

Kitlist
0.7.0

Generated by Doxygen 1.5.6

Mon May 31 18:56:40 2010

Contents

1	Kitlist Documentation	1
1.1	Introduction	1
1.2	Additional Documentation	1
2	Namespace Index	3
2.1	Namespace List	3
3	Class Index	5
3.1	Class Hierarchy	5
4	Class Index	7
4.1	Class List	7
5	File Index	9
5.1	File List	9
6	Namespace Documentation	11
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	DEFAULT_MAX_RECENT_FILES	12
6.1.1.4	GCONF_KEY	13
6.1.1.5	GCONF_KEY_CURRENT_FILENAME	13
6.1.1.6	GCONF_KEY_MAX_RECENT_FILES	13
6.1.1.7	GCONF_KEY_PAGE_TITLE	13
6.1.1.8	GCONF_KEY_RECENT_FILES	13
6.1.1.9	GLADE_APP_FILE	13
6.1.1.10	item_target_custom	14
6.1.1.11	item_target_text	14

6.1.1.12	PDF_FILENAME_EXTENSION	14
6.1.1.13	SB_ITEM_COUNT	14
6.1.1.14	SB_MSG	14
6.1.1.15	SB_PRINT	14
6.1.1.16	SB_SAVE	15
6.1.1.17	XML_ELEMENT_ID	15
7	Class Documentation	17
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	set_id	18
7.1.3.2	get_id	19
7.1.3.3	set_name	19
7.1.3.4	get_name	19
7.1.3.5	add_item	19
7.1.3.6	remove_item	19
7.1.3.7	item_count	20
7.1.3.8	has_items	20
7.1.3.9	foreach_item	20
7.1.3.10	execute	20
7.1.4	Friends And Related Function Documentation	20
7.1.4.1	CategoryCompareName	20
7.1.4.2	CategoryCompareId	20
7.1.4.3	KitModel	21
7.1.5	Member Data Documentation	21
7.1.5.1	m_id	21
7.1.5.2	m_name	21
7.1.5.3	m_items	21
7.2	CategoryCompareId Class Reference	22
7.2.1	Detailed Description	22
7.2.2	Constructor & Destructor Documentation	22
7.2.2.1	CategoryCompareId	22
7.2.3	Member Function Documentation	22
7.2.3.1	operator()	22

7.3	CategoryCompareName Class Reference	23
7.3.1	Detailed Description	23
7.3.2	Constructor & Destructor Documentation	23
7.3.2.1	CategoryCompareName	23
7.3.3	Member Function Documentation	23
7.3.3.1	operator()	23
7.4	FilterItem Class Reference	24
7.4.1	Detailed Description	24
7.4.2	Constructor & Destructor Documentation	24
7.4.2.1	FilterItem	24
7.4.3	Member Function Documentation	24
7.4.3.1	operator()	24
7.4.4	Member Data Documentation	24
7.4.4.1	m_model	24
7.5	GuiState Class Reference	25
7.5.1	Detailed Description	25
7.5.2	Constructor & Destructor Documentation	25
7.5.2.1	GuiState	25
7.5.3	Member Function Documentation	26
7.5.3.1	is_dirty	26
7.5.3.2	set_dirty	26
7.5.3.3	is_deleted	26
7.5.3.4	set_deleted	26
7.5.3.5	set_new_flag	26
7.5.3.6	is_new	26
7.5.3.7	reset	26
7.5.4	Member Data Documentation	27
7.5.4.1	m_dirty	27
7.5.4.2	m_deleted	27
7.5.4.3	m_new	27
7.6	Item Class Reference	28
7.6.1	Detailed Description	28
7.6.2	Constructor & Destructor Documentation	29
7.6.2.1	Item	29
7.6.2.2	Item	29
7.6.3	Member Function Documentation	29

7.6.3.1	set_id	29
7.6.3.2	get_id	29
7.6.3.3	set_description	29
7.6.3.4	get_description	29
7.6.3.5	set_checked	29
7.6.3.6	get_checked	30
7.6.4	Friends And Related Function Documentation	30
7.6.4.1	ItemCompareName	30
7.6.4.2	ItemCompareId	30
7.6.4.3	operator<<	30
7.6.5	Member Data Documentation	30
7.6.5.1	m_id	30
7.6.5.2	m_desc	30
7.6.5.3	m_checked	30
7.7	ItemCompareId Class Reference	32
7.7.1	Detailed Description	32
7.7.2	Constructor & Destructor Documentation	32
7.7.2.1	ItemCompareId	32
7.7.3	Member Function Documentation	32
7.7.3.1	operator()	32
7.8	ItemCompareName Class Reference	33
7.8.1	Detailed Description	33
7.8.2	Constructor & Destructor Documentation	33
7.8.2.1	ItemCompareName	33
7.8.3	Member Function Documentation	33
7.8.3.1	operator()	33
7.9	ItemFunctor Class Reference	34
7.9.1	Detailed Description	34
7.9.2	Member Function Documentation	34
7.9.2.1	operator()	34
7.10	KitList Class Reference	35
7.10.1	Detailed Description	36
7.10.2	Constructor & Destructor Documentation	36
7.10.2.1	KitList	36
7.10.2.2	~KitList	36
7.10.3	Member Function Documentation	36

7.10.3.1	get_dao	36
7.10.3.2	add_item	37
7.10.3.3	add_item	37
7.10.3.4	append_items_to_category	37
7.10.3.5	associate_item_with_category	37
7.10.3.6	list_items_start	38
7.10.3.7	list_item	38
7.10.3.8	list_items_end	38
7.10.3.9	list_items	38
7.10.3.10	on_list_item	39
7.10.3.11	list_items	39
7.10.3.12	list_items	39
7.10.3.13	execute	39
7.10.3.14	tick_items	39
7.10.3.15	tick_items	39
7.10.3.16	tick_items	40
7.10.3.17	list_categories	40
7.10.3.18	new_category	40
7.10.3.19	delete_category	40
7.10.3.20	delete_item	41
7.10.3.21	remove_item_from_category	41
7.10.3.22	set_item_flag	41
7.10.3.23	unset_item_flag	41
7.10.3.24	set_category_flag	41
7.10.3.25	unset_category_flag	41
7.10.3.26	set_all_flags	42
7.10.3.27	unset_all_flags	42
7.10.4	Member Data Documentation	42
7.10.4.1	m_dao	42
7.11	KitListDao Class Reference	43
7.11.1	Detailed Description	44
7.11.2	Constructor & Destructor Documentation	44
7.11.2.1	KitListDao	44
7.11.2.2	~KitListDao	44
7.11.3	Member Function Documentation	44
7.11.3.1	get_model	44

7.11.3.2	save_model	45
7.11.3.3	set_verbose	45
7.11.3.4	is_verbose	45
7.11.3.5	get_category	45
7.11.3.6	get_all_items	45
7.11.3.7	add_item	46
7.11.3.8	add_item	46
7.11.3.9	append_items_to_category	46
7.11.3.10	associate_item_with_category	46
7.11.3.11	get_categories	47
7.11.3.12	new_category	47
7.11.3.13	delete_item	47
7.11.3.14	update_item_checked_state	47
7.11.3.15	remove_item_from_category	47
7.11.3.16	get_next_item_id	48
7.11.3.17	get_next_category_id	48
7.11.3.18	delete_category	48
7.11.3.19	set_item_flag	48
7.11.3.20	unset_item_flag	48
7.11.3.21	set_category_flag	48
7.11.3.22	unset_category_flag	49
7.11.3.23	set_all_flags	49
7.11.3.24	unset_all_flags	49
7.11.3.25	require_filename	49
7.11.4	Member Data Documentation	49
7.11.4.1	m_verbose_flag	49
7.12	KitListGui Class Reference	50
7.12.1	Detailed Description	55
7.12.2	Constructor & Destructor Documentation	55
7.12.2.1	KitListGui	55
7.12.2.2	~KitListGui	55
7.12.3	Member Function Documentation	55
7.12.3.1	init	55
7.12.3.2	get_max_recent_files	56
7.12.3.3	get_selected_items	56
7.12.3.4	add_items	56

7.12.3.5	set_page_title	56
7.12.3.6	close_preferences_window	57
7.12.3.7	cancel_preferences_window	57
7.12.3.8	close_add_item_window	57
7.12.3.9	cancel_add_item_window	57
7.12.3.10	close_add_category_window	57
7.12.3.11	cancel_add_category_window	58
7.12.3.12	get_selected_category	58
7.12.3.13	init_add_item_window	58
7.12.3.14	delete_selected_items	58
7.12.3.15	copy_selected_items_to_clipboard	58
7.12.3.16	confirm_lose_changes	59
7.12.3.17	update_recent_files_menu	59
7.12.3.18	update_recent_files	59
7.12.3.19	on_delete_event	59
7.12.3.20	on_menu_quit	59
7.12.3.21	on_menu_file_new	60
7.12.3.22	on_menu_file_open	60
7.12.3.23	on_menu_save	60
7.12.3.24	on_menu_save_as	60
7.12.3.25	on_printoperation_done	61
7.12.3.26	on_printoperation_status_changed	61
7.12.3.27	on_menu_print	61
7.12.3.28	on_menu_export_to_pdf	61
7.12.3.29	on_menu_recent_file	61
7.12.3.30	on_menu_preferences	62
7.12.3.31	on_menu_add	62
7.12.3.32	on_menu_delete	62
7.12.3.33	on_menu_cut	62
7.12.3.34	on_menu_copy	62
7.12.3.35	on_menu_paste	63
7.12.3.36	on_menu_show_all	63
7.12.3.37	on_menu_show_checked	63
7.12.3.38	on_menu_show_unchecked	63
7.12.3.39	on_menu_select_all	63
7.12.3.40	on_menu_check_selected	63

7.12.3.41	<code>on_menu_uncheck_selected</code>	64
7.12.3.42	<code>on_menu_create_category</code>	64
7.12.3.43	<code>on_menu_delete_category</code>	64
7.12.3.44	<code>on_menu_rename_category</code>	64
7.12.3.45	<code>on_menu_help_about</code>	64
7.12.3.46	<code>on_menu_help_contents</code>	65
7.12.3.47	<code>on_clipboard_get</code>	65
7.12.3.48	<code>on_clipboard_clear</code>	65
7.12.3.49	<code>on_clipboard_received</code>	65
7.12.3.50	<code>on_category_change</code>	65
7.12.3.51	<code>on_cell_edit</code>	66
7.12.3.52	<code>choose_filename</code>	66
7.12.3.53	<code>choose_pdf_filename</code>	66
7.12.3.54	<code>update_paste_status</code>	67
7.12.3.55	<code>paste_status_received</code>	67
7.12.3.56	<code>paste_from_xml</code>	67
7.12.3.57	<code>refresh_item_list</code>	67
7.12.3.58	<code>refresh_category_list</code>	67
7.12.3.59	<code>selected_row_callback</code>	68
7.12.3.60	<code>set_selected</code>	68
7.12.3.61	<code>toggle_selected</code>	68
7.12.3.62	<code>on_row_changed</code>	68
7.12.3.63	<code>update_item_count</code>	69
7.12.3.64	<code>open_file</code>	69
7.12.3.65	<code>raise</code>	69
7.12.3.66	<code>safe_open_file</code>	69
7.12.3.67	<code>run</code>	69
7.12.4	Member Data Documentation	70
7.12.4.1	<code>m_filename</code>	70
7.12.4.2	<code>m_page_title</code>	70
7.12.4.3	<code>m_clipboard_items</code>	70
7.12.4.4	<code>m_ignore_list_events</code>	70
7.12.4.5	<code>m_kit</code>	70
7.12.4.6	<code>m_window</code>	70
7.12.4.7	<code>m_window_preferences</code>	71
7.12.4.8	<code>m_entry_page_title</code>	71

7.12.4.9	m_window_add_item	71
7.12.4.10	m_window_add_category	71
7.12.4.11	m_entry_add_item	71
7.12.4.12	m_entry_add_category	71
7.12.4.13	m_file_save_menu_item	72
7.12.4.14	m_file_save_tool_button	72
7.12.4.15	m_recent_files_menu_item	72
7.12.4.16	m_paste_menu_item	72
7.12.4.17	m_paste_tool_button	72
7.12.4.18	m_checkbutton_add_item	72
7.12.4.19	m_category_combo	72
7.12.4.20	m_ref_category_list_store	73
7.12.4.21	m_category_cols	73
7.12.4.22	m_item_tree_view	73
7.12.4.23	m_item_cols	73
7.12.4.24	m_service	73
7.12.4.25	m_ref_item_tree_model	73
7.12.4.26	m_status_bar	74
7.12.4.27	m_ref_page_setup	74
7.12.4.28	m_ref_printer_settings	74
7.12.4.29	m_state	74
7.12.4.30	m_current_cat_id	74
7.13	KitModel Class Reference	75
7.13.1	Detailed Description	76
7.13.2	Constructor & Destructor Documentation	77
7.13.2.1	KitModel	77
7.13.2.2	~KitModel	77
7.13.3	Member Function Documentation	77
7.13.3.1	foreach_item_iter	77
7.13.3.2	foreach_item	77
7.13.3.3	foreach_category_iter	77
7.13.3.4	foreach_category	77
7.13.3.5	find_category	78
7.13.3.6	find_item	78
7.13.3.7	get_categories	78
7.13.3.8	get_all_items	78

7.13.3.9	get_all_items	78
7.13.3.10	add_category	78
7.13.3.11	add_item	79
7.13.3.12	add_item	79
7.13.3.13	copy_items	79
7.13.3.14	filter	79
7.13.3.15	set_dirty	80
7.13.3.16	is_dirty	80
7.13.3.17	show_all	80
7.13.3.18	show_checked_only	80
7.13.3.19	show_unchecked_only	80
7.13.3.20	reset	80
7.13.3.21	purge	81
7.13.4	Member Data Documentation	81
7.13.4.1	m_dirty	81
7.13.4.2	m_category_map	81
7.13.4.3	m_item_map	81
7.13.4.4	m_item_filter	81
7.14	KitParser Class Reference	83
7.14.1	Detailed Description	84
7.14.2	Constructor & Destructor Documentation	84
7.14.2.1	KitParser	84
7.14.2.2	~KitParser	84
7.14.3	Member Function Documentation	84
7.14.3.1	process_item	84
7.14.3.2	process_category	85
7.14.3.3	process_category_item	85
7.14.3.4	on_start_document	85
7.14.3.5	on_end_document	85
7.14.3.6	on_start_element	85
7.14.3.7	on_end_element	86
7.14.3.8	on_characters	86
7.14.3.9	on_comment	86
7.14.3.10	on_warning	86
7.14.3.11	on_error	86
7.14.3.12	on_fatal_error	86

