81
- save
  - Service, 132
  - YamlConfig, 154
- save\_as\_xml
  - Service, 132
- save\_model
  - KitListDao, 55
  - XmlDao, 146, 147
- select\_items
  - Service, 133
- selected\_row\_callback
  - KitListGui, 81
- Service, 123
  - ~Service, 125

- copy\_items, 126
- create\_category, 126
- create\_default\_model, 126
- create\_item, 126
- delete\_category, 127
- delete\_item, 127
- filter, 127
- find\_category, 129
- find\_item, 129
- get\_categories, 129
- get\_filtered\_items, 130
- get\_items, 130
- get\_next\_category\_id, 131
- get\_next\_item\_id, 131
- is\_model\_dirty, 131
- load\_model, 131
- m\_dao, 135
- m\_model, 135
- open\_as\_xml, 132
- require\_filename, 132
- save, 132
- save\_as\_xml, 132
- select\_items, 133
- Service, 125
- set\_model\_dirty, 133
- show\_all, 133
- show\_checked\_only, 134
- show\_unchecked\_only, 134
- toggle\_selected\_items, 134
- update\_item, 135
- set\_all\_flags
  - KitList, 45
  - KitListDao, 56
  - XmlDao, 147
- set\_category\_flag
  - KitList, 46
  - KitListDao, 56
  - XmlDao, 147
- set\_checked
  - Item, 32
  - ModelItem, 120
- set\_current\_filename
  - YamlConfig, 155
- set\_deleted
  - GuiState, 28
- set\_description
  - Item, 32
- set\_dirty
  - GuiState, 28
  - KitModel, 98
- set\_filename
  - XmlDao, 147
- set\_id
  - Category, 21
  - Item, 32
- set\_item\_flag
  - KitList, 46
  - KitListDao, 56
- XmlDao, 148
- set\_items
  - KitPrintOperation, 110
- set\_model\_dirty
  - Service, 133
- set\_name
  - Category, 21
- set\_new\_flag
  - GuiState, 28
- set\_page\_title
  - KitListGui, 81
  - KitPrintOperation, 110
  - YamlConfig, 155
- set\_selected
  - KitListGui, 81
- set\_verbose
  - KitListDao, 57
- show\_all
  - KitModel, 98
  - Service, 133
- show\_checked\_only
  - KitModel, 98
  - Service, 134
- show\_unchecked\_only
  - KitModel, 99
  - Service, 134
- SlotForeachCategory
  - kitmodel.hpp, 169
- SlotForeachCategoryIter
  - kitmodel.hpp, 169
- SlotForeachItem
  - item.hpp, 161
- SlotForeachModelItem
  - kitmodel.hpp, 170
- SlotForeachModelItemIter
  - kitmodel.hpp, 170
- tick\_items
  - KitList, 46, 47
- TickItem, 136
  - m\_changed, 137
  - operator(), 137
  - TickItem, 136
- toggle\_selected
  - KitListGui, 82
- toggle\_selected\_items
  - Service, 134
- type\_children
  - kitlistgui.cpp, 164
- unset\_all\_flags
  - KitList, 47
  - KitListDao, 57
  - XmlDao, 148
- unset\_category\_flag
  - KitList, 47
  - KitListDao, 57
  - XmlDao, 148
- unset\_item\_flag

- KitList, [47](#)
- KitListDao, [57](#)
- XmlDao, [148](#)
- update\_item
  - Service, [135](#)
- update\_item\_checked\_state
  - KitListDao, [58](#)
  - XmlDao, [149](#)
- update\_item\_count
  - KitListGui, [82](#)
- update\_paste\_status
  - KitListGui, [82](#)
- update\_recent\_files
  - KitListGui, [82](#)
- update\_recent\_files\_menu
  - KitListGui, [83](#)
- verbose\_flag
  - main.cpp, [172](#)
- XML\_ELEMENT\_ID
  - anonymous\_namespace{kitlistgui.cpp}, [15](#)
- XMLDAO\_H
  - xmldao.hpp, [176](#)
- XmlDao, [137](#)
  - add\_category\_item\_to\_dom, [139](#)
  - add\_category\_to\_dom, [139](#)
  - add\_item, [140](#)
  - add\_item\_to\_dom, [141](#)
  - append\_items\_to\_category, [141](#)
  - associate\_item\_with\_category, [141](#)
  - delete\_category, [142](#)
  - delete\_item, [142](#)
  - get\_all\_items, [143](#)
  - get\_categories, [143](#)
  - get\_category, [143](#)
  - get\_model, [144](#)
  - get\_next\_category\_id, [144](#)
  - get\_next\_item\_id, [145](#)
  - m\_cat\_items\_node, [149](#)
  - m\_categories\_node, [149](#)
  - m\_filename, [149](#)
  - m\_items\_node, [150](#)
  - m\_max\_category\_id, [150](#)
  - m\_max\_item\_id, [150](#)
  - new\_category, [145](#)
  - remove\_item\_from\_category, [145](#)
  - require\_filename, [146](#)
  - save\_model, [146](#), [147](#)
  - set\_all\_flags, [147](#)
  - set\_category\_flag, [147](#)
  - set\_filename, [147](#)
  - set\_item\_flag, [148](#)
  - unset\_all\_flags, [148](#)
  - unset\_category\_flag, [148](#)
  - unset\_item\_flag, [148](#)
  - update\_item\_checked\_state, [149](#)
  - XmlDao, [139](#)
- xmldao.hpp
  - NYI, [176](#)
  - XMLDAO\_H, [176](#)
- YamlConfig, [150](#)
  - ~YamlConfig, [152](#)
  - add\_recent\_filename, [152](#)
  - get\_config\_filename, [152](#)
  - get\_current\_filename, [153](#)
  - get\_debug\_log\_filename, [153](#)
  - get\_page\_title, [153](#)
  - get\_recent\_filenames, [154](#)
  - load, [154](#)
  - m\_current\_filename, [156](#)
  - m\_debug\_log\_filename, [156](#)
  - m\_max\_recent\_files, [156](#)
  - m\_mru\_file\_history, [156](#)
  - m\_page\_title, [156](#)
  - save, [154](#)
  - set\_current\_filename, [155](#)
  - set\_page\_title, [155](#)
  - YamlConfig, [152](#)
- yamlconfig.cpp
  - CONFIG\_FILENAME, [177](#)
  - CURRENT\_FILENAME\_CONFIG\_KEY, [177](#)
  - DEBUG\_LOG\_FILENAME\_CONFIG\_KEY, [177](#)
  - MAX\_RECENT\_FILES\_CONFIG\_KEY, [177](#)
  - PAGE\_TITLE\_CONFIG\_KEY, [177](#)
  - RECENT\_FILES\_CONFIG\_KEY, [178](#)
- yamlconfig.hpp
  - DEFAULT\_MAX\_RECENT\_FILES, [178](#)
  - DEFAULT\_PAGE\_TITLE, [179](#)