7.14.4	Member Data Documentation	87
7.14.4.1	m_category	87
7.14.4.2	m_item	87
7.14.4.3	m_cdata	87
7.14.4.4	m_model	87
7.15	KitPrintOperation Class Reference	88
7.15.1	Detailed Description	89
7.15.2	Constructor & Destructor Documentation	89
7.15.2.1	~KitPrintOperation	89
7.15.3	Member Function Documentation	89
7.15.3.1	create	89
7.15.3.2	set_items	89
7.15.3.3	set_page_title	89
7.15.3.4	new_header	89
7.15.3.5	new_footer	90
7.15.3.6	on_begin_print	90
7.15.3.7	on_draw_page	90
7.15.4	Member Data Documentation	90
7.15.4.1	m_items	90
7.15.4.2	m_page_title	91
7.15.4.3	m_ref_layout	91
7.15.4.4	m_page_breaks	91
7.15.4.5	m_ref_headers	91
7.15.4.6	m_ref_footers	91
7.16	ModelCategory Class Reference	92
7.16.1	Detailed Description	93
7.16.2	Constructor & Destructor Documentation	93
7.16.2.1	ModelCategory	93
7.16.2.2	~ModelCategory	93
7.16.3	Member Function Documentation	93
7.16.3.1	get_model_items	93
7.16.3.2	get_items	93
7.16.3.3	get_items	94
7.16.3.4	get_removed_children	94
7.16.3.5	get_added_children	94
7.16.3.6	add_item	94

7.16.3.7	remove_item	94
7.16.3.8	remove_items	95
7.16.3.9	reset	95
7.16.3.10	purge	95
7.16.4	Friends And Related Function Documentation	95
7.16.4.1	KitModel	95
7.16.5	Member Data Documentation	95
7.16.5.1	m_removed_children	95
7.16.5.2	m_added_children	95
7.17	ModelCategoryColumns Class Reference	97
7.17.1	Detailed Description	97
7.17.2	Constructor & Destructor Documentation	97
7.17.2.1	ModelCategoryColumns	97
7.17.3	Member Data Documentation	97
7.17.3.1	m_col_text	97
7.17.3.2	m_col_num	97
7.18	ModelItem Class Reference	98
7.18.1	Detailed Description	98
7.18.2	Constructor & Destructor Documentation	98
7.18.2.1	ModelItem	98
7.18.3	Member Function Documentation	98
7.18.3.1	set_checked	98
7.18.4	Friends And Related Function Documentation	99
7.18.4.1	ModelItemCompareId	99
7.19	ModelItemColumns Class Reference	100
7.19.1	Detailed Description	100
7.19.2	Constructor & Destructor Documentation	100
7.19.2.1	ModelItemColumns	100
7.19.3	Member Data Documentation	100
7.19.3.1	m_col_text	100
7.19.3.2	m_col_checked	100
7.19.3.3	m_col_num	100
7.20	ModelItemCompareId Class Reference	102
7.20.1	Detailed Description	102
7.20.2	Constructor & Destructor Documentation	102
7.20.2.1	ModelItemCompareId	102

7.20.3	Member Function Documentation	102
7.20.3.1	operator()	102
7.21	Service Class Reference	103
7.21.1	Detailed Description	104
7.21.2	Constructor & Destructor Documentation	105
7.21.2.1	Service	105
7.21.2.2	~Service	105
7.21.3	Member Function Documentation	105
7.21.3.1	load_model	105
7.21.3.2	get_next_item_id	105
7.21.3.3	get_next_category_id	105
7.21.3.4	find_item	105
7.21.3.5	find_category	106
7.21.3.6	copy_items	106
7.21.3.7	create_item	106
7.21.3.8	delete_item	106
7.21.3.9	delete_category	106
7.21.3.10	create_category	107
7.21.3.11	is_model_dirty	107
7.21.3.12	set_model_dirty	107
7.21.3.13	filter	107
7.21.3.14	create_default_model	108
7.21.3.15	open_as_xml	108
7.21.3.16	save	108
7.21.3.17	save_as_xml	108
7.21.3.18	update_item	109
7.21.3.19	get_items	109
7.21.3.20	get_filtered_items	109
7.21.3.21	get_categories	110
7.21.3.22	show_all	110
7.21.3.23	show_checked_only	110
7.21.3.24	show_unchecked_only	110
7.21.3.25	select_items	110
7.21.3.26	toggle_selected_items	111
7.21.3.27	require_filename	111
7.21.4	Member Data Documentation	111

7.21.4.1	m_dao	111
7.21.4.2	m_model	111
7.22	TickItem Class Reference	112
7.22.1	Detailed Description	112
7.22.2	Constructor & Destructor Documentation	112
7.22.2.1	TickItem	112
7.22.3	Member Function Documentation	112
7.22.3.1	operator()	112
7.22.4	Member Data Documentation	112
7.22.4.1	m_changed	112
7.23	XmlDao Class Reference	114
7.23.1	Detailed Description	116
7.23.2	Constructor & Destructor Documentation	116
7.23.2.1	XmlDao	116
7.23.3	Member Function Documentation	116
7.23.3.1	add_item_to_dom	116
7.23.3.2	add_category_item_to_dom	116
7.23.3.3	add_category_to_dom	116
7.23.3.4	get_model	117
7.23.3.5	get_model	117
7.23.3.6	save_model	117
7.23.3.7	save_model	117
7.23.3.8	get_category	118
7.23.3.9	get_all_items	118
7.23.3.10	add_item	118
7.23.3.11	add_item	118
7.23.3.12	append_items_to_category	119
7.23.3.13	associate_item_with_category	119
7.23.3.14	get_categories	119
7.23.3.15	new_category	119
7.23.3.16	delete_item	120
7.23.3.17	update_item_checked_state	120
7.23.3.18	remove_item_from_category	120
7.23.3.19	get_next_item_id	120
7.23.3.20	get_next_category_id	121
7.23.3.21	delete_category	121

7.23.3.22	set_item_flag	121
7.23.3.23	unset_item_flag	121
7.23.3.24	set_category_flag	121
7.23.3.25	unset_category_flag	121
7.23.3.26	set_all_flags	122
7.23.3.27	unset_all_flags	122
7.23.3.28	set_filename	122
7.23.3.29	require_filename	122
7.23.4	Member Data Documentation	122
7.23.4.1	m_filename	122
7.23.4.2	m_items_node	123
7.23.4.3	m_categories_node	123
7.23.4.4	m_cat_items_node	123
7.23.4.5	m_max_item_id	123
7.23.4.6	m_max_category_id	123
8	File Documentation	125
8.1	/home/frank/projects/kitlist/src/category.cpp File Reference	125
8.2	/home/frank/projects/kitlist/src/category.hpp File Reference	126
8.2.1	Define Documentation	126
8.2.1.1	CATEGORY_H	126
8.2.2	Typedef Documentation	126
8.2.2.1	CategoryContainer	126
8.2.2.2	CategoryIter	126
8.3	/home/frank/projects/kitlist/src/item.hpp File Reference	127
8.3.1	Define Documentation	127
8.3.1.1	ITEM_H	127
8.3.2	Typedef Documentation	127
8.3.2.1	ItemContainer	127
8.3.2.2	ItemIter	128
8.3.2.3	ItemList	128
8.3.2.4	ItemListIter	128
8.3.2.5	SlotForeachItem	128
8.4	/home/frank/projects/kitlist/src/kitlistdao.hpp File Reference	129
8.4.1	Define Documentation	129
8.4.1.1	KIT_LIST_DAO_H	129
8.4.2	Enumeration Type Documentation	129

8.4.2.1	item_choice	129
8.5	/home/frank/projects/kitlist/src/kitlistgui.cpp File Reference	130
8.5.1	Define Documentation	132
8.5.1.1	KITLIST_SERVICE_IFACE	132
8.5.1.2	KITLIST_SERVICE_NAME	132
8.5.1.3	KITLIST_SERVICE_OBJECT	132
8.5.2	Typedef Documentation	133
8.5.2.1	type_children	133
8.5.3	Function Documentation	133
8.5.3.1	file_exists	133
8.5.3.2	get_glade_ref_ptr	133
8.5.3.3	load_resource_glade_file	133
8.6	/home/frank/projects/kitlist/src/kitlistgui.hpp File Reference	134
8.6.1	Define Documentation	135
8.6.1.1	KIT_LIST_GUI_H	135
8.6.2	Enumeration Type Documentation	135
8.6.2.1	gui_state	135
8.6.3	Variable Documentation	135
8.6.3.1	CHECKED_COL_POSITION	135
8.7	/home/frank/projects/kitlist/src/kitlistpgsqldao.cpp File Reference	136
8.8	/home/frank/projects/kitlist/src/kitlistpgsqldao.hpp File Reference	137
8.8.1	Define Documentation	137
8.8.1.1	KIT_LIST_PGSQL_DAO_H	137
8.9	/home/frank/projects/kitlist/src/kitmodel.cpp File Reference	138
8.10	/home/frank/projects/kitlist/src/kitmodel.hpp File Reference	139
8.10.1	Define Documentation	140
8.10.1.1	KIT_MODEL_H	140
8.10.2	Typedef Documentation	140
8.10.2.1	CategoryMap	140
8.10.2.2	CategoryMapIter	140
8.10.2.3	ItemMap	140
8.10.2.4	ItemMapIter	140
8.10.2.5	ModelCategoryContainer	140
8.10.2.6	ModelCategoryIter	140
8.10.2.7	ModelItemContainer	140
8.10.2.8	ModelItemIter	140

8.10.2.9	SlotForeachCategory	140
8.10.2.10	SlotForeachCategoryIter	140
8.10.2.11	SlotForeachModelItem	141
8.10.2.12	SlotForeachModelItemIter	141
8.10.3	Enumeration Type Documentation	141
8.10.3.1	item_filter_types	141
8.11	/home/frank/projects/kitlist/src/kitparser.cpp File Reference	142
8.12	/home/frank/projects/kitlist/src/kitparser.hpp File Reference	143
8.12.1	Define Documentation	143
8.12.1.1	KIT_PARSER_H	143
8.13	/home/frank/projects/kitlist/src/main.cpp File Reference	144
8.13.1	Function Documentation	144
8.13.1.1	main	144
8.13.2	Variable Documentation	145
8.13.2.1	html_flag	145
8.13.2.2	verbose_flag	145
8.14	/home/frank/projects/kitlist/src/printing.cpp File Reference	146
8.14.1	Variable Documentation	146
8.14.1.1	BORDER_SPACING	146
8.14.1.2	FOOTER_SPACING	146
8.14.1.3	FOOTER_TEXT	146
8.14.1.4	HEADER_SPACING	147
8.14.1.5	PAGE_TOLERANCE	147
8.15	/home/frank/projects/kitlist/src/printing.hpp File Reference	148
8.15.1	Typedef Documentation	148
8.15.1.1	layout_refptr	148
8.16	/home/frank/projects/kitlist/src/service.cpp File Reference	149
8.17	/home/frank/projects/kitlist/src/service.hpp File Reference	150
8.17.1	Define Documentation	150
8.17.1.1	SERVICE_H	150
8.18	/home/frank/projects/kitlist/src/xmldao.cpp File Reference	151
8.19	/home/frank/projects/kitlist/src/xmldao.hpp File Reference	152
8.19.1	Define Documentation	152
8.19.1.1	NYI	152
8.19.1.2	XMLDAO_H	152

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 5.0 (Lenny)
- Maemo (Nokia Tablet)
- Windows XP

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

Class Index

3.1 Class Hierarchy

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

Category	17
ModelCategory	92
CategoryCompareId	22
CategoryCompareName	23
GuiState	25
ModelCategory	92
ModelItem	98
Item	28
ModelItem	98
ItemCompareId	32
ItemCompareName	33
ItemFunctor	34
FilterItem	24
TickItem	112
KitList	35
KitListDao	43
XmlDao	114
KitListGui	50
KitModel	75
KitParser	83
KitPrintOperation	88
ModelCategoryColumns	97
ModelItemColumns	100
ModelItemCompareId	102
Service	103

Chapter 4

Class Index

4.1 Class List

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

Category (Represents a Category)	17
CategoryCompareId (Comparator used for comparing Categories by id)	22
CategoryCompareName (Comparator used for sorting Categories by name)	23
FilterItem	24
GuiState (Class encapsulating state of an object in the data model)	25
Item (Represents an Item)	28
ItemCompareId (Comparator used for comparing Items by id)	32
ItemCompareName (Comparator used for sorting Items by name)	33
ItemFunctor (Functor for processing items)	34
KitList (Main application class)	35
KitListDao (Defines the methods that an implementation of this class must implement)	43
KitListGui (Encapsulates the methods for the application's GUI front end)	50
KitModel (Holds a rich graph of objects representing the application's data model)	75
KitParser (SaxParser implementation for reading the KitModel from an XML document)	83
KitPrintOperation (Prints the kitlist)	88
ModelCategory (Represents a Category combined with GuiState attributes)	92
ModelCategoryColumns (A definition for displaying a ModelCategory in a combo box)	97
ModelItem (Represents an Item combined with GuiState attributes)	98
ModelItemColumns (A definition for displaying an item in a multi-column list)	100
ModelItemCompareId (Comparator for comparing items by their unique ID)	102
Service (Business/service layer implementation)	103
TickItem	112
XmlDao (Implementation of a KitListDao using XML as the persistence store)	114

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

/home/frank/projects/kitlist/src/category.cpp	125
/home/frank/projects/kitlist/src/category.hpp	126
/home/frank/projects/kitlist/src/item.hpp	127
/home/frank/projects/kitlist/src/kitlistdao.hpp	129
/home/frank/projects/kitlist/src/kitlistgui.cpp	130
/home/frank/projects/kitlist/src/kitlistgui.hpp	134
/home/frank/projects/kitlist/src/kitlistpgsqldao.cpp	136
/home/frank/projects/kitlist/src/kitlistpgsqldao.hpp	137
/home/frank/projects/kitlist/src/kitmodel.cpp	138
/home/frank/projects/kitlist/src/kitmodel.hpp	139
/home/frank/projects/kitlist/src/kitparser.cpp	142
/home/frank/projects/kitlist/src/kitparser.hpp	143
/home/frank/projects/kitlist/src/main.cpp	144
/home/frank/projects/kitlist/src/printing.cpp	146
/home/frank/projects/kitlist/src/printing.hpp	148
/home/frank/projects/kitlist/src/service.cpp	149
/home/frank/projects/kitlist/src/service.hpp	150
/home/frank/projects/kitlist/src/xmldao.cpp	151
/home/frank/projects/kitlist/src/xmldao.hpp	152

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 gint DEFAULT_MAX_RECENT_FILES = 4`
The maximum number of recent files to maintain.
- `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 `const Glib::ustring anonymous_namespace{kitlistgui.cpp}::DEFAULT_FILENAME = "kitlist" + DEFAULT_FILENAME_EXTENSION`

The default filename.

Definition at line 94 of file `kitlistgui.cpp`.

Referenced by `KitListGui::init()`.

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

Default filename extension.

Definition at line 88 of file `kitlistgui.cpp`.

Referenced by `KitListGui::choose_filename()`, and `KitListGui::on_menu_file_open()`.

6.1.1.3 `const gint anonymous_namespace{kitlistgui.cpp}::DEFAULT_MAX_RECENT_FILES = 4`

The maximum number of recent files to maintain.

Definition at line 117 of file `kitlistgui.cpp`.

Referenced by `KitListGui::get_max_recent_files()`.

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

The application's root key in the GConf hierarchy.

Definition at line 100 of file kitlistgui.cpp.

Referenced by KitListGui::init().

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

GConf entry for the current filename.

Definition at line 104 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().

6.1.1.6 `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 123 of file kitlistgui.cpp.

Referenced by KitListGui::get_max_recent_files().

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

GConf entry for the page title.

Definition at line 107 of file kitlistgui.cpp.

Referenced by KitListGui::init(), and KitListGui::set_page_title().

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

GConf entry for recent files.

Definition at line 120 of file kitlistgui.cpp.

Referenced by KitListGui::update_recent_files(), and KitListGui::update_recent_files_menu().

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

Resource file name.

Definition at line 67 of file kitlistgui.cpp.

Referenced by KitListGui::init().

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

Key used for custom clipboard.

Definition at line 80 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().

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

Mime type for clipboard content.

Definition at line 82 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().

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

PDF filename extension.

Definition at line 91 of file kitlistgui.cpp.

Referenced by KitListGui::choose_pdf_filename().

6.1.1.13 const guint anonymous_namespace{kitlistgui.cpp}::SB_ITEM_COUNT = 1000

Status bar message constant for displaying item counts.

Definition at line 70 of file kitlistgui.cpp.

Referenced by KitListGui::update_item_count().

6.1.1.14 const guint anonymous_namespace{kitlistgui.cpp}::SB_MSG = SB_SAVE + 1

Status bar message constant for general messages.

Definition at line 74 of file kitlistgui.cpp.

Referenced by KitListGui::on_menu_cut(), KitListGui::on_menu_delete_category(), and KitListGui::on_menu_rename_category().

6.1.1.15 const guint anonymous_namespace{kitlistgui.cpp}::SB_PRINT = SB_MSG + 1

Status bar message constant for printer messages.

Definition at line 77 of file kitlistgui.cpp.

Referenced by KitListGui::on_printoperation_status_changed().

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

Status bar message constant for save notifications.

Definition at line 72 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().

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

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

Definition at line 85 of file kitlistgui.cpp.

Referenced by KitListGui::copy_selected_items_to_clipboard(), and KitListGui::paste_from_xml().

Chapter 7

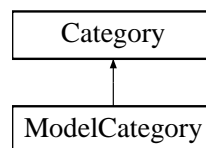
Class Documentation

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](#) ([ItemFunctor](#) &functor)

Executes the passed [ItemFunctor](#).

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](#) () [\[inline\]](#)

Definition at line 43 of file [category.hpp](#).

7.1.3 Member Function Documentation

7.1.3.1 void [Category::set_id](#) (long *id*) [\[inline\]](#)

Definition at line 44 of file [category.hpp](#).

References [m_id](#).

Referenced by [Service::create_category\(\)](#), and [KitParser::process_category\(\)](#).

7.1.3.2 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()`, `KitListGui::on_menu_create_category()`, `KitListGui::on_menu_rename_category()`, `KitParser::process_category_item()`, and `KitListGui::refresh_category_list()`.

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

Definition at line 46 of file category.hpp.

References `m_name`.

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

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

Definition at line 47 of file category.hpp.

References `m_name`.

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

7.1.3.5 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()`.

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

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

The passed item is not deleted.

Parameters:

item the item to un-associate.

Reimplemented in [ModelCategory](#).

Definition at line 40 of file category.cpp.

References `m_items`.

Referenced by `ModelCategory::remove_item()`.

7.1.3.7 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.

References m_items.

Referenced by KitList::list_items().

7.1.3.8 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 m_items.

Referenced by KitList::list_items().

7.1.3.9 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:

slot the callback function.

Definition at line 55 of file category.cpp.

References m_items.

Referenced by XmlDao::add_category_to_dom(), and KitList::list_items().

7.1.3.10 void Category::execute (ItemFunctor & functor)

Executes the passed [ItemFunctor](#).

The ItemFunctor'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 KitList::tick_items().

7.1.4 Friends And Related Function Documentation

7.1.4.1 friend class CategoryCompareName [friend]

Definition at line 56 of file category.hpp.

7.1.4.2 friend class CategoryCompareId [friend]

Definition at line 57 of file category.hpp.

7.1.4.3 friend class `KitModel` [friend]

Reimplemented in [ModelCategory](#).

Definition at line 58 of file `category.hpp`.

7.1.5 Member Data Documentation

7.1.5.1 long `Category::m_id` [protected]

Unique id.

Definition at line 39 of file `category.hpp`.

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

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

The category name.

Definition at line 40 of file `category.hpp`.

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

7.1.5.3 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()`, `has_items()`, `item_count()`, `ModelCategory::purge()`, and `remove_item()`.

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 ()` [inline]

Definition at line 79 of file category.hpp.

7.2.3 Member Function Documentation

7.2.3.1 `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 ()` [inline]

Definition at line 68 of file category.hpp.

7.3.3 Member Function Documentation

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

Definition at line 69 of file category.hpp.

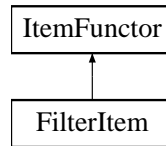
References `Category::m_name`.

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 (KitModel & model)` [inline]

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

7.4.3 Member Function Documentation

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

Implements [ItemFuncator](#).

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

References [KitModel::filter\(\)](#), [Item::get_checked\(\)](#), and [m_model](#).

7.4.4 Member Data Documentation

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

Definition at line 31 of file `service.cpp`.

Referenced by `operator()`.

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

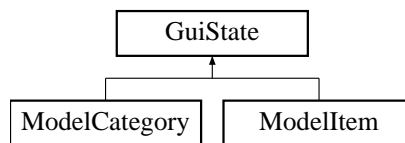
- `/home/frank/projects/kitlist/src/service.cpp`

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 () [inline]

Definition at line 44 of file kitmodel.hpp.

7.5.3 Member Function Documentation

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

Definition at line 45 of file kitmodel.hpp.

References `m_dirty`.

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

Definition at line 46 of file kitmodel.hpp.

References `m_dirty`.

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()`.

7.5.3.3 `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()`.

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

Definition at line 48 of file kitmodel.hpp.

References `m_deleted`.

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

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

Definition at line 49 of file kitmodel.hpp.

References `m_new`.

Referenced by `Service::create_category()`, and `Service::create_item()`.

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

Definition at line 50 of file kitmodel.hpp.

References `m_new`.

7.5.3.7 `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 `KitModel::reset()`, and `ModelCategory::reset()`.

7.5.4 Member Data Documentation

7.5.4.1 `bool GuiState::m_dirty` [protected]

Definition at line 40 of file kitmodel.hpp.

Referenced by `is_dirty()`, `reset()`, and `set_dirty()`.

7.5.4.2 `bool GuiState::m_deleted` [protected]

Definition at line 41 of file kitmodel.hpp.

Referenced by `is_deleted()`, `reset()`, and `set_deleted()`.

7.5.4.3 `bool GuiState::m_new` [protected]

Definition at line 42 of file kitmodel.hpp.

Referenced by `is_new()`, `reset()`, and `set_new_flag()`.

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

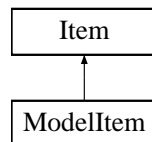
- [/home/frank/projects/kitlist/src/kitmodel.hpp](#)
- [/home/frank/projects/kitlist/src/kitmodel.cpp](#)

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::Item ()` [inline]

Definition at line 42 of file `item.hpp`.

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

Definition at line 45 of file `item.hpp`.

References `m_id`.

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

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

Definition at line 46 of file `item.hpp`.

References `m_id`.

Referenced by `XmlDao::add_category_item_to_dom()`, `KitModel::add_item()`, `ModelCategory::add_item()`, `XmlDao::add_item_to_dom()`, `XmlDao::get_model()`, `KitList::list_item()`, `TickItem::operator()`, `ModelItemCompareId::operator()`, and `ModelCategory::remove_item()`.

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

Definition at line 47 of file `item.hpp`.

References `m_desc`.

Referenced by `KitListGui::close_add_item_window()`, `KitParser::on_end_element()`, `KitListGui::on_menu_add()`, and `Service::update_item()`.

7.6.3.4 `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()`.

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

Reimplemented in [ModelItem](#).

Definition at line 49 of file item.hpp.

References m_checked.

Referenced by KitListGui::close_add_item_window(), KitListGui::on_menu_add(), TickItem::operator(), and ModelItem::set_checked().

7.6.3.6 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().

7.6.4 Friends And Related Function Documentation

7.6.4.1 friend class ItemCompareName [friend]

Definition at line 51 of file item.hpp.

7.6.4.2 friend class ItemCompareId [friend]

Definition at line 52 of file item.hpp.

7.6.4.3 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 long Item::m_id [private]

Unique ID.

Definition at line 38 of file item.hpp.

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

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

The item's description.

Definition at line 39 of file item.hpp.

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

7.6.5.3 bool Item::m_checked [private]

Whether checked/ticked or not.

Definition at line 40 of file item.hpp.

Referenced by `get_checked()`, and `set_checked()`.

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 ()` [inline]

Definition at line 74 of file `item.hpp`.

7.7.3 Member Function Documentation

7.7.3.1 `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 ()` [`inline`]

Definition at line 63 of file `item.hpp`.

7.8.3 Member Function Documentation

7.8.3.1 `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:

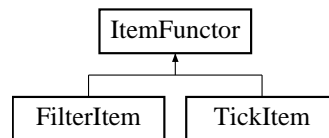
- `/home/frank/projects/kitlist/src/item.hpp`

7.9 ItemFuncor Class Reference

Funcor for processing items.

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

Inheritance diagram for ItemFuncor::



Public Member Functions

- virtual bool `operator()` (`Item &item`)=0

7.9.1 Detailed Description

Funcor 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 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 83 of file main.cpp.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 `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:

dbname The name of the database.

user The database user name to connect with.

port The database port to connect to.

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

Definition at line 167 of file main.cpp.

References `m_dao`.

7.10.2.2 `KitList::~~KitList ()`

Definition at line 176 of file main.cpp.

References `m_dao`.

7.10.3 Member Function Documentation

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

Definition at line 89 of file main.cpp.

References `m_dao`.

Referenced by `main()`.

7.10.3.2 void KitList::add_item (const string *name*)

Referenced by main().

7.10.3.3 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:

name The name of the item to create.

cat_id 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 189 of file main.cpp.

References KitListDao::add_item(), and m_dao.

7.10.3.4 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:

from_cat_id The ID of the source category.

to_cat_id The ID of the target category.

choice One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Definition at line 208 of file main.cpp.

References KitListDao::append_items_to_category(), and m_dao.

Referenced by main().

7.10.3.5 void KitList::associate_item_with_category (long *id*, long *cat_id*)

Associates an existing item with an existing category.

Parameters:

id The [Item](#) ID

cat_id The [Category](#) ID

Definition at line 219 of file main.cpp.

References KitListDao::associate_item_with_category(), and m_dao.

Referenced by main().

7.10.3.6 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:

empty_list Set to true if the list is empty. If so, table headings will not be output.

Definition at line 243 of file `main.cpp`.

References `html_flag`.

Referenced by `list_items()`.

7.10.3.7 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 297 of file `main.cpp`.

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

Referenced by `list_items()`, and `on_list_item()`.

7.10.3.8 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:

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

Definition at line 274 of file `main.cpp`.

References `html_flag`.

Referenced by `list_items()`.

7.10.3.9 void KitList::list_items (Category & *c*)

Lists details of all items in the passed [Category](#) to STDOUT.

Definition at line 313 of file `main.cpp`.

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

Referenced by `list_items()`, and `main()`.

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

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

Definition at line 228 of file main.cpp.

References [list_item\(\)](#).

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

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

Lists details of all items in the passed [ItemContainer](#) to STDOUT.

Definition at line 323 of file main.cpp.

References [list_item\(\)](#), [list_items_end\(\)](#), and [list_items_start\(\)](#).

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

Lists details of items to STDOUT.

Parameters:

cat_id The ID Of the [Category](#) to list. If the value is less than zero, all items are listed.

choice Optional. One of ALL_ITEMS (default), CHECKED_ITEMS or UNCHECKED_ITEMS.

Definition at line 341 of file main.cpp.

References [KitListDao::get_all_items\(\)](#), [KitListDao::get_category\(\)](#), [list_items\(\)](#), and [m_dao](#).

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

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

See also:

[ItemFuncion](#).

Definition at line 361 of file main.cpp.

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

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

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

Definition at line 383 of file main.cpp.

References [Category::execute\(\)](#), [m_dao](#), and [KitListDao::update_item_checked_state\(\)](#).

Referenced by [main\(\)](#), and [tick_items\(\)](#).

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

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

Definition at line 372 of file main.cpp.

References `execute()`, `m_dao`, and `KitListDao::update_item_checked_state()`.

7.10.3.16 `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:

choice One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Definition at line 399 of file main.cpp.

References `KitListDao::get_all_items()`, `KitListDao::get_category()`, `m_dao`, and `tick_items()`.

7.10.3.17 `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 452 of file main.cpp.

References `KitListDao::get_categories()`, `html_flag`, and `m_dao`.

Referenced by `main()`.

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

Creates a new category with the passed name.

Definition at line 501 of file main.cpp.

References `m_dao`, and `KitListDao::new_category()`.

Referenced by `main()`.

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

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

Parameters:

id The [Category](#) id.

Definition at line 440 of file main.cpp.

References `KitListDao::delete_category()`, and `m_dao`.

Referenced by `main()`.

7.10.3.20 void KitList::delete_item (long id)

Deletes the item with the passed id.

Definition at line 418 of file main.cpp.

References KitListDao::delete_item(), and m_dao.

Referenced by main().

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

Un-associates the specified item from the specified category.

Parameters:

id The [Item](#) id.

cat_id The [Category](#) id.

Definition at line 430 of file main.cpp.

References m_dao, and KitListDao::remove_item_from_category().

Referenced by main().

7.10.3.22 void KitList::set_item_flag (long id)

Checks/ticks the specified item.

Definition at line 510 of file main.cpp.

References m_dao, and KitListDao::set_item_flag().

Referenced by main().

7.10.3.23 void KitList::unset_item_flag (long id)

Unchecks/unticks the specified item.

Definition at line 519 of file main.cpp.

References m_dao, and KitListDao::unset_item_flag().

Referenced by main().

7.10.3.24 void KitList::set_category_flag (long id)

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

Definition at line 528 of file main.cpp.

References m_dao, and KitListDao::set_category_flag().

Referenced by main().

7.10.3.25 void KitList::unset_category_flag (long id)

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

Definition at line 537 of file main.cpp.

References `m_dao`, and `KitListDao::unset_category_flag()`.

Referenced by `main()`.

7.10.3.26 void KitList::set_all_flags ()

Checks/ticks all items.

Definition at line 546 of file main.cpp.

References `m_dao`, and `KitListDao::set_all_flags()`.

Referenced by `main()`.

7.10.3.27 void KitList::unset_all_flags ()

Unchecks/unticks all items.

Definition at line 555 of file main.cpp.

References `m_dao`, and `KitListDao::unset_all_flags()`.

Referenced by `main()`.

7.10.4 Member Data Documentation

7.10.4.1 KitListDao* KitList::m_dao [protected]

Reference to an implementation DAO class.

Definition at line 85 of file main.cpp.

Referenced by `add_item()`, `append_items_to_category()`, `associate_item_with_category()`, `delete_category()`, `delete_item()`, `get_dao()`, `KitList()`, `list_categories()`, `list_items()`, `new_category()`, `remove_item_from_category()`, `set_all_flags()`, `set_category_flag()`, `set_item_flag()`, `tick_items()`, `unset_all_flags()`, `unset_category_flag()`, `unset_item_flag()`, and `~KitList()`.

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

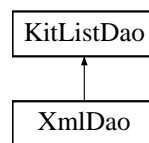
- [/home/frank/projects/kitlist/src/main.cpp](#)

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

Constructor which will use default database connection parameters.

Parameters:

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

Definition at line 61 of file `kitlistdao.hpp`.

7.11.3 Member Function Documentation

7.11.3.1 `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()`, and `Service::load_model()`.

7.11.3.2 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()`.

7.11.3.3 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 `m_verbose_flag`.

7.11.3.4 int KitListDao::is_verbose () [inline]

Indicates whether verbose output should be written to STDOUT.

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

References `m_verbose_flag`.

7.11.3.5 virtual Category* KitListDao::get_category (long *cat_id*, item_choice *choice* = ALL_ITEMS) [pure virtual]

Loads a category.

Parameters:

choice Which items to load for the category. One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Implemented in [XmlDao](#).

Referenced by `KitList::list_items()`, and `KitList::tick_items()`.

7.11.3.6 virtual ItemContainer* KitListDao::get_all_items (item_choice *choice* = ALL_ITEMS) [pure virtual]

Returns a list of all items.

Parameters:

choice Which items to load. One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Implemented in [XmlDao](#).

Referenced by `KitList::list_items()`, and `KitList::tick_items()`.

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

Creates a new item.

Parameters:

name The name of the new item.

Implemented in [XmlDao](#).

Referenced by `KitList::add_item()`.

7.11.3.8 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:

name The name of the new item.

Implemented in [XmlDao](#).

7.11.3.9 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:

from_cat_id The ID of the source category.

to_cat_id The ID of the target category.

choice One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Implemented in [XmlDao](#).

Referenced by `KitList::append_items_to_category()`.

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

Associates an existing item with an existing category.

Parameters:

id The [Item](#) ID

cat_id The [Category](#) ID

Implemented in [XmlDao](#).

Referenced by `KitList::associate_item_with_category()`.

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

Returns a list of all categories.

Implemented in [XmlDao](#).

Referenced by `KitList::list_categories()`.

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

Creates a new category.

Parameters:

name the name of the new category.

Implemented in [XmlDao](#).

Referenced by `KitList::new_category()`.

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

Deletes an item by it's ID.

Parameters:

id the ID of the item to delete.

Implemented in [XmlDao](#).

Referenced by `KitList::delete_item()`.

7.11.3.14 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 `KitList::tick_items()`.

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

Un-associates the specified item from the specified category.

Parameters:

id The [Item](#) id.

cat_id The [Category](#) id.

Implemented in [XmlDao](#).

Referenced by [KitList::remove_item_from_category\(\)](#).

7.11.3.16 **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\(\)](#).

7.11.3.17 **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\(\)](#).

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

Deletes a category.

Implemented in [XmlDao](#).

Referenced by [KitList::delete_category\(\)](#).

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

Sets the checked flag for an item.

Parameters:

id The id of the item to change.

Implemented in [XmlDao](#).

Referenced by [KitList::set_item_flag\(\)](#).

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

Clears the checked flag for an item.

Implemented in [XmlDao](#).

Referenced by [KitList::unset_item_flag\(\)](#).

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

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

Implemented in [XmlDao](#).

Referenced by [KitList::set_category_flag\(\)](#).

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

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

Implemented in [XmlDao](#).

Referenced by [KitList::unset_category_flag\(\)](#).

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

Checks/ticks all items.

Implemented in [XmlDao](#).

Referenced by [KitList::set_all_flags\(\)](#).

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

Unchecks/unticks all items.

Implemented in [XmlDao](#).

Referenced by [KitList::unset_all_flags\(\)](#).

7.11.3.25 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\(\)](#).

7.11.4 Member Data Documentation

7.11.4.1 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\(\)](#), and [set_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 [ModelItemContainer](#) * [get_selected_items](#) ()
Returns a list of items selected in the item list.
- virtual void [add_items](#) (const [ModelItemContainer](#) &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 `ModelItemContainer * 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_menu_help_contents \(\)](#)
- 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::CheckButton * 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

7.12.1 Detailed Description

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

This is the GTK+ implementation.

Definition at line 100 of file `kitlistgui.hpp`.

7.12.2 Constructor & Destructor Documentation

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

Constructor to create the GUI application.

Parameters:

- argc* command line argument count passed to `Gtk::Main`
- argv* command line arguments passed to `Gtk::Main`
- service* a reference to the [Service](#) object, providing all business/service methods.

Definition at line 294 of file `kitlistgui.cpp`.

References `init()`, `m_ref_page_setup`, and `m_ref_printer_settings`.

7.12.2.2 KitListGui::~~KitListGui ()

Definition at line 325 of file `kitlistgui.cpp`.

7.12.3 Member Function Documentation

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

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

Definition at line 2166 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`, `file_exists()`, `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY`, `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_CURRENT_FILENAME`, `anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_PAGE_TITLE`, `get_glade_ref_ptr()`, `Service::get_items()`, `anonymous_namespace{kitlistgui.cpp}::GLADE_APP_FILE`, `KITLIST_SERVICE_IFACE`, `KITLIST_SERVICE_NAME`, `KITLIST_SERVICE_OBJECT`,

m_category_cols, m_category_combo, m_checkbutton_add_item, ModelItemColumns::m_col_checked, ModelItemColumns::m_col_num, ModelItemColumns::m_col_text, ModelCategoryColumns::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, 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_help_contents(), 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().

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

Definition at line 358 of file kitlistgui.cpp.

References anonymous_namespace{kitlistgui.cpp}::DEFAULT_MAX_RECENT_FILES, and anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_MAX_RECENT_FILES.

Referenced by update_recent_files().

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

Returns a list of items selected in the item list.

Definition at line 959 of file kitlistgui.cpp.

References Service::find_item(), ModelItemColumns::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().

7.12.3.4 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 1472 of file kitlistgui.cpp.

References Service::copy_items(), get_selected_category(), m_service, and refresh_item_list().

Referenced by paste_from_xml().

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

Definition at line 2004 of file kitlistgui.cpp.

References anonymous_namespace{kitlistgui.cpp}::GCONF_KEY_PAGE_TITLE, and m_page_title.

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

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

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

Definition at line 2023 of file `kitlistgui.cpp`.

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

Referenced by `init()`.

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

Definition at line 2029 of file `kitlistgui.cpp`.

References `m_window_preferences`.

Referenced by `init()`.

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

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

Definition at line 2121 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 `void KitListGui::cancel_add_item_window ()` [protected, virtual]

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

Definition at line 2141 of file `kitlistgui.cpp`.

References `m_window_add_item`.

Referenced by `init()`.

7.12.3.10 `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 1226 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()`.

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

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

Definition at line 1251 of file kitlistgui.cpp.

References `m_window_add_category`.

Referenced by `init()`.

7.12.3.12 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 2104 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_add()`, `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()`.

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

Initialises the 'Add Item' dialog prior to it being displayed.

Definition at line 2091 of file kitlistgui.cpp.

References `m_entry_add_item`.

Referenced by `on_menu_add()`.

7.12.3.14 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 1005 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()`.

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

Copies the currently selected items to the clipboard.

Definition at line 1060 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()`.

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

Shows a confirmation message.

Definition at line 415 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()`.

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

Updates the recent files sub menu.

Definition at line 453 of file `kitlistgui.cpp`.

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

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

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

Updates the list of most recently used files.

Parameters:

filename The filename to append to the list if it does not already exist.

Definition at line 490 of file `kitlistgui.cpp`.

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

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

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

Called when the application is closed by the user.

Definition at line 519 of file `kitlistgui.cpp`.

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

Referenced by `init()`.

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

Called when the user chooses to quit the application.

Definition at line 533 of file kitlistgui.cpp.

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

Referenced by `init()`.

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

Creates a new empty model.

Definition at line 550 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`, `refresh_category_list()`, `refresh_item_list()`, `Service::require_filename()`, and `anonymous_namespace{kitlistgui.cpp}::SB_SAVE`.

Referenced by `init()`.

7.12.3.22 `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 597 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()`.

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

Saves the current application state.

Definition at line 700 of file kitlistgui.cpp.

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

Referenced by `init()`.

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

Saves the current application state in a new document.

Definition at line 753 of file kitlistgui.cpp.

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

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

7.12.3.25 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 826 of file kitlistgui.cpp.

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

Referenced by `on_menu_print()`.

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

Called when the print status changes.

Definition at line 811 of file kitlistgui.cpp.

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

Referenced by `on_printoperation_done()`.

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

Prints the kit list.

Definition at line 841 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()`.

7.12.3.28 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 870 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()`.

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

displays the most recent files menu

Definition at line 926 of file kitlistgui.cpp.

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

Referenced by `update_recent_files_menu()`.

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

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

Definition at line 1958 of file kitlistgui.cpp.

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

Referenced by `init()`.

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

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

Definition at line 2037 of file kitlistgui.cpp.

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

Referenced by `init()`.

7.12.3.32 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, the will be permanently deleted from the persistence store.

Definition at line 1037 of file kitlistgui.cpp.

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

Referenced by `init()`.

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

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

Definition at line 1093 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()`.

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

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

Definition at line 1122 of file kitlistgui.cpp.

References `copy_selected_items_to_clipboard()`.

Referenced by `init()`.

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

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

Definition at line 1131 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 virtual void KitListGui::on_menu_show_all () [inline, protected, virtual]

Causes all items to be displayed.

Definition at line 233 of file kitlistgui.hpp.

References m_service, refresh_item_list(), and Service::show_all().

Referenced by init().

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

Causes only checked items to be displayed.

Definition at line 235 of file kitlistgui.hpp.

References m_service, refresh_item_list(), and Service::show_checked_only().

Referenced by init().

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

Causes only unchecked items to be displayed.

Definition at line 237 of file kitlistgui.hpp.

References m_service, refresh_item_list(), and Service::show_unchecked_only().

Referenced by init().

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

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

Definition at line 1145 of file kitlistgui.cpp.

References m_item_tree_view.

Referenced by init().

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

Marks all selected items as checked.

Definition at line 240 of file kitlistgui.hpp.

References `set_selected()`.

Referenced by `init()`.

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

Marks all selected items as unchecked.

Definition at line 242 of file kitlistgui.hpp.

References `set_selected()`.

Referenced by `init()`.

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

Called when the user chooses to create a new category.

Definition at line 1156 of file kitlistgui.cpp.

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

Referenced by `init()`.

7.12.3.43 `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 1262 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()`.

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

Called when the user chooses to rename a category.

Definition at line 1299 of file kitlistgui.cpp.

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

Referenced by `init()`.

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

Shows the help about dialog

Definition at line 1393 of file kitlistgui.cpp.

Referenced by `init()`.

7.12.3.46 `void KitListGui::on_menu_help_contents ()` [protected, virtual]

Displays the help documentation

Definition at line 1379 of file kitlistgui.cpp.

Referenced by `init()`.

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

Called when the current clipboard contents are required.

Definition at line 1430 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()`.

7.12.3.48 `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 1461 of file kitlistgui.cpp.

Referenced by `copy_selected_items_to_clipboard()`.

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

Callback notification method for capturing clipboard contents.

Definition at line 1520 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()`.

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

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

Definition at line 2149 of file kitlistgui.cpp.

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

Referenced by `init()`.

7.12.3.51 `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 1570 of file kitlistgui.cpp.

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

7.12.3.52 `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:

filename A reference to a string to populate with the chosen filename.

Returns:

true if the user selects OK, false otherwise.

Definition at line 1587 of file kitlistgui.cpp.

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

Referenced by `on_menu_save_as()`.

7.12.3.53 `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:

filename A reference to a string to populate with the chosen filename.

Returns:

true if the user selects OK, false otherwise.

Definition at line 1691 of file kitlistgui.cpp.

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

Referenced by `on_menu_export_to_pdf()`.

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

Enables or disables the paste menu option.

Definition at line 1779 of file kitlistgui.cpp.

References `paste_status_received()`.

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

7.12.3.55 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 1791 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()`.

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

Performs a paste of items from the clipboard.

Definition at line 1484 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()`.

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

Refreshes the item list.

Definition at line 1864 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_add()`, `on_menu_create_category()`, `on_menu_cut()`, `on_menu_delete_category()`, `on_menu_file_new()`, `on_menu_recent_file()`, `on_menu_rename_category()`, `on_menu_show_all()`, `on_menu_show_checked()`, `on_menu_show_unchecked()`, `open_file()`, `set_selected()`, and `toggle_selected()`.

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

Refreshes the category combo box list.

Parameters:

cat_id 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 1903 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, ModelCategoryColumns::m_col_text, m_ignore_list_events, m_ref_category_list_store, and m_service.

Referenced by close_add_category_window(), init(), on_menu_create_category(), on_menu_delete_category(), on_menu_file_new(), on_menu_recent_file(), on_menu_rename_category(), and open_file().

7.12.3.59 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 1834 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 void KitListGui::set_selected (bool checked) [protected, virtual]

Checks or unchecks the currently selected items.

Definition at line 1408 of file kitlistgui.cpp.

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

Referenced by on_menu_check_selected(), and on_menu_uncheck_selected().

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

Toggles the checked state of the currently selected items.

Definition at line 1419 of file kitlistgui.cpp.

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

Referenced by init().

7.12.3.62 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 1818 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()`.

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

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

Definition at line 1534 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 `void KitListGui::open_file (const Glib::ustring & filename)` [virtual]

Opens an existing XML document.

Definition at line 655 of file `kitlistgui.cpp`.

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

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

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

Make this application topmost.

Definition at line 2157 of file `kitlistgui.cpp`.

References `m_window`.

7.12.3.66 `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 686 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`.

7.12.3.67 `void KitListGui::run ()`

Starts the GUI application running.

Definition at line 336 of file `kitlistgui.cpp`.

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

7.12.4 Member Data Documentation

7.12.4.1 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 111 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()`, `open_file()`, and `run()`.

7.12.4.2 Glib::ustring KitListGui::m_page_title [protected]

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

Definition at line 113 of file kitlistgui.hpp.

Referenced by `init()`, `on_menu_export_to_pdf()`, `on_menu_preferences()`, `on_menu_print()`, and `set_page_title()`.

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

Holder for items pasted to the clipboard.

Definition at line 119 of file kitlistgui.hpp.

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

7.12.4.4 bool KitListGui::m_ignore_list_events [protected]

Temporarily ignore events on the item list.

Definition at line 121 of file kitlistgui.hpp.

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

7.12.4.5 Gtk::Main KitListGui::m_kit [protected]

The main application.

Definition at line 123 of file kitlistgui.hpp.

Referenced by `run()`.

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

The main application window.

Definition at line 135 of file kitlistgui.hpp.

Referenced by `choose_filename()`, `choose_pdf_filename()`, `confirm_lose_changes()`, `init()`, `on_menu_add()`, `on_menu_create_category()`, `on_menu_delete()`, `on_menu_delete_category()`, `on_menu_file_`

open(), on_menu_preferences(), on_menu_quit(), on_menu_rename_category(), on_printoperation_done(), raise(), and run().

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

The 'Preferences' dialog.

Definition at line 138 of file kitlistgui.hpp.

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

7.12.4.8 Gtk::Entry* KitListGui::m_entry_page_title [protected]

the text entry field for the page title

Definition at line 140 of file kitlistgui.hpp.

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

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

The 'Add Item' dialog.

Definition at line 142 of file kitlistgui.hpp.

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

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

The 'Add Category' dialog.

Definition at line 144 of file kitlistgui.hpp.

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

7.12.4.11 Gtk::Entry* KitListGui::m_entry_add_item [protected]

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

Definition at line 146 of file kitlistgui.hpp.

Referenced by close_add_item_window(), init(), init_add_item_window(), and on_menu_add().

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

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

Definition at line 148 of file kitlistgui.hpp.

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

7.12.4.13 `Gtk::ImageMenuItem* KitListGui::m_file_save_menu_item` [protected]

The file save menu item.

Definition at line 150 of file kitlistgui.hpp.

Referenced by `init()`.

7.12.4.14 `Gtk::ToolButton* KitListGui::m_file_save_tool_button` [protected]

The file save toolbar button.

Definition at line 152 of file kitlistgui.hpp.

Referenced by `init()`.

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

The recent files menu item.

Definition at line 154 of file kitlistgui.hpp.

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

7.12.4.16 `Gtk::ImageMenuItem* KitListGui::m_paste_menu_item` [protected]

The menu paste button.

Definition at line 156 of file kitlistgui.hpp.

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

7.12.4.17 `Gtk::ToolButton* KitListGui::m_paste_tool_button` [protected]

The toolbar paste button.

Definition at line 158 of file kitlistgui.hpp.

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

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

The check button field of the 'Add Item' dialog.

Definition at line 160 of file kitlistgui.hpp.

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

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

The combo box holding a list of categories.

Definition at line 162 of file kitlistgui.hpp.

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

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

The model backing the category combo box.

Definition at line 164 of file kitlistgui.hpp.

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

7.12.4.21 ModelCategoryColumns KitListGui::m_category_cols [protected]

The definition of the category combo box columns.

Definition at line 166 of file kitlistgui.hpp.

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

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

The item list view definition.

Definition at line 168 of file kitlistgui.hpp.

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

7.12.4.23 ModelItemColumns KitListGui::m_item_cols [protected]

The definition of the item list's columns.

Definition at line 170 of file kitlistgui.hpp.

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

7.12.4.24 Service& KitListGui::m_service [protected]

The business/service object.

Definition at line 172 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_add()`, `on_menu_create_category()`, `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_menu_show_all()`, `on_menu_show_checked()`, `on_menu_show_unchecked()`, `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 Glib::RefPtr<Gtk::ListStore> KitListGui::m_ref_item_tree_model [protected]

The model backing the item list.

Definition at line 174 of file kitlistgui.hpp.

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

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

The application status bar.

Definition at line 176 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 `Glib::RefPtr<Gtk::PageSetup> KitListGui::m_ref_page_setup` [protected]

Printer page setup settings.

Definition at line 178 of file `kitlistgui.hpp`.

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

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

Printer settings.

Definition at line 180 of file `kitlistgui.hpp`.

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

7.12.4.29 `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 188 of file `kitlistgui.hpp`.

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

7.12.4.30 `long KitListGui::m_current_cat_id` [protected]

temporary reference to a category id, usually being renamed

Definition at line 189 of file `kitlistgui.hpp`.

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

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 [SlotForeachModelItemIter](#) &slot)
Calls the provided slot for each item in the map.
- void [foreach_item](#) (const [SlotForeachModelItem](#) &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 [ModelItem](#) * [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](#) &functor)
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](#) ([ModelItem](#) *item)
Adds an item to the model.
- virtual void [add_item](#) ([ModelItem](#) *item, long cat_id)
Adds an item to the model and associates it with the specified category.

- virtual void `copy_items` (const `ModelItemContainer` &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 ()

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 ()

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 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`.

7.13.3.2 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()`.

7.13.3.3 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`.

7.13.3.4 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()`.

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

Finds a [Category](#) by its 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()`.

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

Finds an item by its 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()`.

7.13.3.7 `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()`.

7.13.3.8 `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()`.

7.13.3.9 `ItemContainer * KitModel::get_all_items (ItemFuncion & 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`.

7.13.3.10 `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().

7.13.3.11 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().

7.13.3.12 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.

7.13.3.13 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().

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

Applies the current filter.

Parameters:

checked 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()().

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

Definition at line 176 of file kitmodel.hpp.

References `m_dirty`.

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()`.

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

Definition at line 177 of file kitmodel.hpp.

References `m_dirty`.

Referenced by `Service::is_model_dirty()`.

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

Removes filter. All items are shown.

Definition at line 179 of file kitmodel.hpp.

References `ALL`, and `m_item_filter`.

Referenced by `Service::show_all()`.

7.13.3.18 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`, and `m_item_filter`.

Referenced by `Service::show_checked_only()`.

7.13.3.19 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 `m_item_filter`, and `UNCHECKED`.

Referenced by `Service::show_unchecked_only()`.

7.13.3.20 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()`.

7.13.3.21 `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()`.

7.13.4 Member Data Documentation

7.13.4.1 `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 `is_dirty()`, `reset()`, and `set_dirty()`.

7.13.4.2 `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()`.

7.13.4.3 `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()`.

7.13.4.4 `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()`, `show_all()`, `show_checked_only()`, and `show_unchecked_only()`.

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>
```

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 \(KitModel & model\)](#) [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 [virtual KitParser::~~KitParser \(\)](#) [inline, virtual]

Definition at line 46 of file kitparser.hpp.

7.14.3 Member Function Documentation

7.14.3.1 [void KitParser::process_item \(const AttributeList & attributes\)](#) [private]

<The most recently processed CDATA
Reads an item's attributes from an 'item' element.

Parameters:

attributes 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().

7.14.3.2 void KitParser::process_category (const AttributeList & attributes) [private]

Reads a category's attributes from a 'category' element.

Parameters:

attributes 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().

7.14.3.3 void KitParser::process_category_item (const AttributeList & attributes) [private]

Reads details of an item association with a category from a 'category-item' element.

Parameters:

attributes 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().

7.14.3.4 void KitParser::on_start_document () [protected, virtual]

Does nothing. Called at the start of a document.

Definition at line 29 of file kitparser.cpp.

7.14.3.5 void KitParser::on_end_document () [protected, virtual]

Does nothing. Called at the end of a document.

Definition at line 35 of file kitparser.cpp.

7.14.3.6 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().

7.14.3.7 `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()`.

7.14.3.8 `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`.

7.14.3.9 `void KitParser::on_comment (const Glib::ustring & text)` [protected, virtual]

Does nothing. Called for each comment.

Definition at line 171 of file kitparser.cpp.

7.14.3.10 `void KitParser::on_warning (const Glib::ustring & text)` [protected, virtual]

Outputs any warnings to STDOUT.

Definition at line 177 of file kitparser.cpp.

7.14.3.11 `void KitParser::on_error (const Glib::ustring & text)` [protected, virtual]

Outputs any errors to STDOUT.

Definition at line 183 of file kitparser.cpp.

7.14.3.12 `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.

7.14.4 Member Data Documentation

7.14.4.1 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 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().

7.14.4.3 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().

7.14.4.4 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>
```

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 ()

Destructor.

Definition at line 44 of file printing.cpp.

References `m_items`.

7.15.3 Member Function Documentation

7.15.3.1 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 void KitPrintOperation::set_items (ItemContainer * items) [inline]

Sets the list of items to be printed.

Definition at line 41 of file printing.hpp.

References `m_items`.

7.15.3.3 void KitPrintOperation::set_page_title (const Glib::ustring page_title) [inline]

Definition at line 42 of file printing.hpp.

References `m_page_title`.

7.15.3.4 layout_refptr KitPrintOperation::new_header (const Glib::RefPtr< Gtk::PrintContext > & context) [protected]

Creates a new header element for a page

Parameters:

context the print context

Definition at line 60 of file printing.cpp.

References `m_page_title`, and `m_ref_headers`.

Referenced by `on_begin_print()`.

7.15.3.5 `layout_refptr KitPrintOperation::new_footer (const Glib::RefPtr< Gtk::PrintContext > & context)` [protected]

Creates a new footer element for a page

Parameters:

context the print context

Definition at line 82 of file printing.cpp.

References FOOTER_TEXT, and m_ref_footers.

Referenced by on_begin_print().

7.15.3.6 `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:

context the print context

Definition at line 99 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.

7.15.3.7 `void KitPrintOperation::on_draw_page (const Glib::RefPtr< Gtk::PrintContext > & context, int page_number)` [protected, virtual]

Prints a specified page.

Parameters:

context the print context

page_number the number of the page to be printed

Definition at line 171 of file printing.cpp.

References BORDER_SPACING, HEADER_SPACING, m_page_breaks, m_ref_footers, m_ref_headers, and m_ref_layout.

7.15.4 Member Data Documentation

7.15.4.1 `ItemContainer* KitPrintOperation::m_items` [private]

Definition at line 36 of file printing.hpp.

Referenced by on_begin_print(), set_items(), and ~KitPrintOperation().

7.15.4.2 Glib::ustring KitPrintOperation::m_page_title [private]

Definition at line 37 of file printing.hpp.

Referenced by new_header(), and set_page_title().

7.15.4.3 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().

7.15.4.4 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().

7.15.4.5 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 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().

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

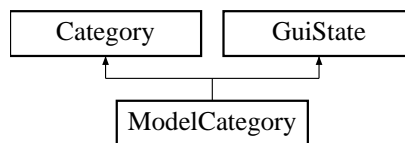
- [/home/frank/projects/kitlist/src/printing.hpp](#)
- [/home/frank/projects/kitlist/src/printing.cpp](#)

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 [ModelItemContainer](#) * [get_model_items](#) ()
- virtual [ItemContainer](#) * [get_items](#) ()
Returns the list of items belonging to the [Category](#).
- virtual [ItemContainer](#) * [get_items](#) ([ItemFuncor](#) &funcor)
Returns the list of items belonging to the [Category](#), filtered by the passed funcor.
- 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](#) ([ModelItemContainer](#) *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 ()

Definition at line 43 of file `kitmodel.cpp`.

References `m_added_children`, and `m_removed_children`.

7.16.2.2 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 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`.

7.16.3.2 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()`.

7.16.3.3 `ItemContainer * ModelCategory::get_items (ItemFunctor & functor)` [virtual]

Returns the list of items belonging to the [Category](#), filtered by the passed functor.
Items flagged as deleted are excluded.

Parameters:

functor 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`.

7.16.3.4 `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`.

References `m_removed_children`.

7.16.3.5 `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 `m_added_children`.

7.16.3.6 `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()`.

7.16.3.7 `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()`.

7.16.3.8 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\(\)](#).

7.16.3.9 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\(\)](#).

7.16.3.10 void ModelCategory::purge () [virtual]

Definition at line 68 of file kitmodel.cpp.

References [Category::m_items](#).

Referenced by [KitModel::purge\(\)](#).

7.16.4 Friends And Related Function Documentation**7.16.4.1 friend class KitModel [friend]**

Reimplemented from [Category](#).

Definition at line 115 of file kitmodel.hpp.

7.16.5 Member Data Documentation**7.16.5.1 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\(\)](#), [get_removed_children\(\)](#), [ModelCategory\(\)](#), [remove_item\(\)](#), [reset\(\)](#), and [~ModelCategory\(\)](#).

7.16.5.2 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()`, `get_added_children()`, `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>
```

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 59 of file kitlistgui.hpp.

7.17.2 Constructor & Destructor Documentation

7.17.2.1 ModelCategoryColumns::ModelCategoryColumns () [inline]

Definition at line 62 of file kitlistgui.hpp.

References [m_col_num](#), and [m_col_text](#).

7.17.3 Member Data Documentation

7.17.3.1 Gtk::TreeModelColumn<Glib::ustring> ModelCategoryColumns::m_col_text

Definition at line 67 of file kitlistgui.hpp.

Referenced by [KitListGui::init\(\)](#), [ModelCategoryColumns\(\)](#), and [KitListGui::refresh_category_list\(\)](#).

7.17.3.2 Gtk::TreeModelColumn<int> ModelCategoryColumns::m_col_num

Definition at line 68 of file kitlistgui.hpp.

Referenced by [KitListGui::get_selected_category\(\)](#), [ModelCategoryColumns\(\)](#), and [KitListGui::refresh_category_list\(\)](#).

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

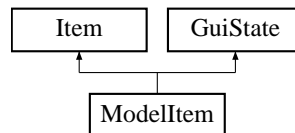
- [/home/frank/projects/kitlist/src/kitlistgui.hpp](#)

7.18 ModelItem Class Reference

Represents an [Item](#) combined with [GuiState](#) attributes.

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

Inheritance diagram for ModelItem::



Public Member Functions

- [ModelItem](#) ()
- virtual void [set_checked](#) (bool checked)

Friends

- class [ModelItemCompareId](#)

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 () [inline]

Definition at line 62 of file kitmodel.hpp.

7.18.3 Member Function Documentation

7.18.3.1 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 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>
```

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 78 of file kitlistgui.hpp.

7.19.2 Constructor & Destructor Documentation

7.19.2.1 ModelItemColumns::ModelItemColumns () [inline]

Definition at line 81 of file kitlistgui.hpp.

References [m_col_checked](#), [m_col_num](#), and [m_col_text](#).

7.19.3 Member Data Documentation

7.19.3.1 Gtk::TreeModelColumn<Glib::ustring> ModelItemColumns::m_col_text

Definition at line 87 of file kitlistgui.hpp.

Referenced by [KitListGui::init\(\)](#), [ModelItemColumns\(\)](#), [KitListGui::on_row_changed\(\)](#), and [KitListGui::refresh_item_list\(\)](#).

7.19.3.2 Gtk::TreeModelColumn<bool> ModelItemColumns::m_col_checked

Definition at line 88 of file kitlistgui.hpp.

Referenced by [KitListGui::init\(\)](#), [ModelItemColumns\(\)](#), [KitListGui::on_row_changed\(\)](#), and [KitListGui::refresh_item_list\(\)](#).

7.19.3.3 Gtk::TreeModelColumn<int> ModelItemColumns::m_col_num

Definition at line 89 of file kitlistgui.hpp.

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

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

- </home/frank/projects/kitlist/src/kitlistgui.hpp>

7.20 ModelItemCompareId Class Reference

Comparator for comparing items by their unique ID.

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

Public Member Functions

- [ModelItemCompareId \(\)](#)
- `int operator() (ModelItem *i1, ModelItem *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 ()` [[inline](#)]

Definition at line 74 of file kitmodel.hpp.

7.20.3 Member Function Documentation

7.20.3.1 `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](#) ()
- [ModelItem](#) * [find_item](#) (long id)
- [ModelCategory](#) * [find_category](#) (long cat_id)
- void [copy_items](#) (const [ModelItemContainer](#) &items, long cat_id)
Copies items to the specified category.
- [Item](#) * [create_item](#) (long cat_id)
Creates a new [ModelItem](#) 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 (ModelItemContainer *items, bool checked=true)`
Checks or unchecks all the passed items.
- virtual void `toggle_selected_items (ModelItemContainer *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 (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 ()

Definition at line 49 of file service.cpp.

References m_model.

7.21.3 Member Function Documentation

7.21.3.1 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().

7.21.3.2 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().

7.21.3.3 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().

7.21.3.4 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().

7.21.3.5 `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()`.

7.21.3.6 `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()`.

7.21.3.7 `Item * Service::create_item (long cat_id)`

Creates a new [ModelItem](#) 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()`, and `KitListGui::on_menu_add()`.

7.21.3.8 `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()`.

7.21.3.9 `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()`.

7.21.3.10 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()`, and `KitListGui::on_menu_create_category()`.

7.21.3.11 bool Service::is_model_dirty () [inline]

Definition at line 55 of file service.hpp.

References `KitModel::is_dirty()`, and `m_model`.

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()`.

7.21.3.12 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()`, `KitListGui::on_cell_edit()`, `KitListGui::on_delete_event()`, `KitListGui::on_menu_create_category()`, `KitListGui::on_menu_cut()`, `KitListGui::on_menu_quit()`, `KitListGui::on_menu_rename_category()`, and `KitListGui::on_row_changed()`.

7.21.3.13 virtual bool Service::filter (bool checked) [inline, virtual]

Applies the current filter.

Parameters:

checked 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 `KitModel::filter()`, and `m_model`.

Referenced by `KitListGui::refresh_item_list()`.

7.21.3.14 `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 `KitListGui::on_menu_file_new()`.

7.21.3.15 `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 `KitListGui::init()`, `KitListGui::on_menu_recent_file()`, and `KitListGui::open_file()`.

7.21.3.16 `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()`, and `KitListGui::on_menu_save()`.

7.21.3.17 `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:

filename 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()`, `KitListGui::on_menu_save()`, `KitListGui::on_menu_save_as()`, and `KitListGui::run()`.

7.21.3.18 `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:

id The unique ID of the item to update.

description The item's descriptive text.

checked 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 `KitListGui::on_row_changed()`.

7.21.3.19 `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:

cat_id 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 `KitListGui::init()`, and `KitListGui::refresh_item_list()`.

7.21.3.20 `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:

cat_id 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 `KitListGui::on_menu_export_to_pdf()`, and `KitListGui::on_menu_print()`.

7.21.3.21 `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 `KitListGui::refresh_category_list()`.

7.21.3.22 `virtual void Service::show_all () [inline, virtual]`

Removes filter. All items are shown.

Definition at line 73 of file `service.hpp`.

References `m_model`, and `KitModel::show_all()`.

Referenced by `KitListGui::on_menu_show_all()`.

7.21.3.23 `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 `m_model`, and `KitModel::show_checked_only()`.

Referenced by `KitListGui::on_menu_show_checked()`.

7.21.3.24 `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 `m_model`, and `KitModel::show_unchecked_only()`.

Referenced by `KitListGui::on_menu_show_unchecked()`.

7.21.3.25 `void Service::select_items (ModelItemContainer * items, bool checked = true) [virtual]`

Checks or unchecks all the passed items.

Parameters:

items 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()`.

7.21.3.26 void Service::toggle_selected_items (ModelItemContainer * *items*) [virtual]

Toggles the checked state of all the passed items.

Parameters:

items The items to change.

Definition at line 252 of file service.cpp.

References m_model, and KitModel::set_dirty().

Referenced by KitListGui::toggle_selected().

7.21.3.27 virtual bool Service::require_filename () [inline, virtual]

Definition at line 80 of file service.hpp.

References m_dao, and KitListDao::require_filename().

Referenced by KitListGui::init(), KitListGui::on_menu_file_new(), and KitListGui::on_menu_save().

7.21.4 Member Data Documentation**7.21.4.1 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(), require_filename(), save(), and save_as_xml().

7.21.4.2 KitModel* Service::m_model [protected]

The application's data model.

Definition at line 41 of file service.hpp.

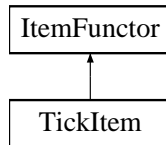
Referenced by copy_items(), create_category(), create_default_model(), create_item(), delete_category(), delete_item(), filter(), find_category(), find_item(), get_categories(), get_filtered_items(), get_items(), is_model_dirty(), open_as_xml(), save(), save_as_xml(), select_items(), Service(), set_model_dirty(), show_all(), show_checked_only(), show_unchecked_only(), 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 119 of file main.cpp.

7.22.2 Constructor & Destructor Documentation

7.22.2.1 `TickItem::TickItem` ([ItemContainer](#) & *changed*) `[inline]`

Definition at line 122 of file main.cpp.

7.22.3 Member Function Documentation

7.22.3.1 `bool TickItem::operator()` ([Item](#) & *item*) `[inline, virtual]`

Implements [ItemFunctor](#).

Definition at line 123 of file main.cpp.

References [Item::get_checked\(\)](#), [Item::get_description\(\)](#), [Item::get_id\(\)](#), [m_changed](#), and [Item::set_checked\(\)](#).

7.22.4 Member Data Documentation

7.22.4.1 `ItemContainer& TickItem::m_changed` `[private]`

Definition at line 120 of file main.cpp.

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

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

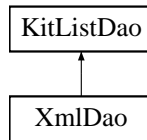
- [/home/frank/projects/kitlist/src/main.cpp](#)

7.23 XmlDao Class Reference

Implementation of a [KitListDao](#) using XML as the persistence store.

```
#include <xml dao.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 xmldao.hpp.

7.23.2 Constructor & Destructor Documentation

7.23.2.1 `XmlDao::XmlDao (int verbose = 0) [inline]`

Definition at line 67 of file xmldao.hpp.

7.23.3 Member Function Documentation

7.23.3.1 `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 xmldao.cpp.

References `Item::get_checked()`, `Item::get_description()`, `Item::get_id()`, `GuiState::is_deleted()`, and `m_items_node`.

Referenced by `save_model()`.

7.23.3.2 `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 xmldao.cpp.

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

Referenced by `add_category_to_dom()`.

7.23.3.3 `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()`, `GuiState::is_deleted()`, `m_cat_items_node`, and `m_categories_node`.

Referenced by `save_model()`.

7.23.3.4 `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:

Returns 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()`, `m_filename`, `m_max_category_id`, `m_max_item_id`, and `KitModel::reset()`.

Referenced by `get_model()`.

7.23.3.5 `KitModel* XmlDao::get_model (Glib::ustring filename)` [inline]

Definition at line 76 of file `xmldao.hpp`.

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

7.23.3.6 `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:

model 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()`, `m_categories_node`, `m_filename`, `m_items_node`, `KitModel::purge()`, and `KitModel::reset()`.

Referenced by `Service::save_as_xml()`, and `save_model()`.

7.23.3.7 `void XmlDao::save_model (KitModel * model, Glib::ustring filename)` [inline]

Saves the model as an XML document.

Parameters:

model The model to save.

filename The name of the file to save to.

Definition at line 86 of file xmldao.hpp.

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

7.23.3.8 `Category* XmlDao::get_category (long cat_id, item_choice choice)` [inline, virtual]

Loads a category.

Parameters:

choice 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.

7.23.3.9 `ItemContainer* XmlDao::get_all_items (item_choice choice)` [inline, virtual]

Returns a list of all items.

Parameters:

choice Which items to load. One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Implements [KitListDao](#).

Definition at line 90 of file xmldao.hpp.

References NYI.

7.23.3.10 `long XmlDao::add_item (const std::string name)` [inline, virtual]

Creates a new item.

Parameters:

name The name of the new item.

Implements [KitListDao](#).

Definition at line 92 of file xmldao.hpp.

References NYI.

7.23.3.11 `long XmlDao::add_item (const std::string name, long cat_id)` [inline, virtual]

Creates a new item and associates it with a category.

Parameters:

name The name of the new item.

Implements [KitListDao](#).

Definition at line 94 of file xmldao.hpp.

References NYI.

7.23.3.12 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:

from_cat_id The ID of the source category.

to_cat_id The ID of the target category.

choice One of ALL_ITEMS, CHECKED_ITEMS or UNCHECKED_ITEMS.

Implements [KitListDao](#).

Definition at line 96 of file xmldao.hpp.

References NYI.

7.23.3.13 void XmlDao::associate_item_with_category (long id, long cat_id) [inline, virtual]

Associates an existing item with an existing category.

Parameters:

id The [Item](#) ID

cat_id The [Category](#) ID

Implements [KitListDao](#).

Definition at line 98 of file xmldao.hpp.

References NYI.

7.23.3.14 CategoryContainer XmlDao::get_categories () [inline, virtual]

Returns a list of all categories.

Implements [KitListDao](#).

Definition at line 100 of file xmldao.hpp.

References NYI.

7.23.3.15 long XmlDao::new_category (const std::string name) [inline, virtual]

Creates a new category.

Parameters:

name the name of the new category.

Implements [KitListDao](#).

Definition at line 102 of file xmldao.hpp.

References NYI.

7.23.3.16 void XmlDao::delete_item (long id) [inline, virtual]

Deletes an item by it's ID.

Parameters:

id the ID of the item to delete.

Implements [KitListDao](#).

Definition at line 104 of file xmldao.hpp.

References NYI.

7.23.3.17 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.

7.23.3.18 void XmlDao::remove_item_from_category (long id, long cat_id) [inline, virtual]

Un-associates the specified item from the specified category.

Parameters:

id The [Item](#) id.

cat_id The [Category](#) id.

Implements [KitListDao](#).

Definition at line 108 of file xmldao.hpp.

References NYI.

7.23.3.19 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.

References [m_max_item_id](#).

7.23.3.20 `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`.

References `m_max_category_id`.

7.23.3.21 `void XmlDao::delete_category (long id)` [inline, virtual]

Deletes a category.

Implements [KitListDao](#).

Definition at line 114 of file `xmldao.hpp`.

References NYI.

7.23.3.22 `void XmlDao::set_item_flag (long id)` [inline, virtual]

Sets the checked flag for an item.

Parameters:

id The id of the item to change.

Implements [KitListDao](#).

Definition at line 116 of file `xmldao.hpp`.

References NYI.

7.23.3.23 `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.

7.23.3.24 `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 `xmldao.hpp`.

References NYI.

7.23.3.25 `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.

7.23.3.26 void XmlDao::set_all_flags () [inline, virtual]

Checks/ticks all items.

Implements [KitListDao](#).

Definition at line 124 of file xmldao.hpp.

References NYI.

7.23.3.27 void XmlDao::unset_all_flags () [inline, virtual]

Unchecks/unticks all items.

Implements [KitListDao](#).

Definition at line 126 of file xmldao.hpp.

References NYI.

7.23.3.28 void XmlDao::set_filename (Glib::ustring filename) [inline]

Definition at line 128 of file xmldao.hpp.

References `m_filename`.

Referenced by `get_model()`, and `save_model()`.

7.23.3.29 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.

7.23.4 Member Data Documentation

7.23.4.1 Glib::ustring XmlDao::m_filename [protected]

The filename to load or save the XML document from.

Definition at line 54 of file xmldao.hpp.

Referenced by `get_model()`, `save_model()`, and `set_filename()`.

7.23.4.2 xmlpp::Element* XmlDao::m_items_node [protected]

Temporary reference to the items' node.

Definition at line 56 of file xmldao.hpp.

Referenced by `add_item_to_dom()`, and `save_model()`.

7.23.4.3 xmlpp::Element* XmlDao::m_categories_node [protected]

Temporary reference to the categories' node.

Definition at line 58 of file xmldao.hpp.

Referenced by `add_category_to_dom()`, and `save_model()`.

7.23.4.4 xmlpp::Element* XmlDao::m_cat_items_node [protected]

Temporary reference to the categories' items' node.

Definition at line 60 of file xmldao.hpp.

Referenced by `add_category_item_to_dom()`, and `add_category_to_dom()`.

7.23.4.5 long XmlDao::m_max_item_id [protected]

The last used ID for items.

Definition at line 62 of file xmldao.hpp.

Referenced by `get_model()`, and `get_next_item_id()`.

7.23.4.6 long XmlDao::m_max_category_id [protected]

The last used ID for categories.

Definition at line 64 of file xmldao.hpp.

Referenced by `get_model()`, and `get_next_category_id()`.

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](#)

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.

Defines

- #define [CATEGORY_H](#) 1

Typedefs

- typedef std::vector< [Category](#) * > [CategoryContainer](#)
- typedef [CategoryContainer](#)::iterator [CategoryIter](#)

8.2.1 Define Documentation

8.2.1.1 #define CATEGORY_H 1

Definition at line 24 of file category.hpp.

8.2.2 Typedef Documentation

8.2.2.1 typedef std::vector<Category*> CategoryContainer

Definition at line 84 of file category.hpp.

8.2.2.2 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](#).

Defines

- #define [ITEM_H](#) 1

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 Define Documentation

8.3.1.1 #define ITEM_H 1

Definition at line 24 of file item.hpp.

8.3.2 Typedef Documentation

8.3.2.1 typedef std::vector<Item*> ItemContainer

Definition at line 91 of file item.hpp.

8.3.2.2 typedef ItemContainer::iterator ItemIter

Definition at line 92 of file item.hpp.

8.3.2.3 typedef std::list<Item> ItemList

Definition at line 94 of file item.hpp.

8.3.2.4 typedef ItemList::iterator ItemListIter

Definition at line 95 of file item.hpp.

8.3.2.5 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.

Defines

- #define [KIT_LIST_DAO_H](#) 1

Enumerations

- enum [item_choice](#) { [ALL_ITEMS](#), [CHECKED_ITEMS](#), [UNCHECKED_ITEMS](#) }

8.4.1 Define Documentation

8.4.1.1 #define KIT_LIST_DAO_H 1

Definition at line 24 of file kitlistdao.hpp.

8.4.2 Enumeration Type Documentation

8.4.2.1 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 <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 <libglademm/xml.h>
#include <libxml++/libxml++.h>
#include <string>
#include <sstream>
#include <sys/stat.h>
#include <config.h>
```

Namespaces

- namespace [anonymous_namespace{kitlistgui.cpp}](#)

Defines

- #define [KITLIST_SERVICE_NAME](#) "com.nokia."PACKAGE_NAME
Maemo service name.
- #define [KITLIST_SERVICE_OBJECT](#) "/com/nokia/"PACKAGE_NAME
Maemo service path.
- #define [KITLIST_SERVICE_IFACE](#) "com.nokia."PACKAGE_NAME
Maemo service interface name.

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 gint anonymous_namespace{kitlistgui.cpp}::DEFAULT_MAX_RECENT_FILES = 4`

The maximum number of recent files to maintain.

- `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 Define Documentation

8.5.1.1 `#define KITLIST_SERVICE_IFACE "com.nokia."PACKAGE_NAME`

Maemo service interface name.

Definition at line 55 of file kitlistgui.cpp.

Referenced by KitListGui::init().

8.5.1.2 `#define KITLIST_SERVICE_NAME "com.nokia."PACKAGE_NAME`

Maemo service name.

Definition at line 51 of file kitlistgui.cpp.

Referenced by KitListGui::init().

8.5.1.3 `#define KITLIST_SERVICE_OBJECT "/com/nokia/"PACKAGE_NAME`

Maemo service path.

Definition at line 53 of file kitlistgui.cpp.

Referenced by KitListGui::init().

8.5.2 Typedef Documentation

8.5.2.1 typedef `Gtk::TreeModel::Children` `type_children`

Definition at line 146 of file `kitlistgui.cpp`.

8.5.3 Function Documentation

8.5.3.1 `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:

filename The file to test for existence.

Returns:

true if the file exists.

Definition at line 157 of file `kitlistgui.cpp`.

Referenced by `KitListGui::choose_filename()`, `KitListGui::choose_pdf_filename()`, `KitListGui::init()`, and `load_resource_glade_file()`.

8.5.3.2 `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 186 of file `kitlistgui.cpp`.

References `load_resource_glade_file()`.

Referenced by `KitListGui::init()`.

8.5.3.3 `const string load_resource_glade_file (const Glib::ustring & filename)`

Locate the Glade resource file from a list of potential locations.

Definition at line 167 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 <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 [ModelItemColumns](#)
A definition for displaying an item in a multi-column list.
- class [KitListGui](#)
Encapsulates the methods for the application's GUI front end.

Defines

- #define [KIT_LIST_GUI_H](#) 1

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 Define Documentation

8.6.1.1 #define KIT_LIST_GUI_H 1

Definition at line 24 of file kitlistgui.hpp.

8.6.2 Enumeration Type Documentation

8.6.2.1 enum gui_state

Enumerator:

ADD_CATEGORY
RENAME_CATEGORY

Definition at line 54 of file kitlistgui.hpp.

8.6.3 Variable Documentation

8.6.3.1 const int CHECKED_COL_POSITION = 0

The position in the list of the tick box column.

Definition at line 73 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>
```

Defines

- #define [KIT_LIST_PGSQL_DAO_H](#) 1

8.8.1 Define Documentation

8.8.1.1 #define KIT_LIST_PGSQL_DAO_H 1

Definition at line 24 of file kitlistpgsqldao.hpp.

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 [ModelItem](#)
Represents an [Item](#) combined with [GuiState](#) attributes.
- class [ModelItemCompareId](#)
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.

Defines

- #define [KIT_MODEL_H](#) 1

Typedefs

- typedef std::vector< [ModelItem](#) * > [ModelItemContainer](#)
- typedef ModelItemContainer::iterator [ModelItemIter](#)
- typedef std::map< long, [ModelItem](#) * > [ItemMap](#)
- typedef ItemMap::iterator [ItemMapIter](#)
- typedef sigc::slot< bool, const ItemMap::iterator & > [SlotForeachModelItemIter](#)
- typedef sigc::slot< bool, [ModelItem](#) & > [SlotForeachModelItem](#)
- 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 Define Documentation

8.10.1.1 `#define KIT_MODEL_H 1`

Definition at line 24 of file kitmodel.hpp.

8.10.2 Typedef Documentation

8.10.2.1 `typedef std::map<long, ModelCategory*> CategoryMap`

Definition at line 122 of file kitmodel.hpp.

8.10.2.2 `typedef CategoryMap::iterator CategoryMapIter`

Definition at line 123 of file kitmodel.hpp.

8.10.2.3 `typedef std::map<long, ModelItem*> ItemMap`

Definition at line 84 of file kitmodel.hpp.

8.10.2.4 `typedef ItemMap::iterator ItemMapIter`

Definition at line 85 of file kitmodel.hpp.

8.10.2.5 `typedef std::vector<ModelCategory*> ModelCategoryContainer`

Definition at line 119 of file kitmodel.hpp.

8.10.2.6 `typedef ModelCategoryContainer::iterator ModelCategoryIter`

Definition at line 120 of file kitmodel.hpp.

8.10.2.7 `typedef std::vector<ModelItem*> ModelItemContainer`

Definition at line 81 of file kitmodel.hpp.

8.10.2.8 `typedef ModelItemContainer::iterator ModelItemIter`

Definition at line 82 of file kitmodel.hpp.

8.10.2.9 `typedef sigc::slot<bool, ModelCategory&> SlotForeachCategory`

Definition at line 126 of file kitmodel.hpp.

8.10.2.10 `typedef sigc::slot<bool, const CategoryMap::iterator&> SlotForeachCategoryIter`

Definition at line 125 of file kitmodel.hpp.

8.10.2.11 typedef sigc::slot<bool, ModelItem> SlotForeachModelItem

Definition at line 88 of file kitmodel.hpp.

8.10.2.12 typedef sigc::slot<bool, const ItemMap::iterator> SlotForeachModelItemIter

Definition at line 87 of file kitmodel.hpp.

8.10.3 Enumeration Type Documentation**8.10.3.1 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.

Defines

- #define [KIT_PARSER_H](#) 1

8.12.1 Define Documentation

8.12.1.1 #define KIT_PARSER_H 1

Definition at line 24 of file kitparser.hpp.

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 "kitlistpgsqldao.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 int main (int argc, char ** argv)

Parses the command line and executes the main application.

Definition at line 564 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 `int html_flag = 0` [static]

Flag set by `'-html'`.

Definition at line 74 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 `int verbose_flag = 0` [static]

Flag set by `'-verbose'`.

Definition at line 71 of file main.cpp.

Referenced by `main()`.

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 `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 `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 `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 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 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 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.

Defines

- #define [SERVICE_H 1](#)

8.17.1 Define Documentation

8.17.1.1 #define SERVICE_H 1

Definition at line 24 of file service.hpp.

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.

Defines

- #define [XMLDAO_H](#) 1
- #define [NYI](#) assert(false == "Method not implemented")

8.19.1 Define Documentation

8.19.1.1 #define NYI assert(false == "Method not implemented")

Definition at line 35 of file xmldao.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 #define XMLDAO_H 1

Definition at line 28 of file xmldao.hpp.

Index

- ~Category
 - Category, 18
- ~KitList
 - KitList, 36
- ~KitListDao
 - KitListDao, 44
- ~KitListGui
 - KitListGui, 55
- ~KitModel
 - KitModel, 77
- ~KitParser
 - KitParser, 84
- ~KitPrintOperation
 - KitPrintOperation, 89
- ~ModelCategory
 - ModelCategory, 93
- ~Service
 - Service, 105
- /home/frank/projects/kitlist/src/category.cpp, 125
- /home/frank/projects/kitlist/src/category.hpp, 126
- /home/frank/projects/kitlist/src/item.hpp, 127
- /home/frank/projects/kitlist/src/kitlistdao.hpp, 129
- /home/frank/projects/kitlist/src/kitlistgui.cpp, 130
- /home/frank/projects/kitlist/src/kitlistgui.hpp, 134
- /home/frank/projects/kitlist/src/kitlistpgsqldao.cpp, 136
- /home/frank/projects/kitlist/src/kitlistpgsqldao.hpp, 137
- /home/frank/projects/kitlist/src/kitmodel.cpp, 138
- /home/frank/projects/kitlist/src/kitmodel.hpp, 139
- /home/frank/projects/kitlist/src/kitparser.cpp, 142
- /home/frank/projects/kitlist/src/kitparser.hpp, 143
- /home/frank/projects/kitlist/src/main.cpp, 144
- /home/frank/projects/kitlist/src/printing.cpp, 146
- /home/frank/projects/kitlist/src/printing.hpp, 148
- /home/frank/projects/kitlist/src/service.cpp, 149
- /home/frank/projects/kitlist/src/service.hpp, 150
- /home/frank/projects/kitlist/src/xmldao.cpp, 151
- /home/frank/projects/kitlist/src/xmldao.hpp, 152
- ADD_CATEGORY
 - kitlistgui.hpp, 135
- add_category
 - KitModel, 78
- add_category_item_to_dom
 - XmlDao, 116
- add_category_to_dom
 - XmlDao, 116
- add_item
 - Category, 19
 - KitList, 36, 37
 - KitListDao, 46
 - KitModel, 79
 - ModelCategory, 94
 - XmlDao, 118
- add_item_to_dom
 - XmlDao, 116
- add_items
 - KitListGui, 56
- ALL
 - kitmodel.hpp, 141
- ALL_ITEMS
 - kitlistdao.hpp, 129
- anonymous_namespace{kitlistgui.cpp}, 11
 - DEFAULT_FILENAME, 12
 - DEFAULT_FILENAME_EXTENSION, 12
 - DEFAULT_MAX_RECENT_FILES, 12
 - GCONF_KEY, 12
 - GCONF_KEY_CURRENT_FILENAME, 13
 - GCONF_KEY_MAX_RECENT_FILES, 13
 - GCONF_KEY_PAGE_TITLE, 13
 - GCONF_KEY_RECENT_FILES, 13
 - 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, 14
 - SB_SAVE, 14
 - XML_ELEMENT_ID, 15
- append_items_to_category
 - KitList, 37
 - KitListDao, 46
 - XmlDao, 119
- associate_item_with_category
 - KitList, 37
 - KitListDao, 46
 - XmlDao, 119

- BORDER_SPACING
 - printing.cpp, 146
- cancel_add_category_window
 - KitListGui, 57
- cancel_add_item_window
 - KitListGui, 57
- cancel_preferences_window
 - KitListGui, 57
- Category, 17
 - ~Category, 18
 - add_item, 19
 - CategoryCompareId, 20
 - CategoryCompareName, 20
 - execute, 20
 - foreach_item, 20
 - get_id, 18
 - get_name, 19
 - has_items, 20
 - item_count, 19
 - KitModel, 20
 - m_id, 21
 - m_items, 21
 - m_name, 21
 - remove_item, 19
 - set_id, 18
 - set_name, 19
- category.hpp
 - CATEGORY_H, 126
 - CategoryContainer, 126
 - CategoryIter, 126
- CATEGORY_H
 - category.hpp, 126
- CategoryCompareId, 22
 - Category, 20
 - CategoryCompareId, 22
 - operator(), 22
- CategoryCompareName, 23
 - Category, 20
 - CategoryCompareName, 23
 - operator(), 23
- CategoryContainer
 - category.hpp, 126
- CategoryIter
 - category.hpp, 126
- CategoryMap
 - kitmodel.hpp, 140
- CategoryMapIter
 - kitmodel.hpp, 140
- CHECKED
 - kitmodel.hpp, 141
- CHECKED_ITEMS
 - kitlistdao.hpp, 129
- CHECKED_COL_POSITION
 - kitlistgui.hpp, 135
- choose_filename
 - KitListGui, 66
- choose_pdf_filename
 - KitListGui, 66
- close_add_category_window
 - KitListGui, 57
- close_add_item_window
 - KitListGui, 57
- close_preferences_window
 - KitListGui, 57
- confirm_lose_changes
 - KitListGui, 59
- copy_items
 - KitModel, 79
 - Service, 106
- copy_selected_items_to_clipboard
 - KitListGui, 58
- create
 - KitPrintOperation, 89
- create_category
 - Service, 107
- create_default_model
 - Service, 108
- create_item
 - Service, 106
- DEFAULT_FILENAME
 - anonymous_namespace{kitlistgui.cpp}, 12
- DEFAULT_FILENAME_EXTENSION
 - anonymous_namespace{kitlistgui.cpp}, 12
- DEFAULT_MAX_RECENT_FILES
 - anonymous_namespace{kitlistgui.cpp}, 12
- delete_category
 - KitList, 40
 - KitListDao, 48
 - Service, 106
 - XmlDao, 121
- delete_item
 - KitList, 40
 - KitListDao, 47
 - Service, 106
 - XmlDao, 120
- delete_selected_items
 - KitListGui, 58
- execute
 - Category, 20
 - KitList, 39
- file_exists
 - kitlistgui.cpp, 133
- filter
 - KitModel, 79

- Service, 107
- FilterItem, 24
 - FilterItem, 24
 - m_model, 24
 - operator(), 24
- find_category
 - KitModel, 77
 - Service, 105
- find_item
 - KitModel, 78
 - Service, 105
- FOOTER_SPACING
 - printing.cpp, 146
- FOOTER_TEXT
 - printing.cpp, 146
- foreach_category
 - KitModel, 77
- foreach_category_iter
 - KitModel, 77
- foreach_item
 - Category, 20
 - KitModel, 77
- foreach_item_iter
 - KitModel, 77
- GCONF_KEY
 - anonymous_namespace{kitlistgui.cpp}, 12
- GCONF_KEY_CURRENT_FILENAME
 - anonymous_namespace{kitlistgui.cpp}, 13
- GCONF_KEY_MAX_RECENT_FILES
 - anonymous_namespace{kitlistgui.cpp}, 13
- GCONF_KEY_PAGE_TITLE
 - anonymous_namespace{kitlistgui.cpp}, 13
- GCONF_KEY_RECENT_FILES
 - anonymous_namespace{kitlistgui.cpp}, 13
- get_added_children
 - ModelCategory, 94
- get_all_items
 - KitListDao, 45
 - KitModel, 78
 - XmlDao, 118
- get_categories
 - KitListDao, 47
 - KitModel, 78
 - Service, 109
 - XmlDao, 119
- get_category
 - KitListDao, 45
 - XmlDao, 118
- get_checked
 - Item, 30
- get_dao
 - KitList, 36
- get_description
 - Item, 29
- get_filtered_items
 - Service, 109
- get_glade_ref_ptr
 - kitlistgui.cpp, 133
- get_id
 - Category, 18
 - Item, 29
- get_items
 - ModelCategory, 93
 - Service, 109
- get_max_recent_files
 - KitListGui, 56
- get_model
 - KitListDao, 44
 - XmlDao, 116, 117
- get_model_items
 - ModelCategory, 93
- get_name
 - Category, 19
- get_next_category_id
 - KitListDao, 48
 - Service, 105
 - XmlDao, 120
- get_next_item_id
 - KitListDao, 48
 - Service, 105
 - XmlDao, 120
- get_removed_children
 - ModelCategory, 94
- get_selected_category
 - KitListGui, 58
- get_selected_items
 - KitListGui, 56
- GLADE_APP_FILE
 - anonymous_namespace{kitlistgui.cpp}, 13
- gui_state
 - kitlistgui.hpp, 135
- GuiState, 25
 - GuiState, 25
 - is_deleted, 26
 - is_dirty, 26
 - is_new, 26
 - m_deleted, 27
 - m_dirty, 27
 - m_new, 27
 - reset, 26
 - set_deleted, 26
 - set_dirty, 26
 - set_new_flag, 26
- has_items
 - Category, 20
- HEADER_SPACING

- printing.cpp, 146
- html_flag
 - main.cpp, 145
- init
 - KitListGui, 55
- init_add_item_window
 - KitListGui, 58
- is_deleted
 - GuiState, 26
- is_dirty
 - GuiState, 26
 - KitModel, 80
- is_model_dirty
 - Service, 107
- is_new
 - GuiState, 26
- is_verbose
 - KitListDao, 45
- Item, 28
 - get_checked, 30
 - get_description, 29
 - get_id, 29
 - Item, 29
 - ItemCompareId, 30
 - ItemCompareName, 30
 - m_checked, 30
 - m_desc, 30
 - m_id, 30
 - operator<<, 30
 - set_checked, 29
 - set_description, 29
 - set_id, 29
- item.hpp
 - ITEM_H, 127
 - ItemContainer, 127
 - ItemIter, 127
 - ItemList, 128
 - ItemListIter, 128
 - SlotForeachItem, 128
- item_choice
 - kitlistdao.hpp, 129
- item_count
 - Category, 19
- item_filter_types
 - kitmodel.hpp, 141
- ITEM_H
 - item.hpp, 127
- item_target_custom
 - anonymous_namespace{kitlistgui.cpp}, 13
- item_target_text
 - anonymous_namespace{kitlistgui.cpp}, 14
- ItemCompareId, 32
 - Item, 30
- ItemCompareName, 33
 - Item, 30
 - ItemCompareName, 33
 - operator(), 33
- ItemContainer
 - item.hpp, 127
- ItemFunctor, 34
 - operator(), 34
- ItemIter
 - item.hpp, 127
- ItemList
 - item.hpp, 128
- ItemListIter
 - item.hpp, 128
- ItemMap
 - kitmodel.hpp, 140
- ItemMapIter
 - kitmodel.hpp, 140
- KIT_LIST_DAO_H
 - kitlistdao.hpp, 129
- KIT_LIST_GUI_H
 - kitlistgui.hpp, 135
- KIT_LIST_PGSQL_DAO_H
 - kitlistpgsqldao.hpp, 137
- KIT_MODEL_H
 - kitmodel.hpp, 140
- KIT_PARSER_H
 - kitparser.hpp, 143
- KitList, 35
 - ~KitList, 36
 - add_item, 36, 37
 - append_items_to_category, 37
 - associate_item_with_category, 37
 - delete_category, 40
 - delete_item, 40
 - execute, 39
 - get_dao, 36
 - KitList, 36
 - list_categories, 40
 - list_item, 38
 - list_items, 38, 39
 - list_items_end, 38
 - list_items_start, 37
 - m_dao, 42
 - new_category, 40
 - on_list_item, 38
 - remove_item_from_category, 41
 - set_all_flags, 42
 - set_category_flag, 41
 - set_item_flag, 41
 - tick_items, 39, 40

- unset_all_flags, 42
- unset_category_flag, 41
- unset_item_flag, 41
- KITLIST_SERVICE_IFACE
 - kitlistgui.cpp, 132
- KITLIST_SERVICE_NAME
 - kitlistgui.cpp, 132
- KITLIST_SERVICE_OBJECT
 - kitlistgui.cpp, 132
- KitListDao, 43
 - ~KitListDao, 44
 - add_item, 46
 - append_items_to_category, 46
 - associate_item_with_category, 46
 - delete_category, 48
 - delete_item, 47
 - get_all_items, 45
 - get_categories, 47
 - get_category, 45
 - get_model, 44
 - get_next_category_id, 48
 - get_next_item_id, 48
 - is_verbose, 45
 - KitListDao, 44
 - m_verbose_flag, 49
 - new_category, 47
 - remove_item_from_category, 47
 - require_filename, 49
 - save_model, 45
 - set_all_flags, 49
 - set_category_flag, 48
 - set_item_flag, 48
 - set_verbose, 45
 - unset_all_flags, 49
 - unset_category_flag, 48
 - unset_item_flag, 48
 - update_item_checked_state, 47
- kitlistdao.hpp
 - ALL_ITEMS, 129
 - CHECKED_ITEMS, 129
 - item_choice, 129
 - KIT_LIST_DAO_H, 129
 - UNCHECKED_ITEMS, 129
- KitListGui, 50
 - ~KitListGui, 55
 - add_items, 56
 - cancel_add_category_window, 57
 - cancel_add_item_window, 57
 - cancel_preferences_window, 57
 - choose_filename, 66
 - choose_pdf_filename, 66
 - close_add_category_window, 57
 - close_add_item_window, 57
 - close_preferences_window, 57
 - confirm_lose_changes, 59
 - copy_selected_items_to_clipboard, 58
 - delete_selected_items, 58
 - get_max_recent_files, 56
 - get_selected_category, 58
 - get_selected_items, 56
 - init, 55
 - init_add_item_window, 58
 - KitListGui, 55
 - m_category_cols, 73
 - m_category_combo, 72
 - m_checkbox_add_item, 72
 - m_clipboard_items, 70
 - m_current_cat_id, 74
 - m_entry_add_category, 71
 - m_entry_add_item, 71
 - m_entry_page_title, 71
 - m_file_save_menu_item, 71
 - m_file_save_tool_button, 72
 - m_filename, 70
 - m_ignore_list_events, 70
 - m_item_cols, 73
 - m_item_tree_view, 73
 - m_kit, 70
 - m_page_title, 70
 - m_paste_menu_item, 72
 - m_paste_tool_button, 72
 - m_recent_files_menu_item, 72
 - m_ref_category_list_store, 72
 - m_ref_item_tree_model, 73
 - m_ref_page_setup, 74
 - m_ref_printer_settings, 74
 - m_service, 73
 - m_state, 74
 - m_status_bar, 73
 - m_window, 70
 - m_window_add_category, 71
 - m_window_add_item, 71
 - m_window_preferences, 71
 - on_category_change, 65
 - on_cell_edit, 66
 - on_clipboard_clear, 65
 - on_clipboard_get, 65
 - on_clipboard_received, 65
 - on_delete_event, 59
 - on_menu_add, 62
 - on_menu_check_selected, 63
 - on_menu_copy, 62
 - on_menu_create_category, 64
 - on_menu_cut, 62
 - on_menu_delete, 62
 - on_menu_delete_category, 64
 - on_menu_export_to_pdf, 61
 - on_menu_file_new, 60

- on_menu_file_open, 60
- on_menu_help_about, 64
- on_menu_help_contents, 65
- on_menu_paste, 62
- on_menu_preferences, 61
- on_menu_print, 61
- on_menu_quit, 59
- on_menu_recent_file, 61
- on_menu_rename_category, 64
- on_menu_save, 60
- on_menu_save_as, 60
- on_menu_select_all, 63
- on_menu_show_all, 63
- on_menu_show_checked, 63
- on_menu_show_unchecked, 63
- on_menu_uncheck_selected, 64
- on_printoperation_done, 60
- on_printoperation_status_changed, 61
- on_row_changed, 68
- open_file, 69
- paste_from_xml, 67
- paste_status_received, 67
- raise, 69
- refresh_category_list, 67
- refresh_item_list, 67
- run, 69
- safe_open_file, 69
- selected_row_callback, 68
- set_page_title, 56
- set_selected, 68
- toggle_selected, 68
- update_item_count, 69
- update_paste_status, 66
- update_recent_files, 59
- update_recent_files_menu, 59
- kitlistgui.cpp
 - file_exists, 133
 - get_glade_ref_ptr, 133
 - KITLIST_SERVICE_IFACE, 132
 - KITLIST_SERVICE_NAME, 132
 - KITLIST_SERVICE_OBJECT, 132
 - load_resource_glade_file, 133
 - type_children, 133
- kitlistgui.hpp
 - ADD_CATEGORY, 135
 - CHECKED_COL_POSITION, 135
 - gui_state, 135
 - KIT_LIST_GUI_H, 135
 - RENAME_CATEGORY, 135
- kitlistpgsqldao.hpp
 - KIT_LIST_PGSQL_DAO_H, 137
- KitModel, 75
 - ~KitModel, 77
 - add_category, 78
 - add_item, 79
 - Category, 20
 - copy_items, 79
 - filter, 79
 - find_category, 77
 - find_item, 78
 - foreach_category, 77
 - foreach_category_iter, 77
 - foreach_item, 77
 - foreach_item_iter, 77
 - get_all_items, 78
 - get_categories, 78
 - is_dirty, 80
 - KitModel, 77
 - m_category_map, 81
 - m_dirty, 81
 - m_item_filter, 81
 - m_item_map, 81
 - ModelCategory, 95
 - purge, 81
 - reset, 80
 - set_dirty, 79
 - show_all, 80
 - show_checked_only, 80
 - show_unchecked_only, 80
- kitmodel.hpp
 - ALL, 141
 - CategoryMap, 140
 - CategoryMapIter, 140
 - CHECKED, 141
 - item_filter_types, 141
 - ItemMap, 140
 - ItemMapIter, 140
 - KIT_MODEL_H, 140
 - ModelCategoryContainer, 140
 - ModelCategoryIter, 140
 - ModelItemContainer, 140
 - ModelItemIter, 140
 - SlotForeachCategory, 140
 - SlotForeachCategoryIter, 140
 - SlotForeachModelItem, 140
 - SlotForeachModelItemIter, 141
 - UNCHECKED, 141
- KitParser, 83
 - ~KitParser, 84
 - KitParser, 84
 - m_category, 87
 - m_cdata, 87
 - m_item, 87
 - m_model, 87
 - on_characters, 86
 - on_comment, 86
 - on_end_document, 85
 - on_end_element, 85

- on_error, 86
- on_fatal_error, 86
- on_start_document, 85
- on_start_element, 85
- on_warning, 86
- process_category, 85
- process_category_item, 85
- process_item, 84
- kitparser.hpp
 - KIT_PARSER_H, 143
- KitPrintOperation, 88
 - ~KitPrintOperation, 89
 - create, 89
 - m_items, 90
 - m_page_breaks, 91
 - m_page_title, 90
 - m_ref_footers, 91
 - m_ref_headers, 91
 - m_ref_layout, 91
 - new_footer, 89
 - new_header, 89
 - on_begin_print, 90
 - on_draw_page, 90
 - set_items, 89
 - set_page_title, 89
- layout_refptr
 - printing.hpp, 148
- list_categories
 - KitList, 40
- list_item
 - KitList, 38
- list_items
 - KitList, 38, 39
- list_items_end
 - KitList, 38
- list_items_start
 - KitList, 37
- load_model
 - Service, 105
- load_resource_glade_file
 - kitlistgui.cpp, 133
- m_added_children
 - ModelCategory, 95
- m_cat_items_node
 - XmlDao, 123
- m_categories_node
 - XmlDao, 123
- m_category
 - KitParser, 87
- m_category_cols
 - KitListGui, 73
- m_category_combo
 - KitListGui, 72
- m_category_map
 - KitModel, 81
- m_cdata
 - KitParser, 87
- m_changed
 - TickItem, 112
- m_checkbutton_add_item
 - KitListGui, 72
- m_checked
 - Item, 30
- m_clipboard_items
 - KitListGui, 70
- m_col_checked
 - ModelItemColumns, 100
- m_col_num
 - ModelCategoryColumns, 97
 - ModelItemColumns, 100
- m_col_text
 - ModelCategoryColumns, 97
 - ModelItemColumns, 100
- m_current_cat_id
 - KitListGui, 74
- m_dao
 - KitList, 42
 - Service, 111
- m_deleted
 - GuiState, 27
- m_desc
 - Item, 30
- m_dirty
 - GuiState, 27
 - KitModel, 81
- m_entry_add_category
 - KitListGui, 71
- m_entry_add_item
 - KitListGui, 71
- m_entry_page_title
 - KitListGui, 71
- m_file_save_menu_item
 - KitListGui, 71
- m_file_save_tool_button
 - KitListGui, 72
- m_filename
 - KitListGui, 70
 - XmlDao, 122
- m_id
 - Category, 21
 - Item, 30
- m_ignore_list_events
 - KitListGui, 70
- m_item
 - KitParser, 87
- m_item_cols

- KitListGui, 73
- m_item_filter
 - KitModel, 81
- m_item_map
 - KitModel, 81
- m_item_tree_view
 - KitListGui, 73
- m_items
 - Category, 21
 - KitPrintOperation, 90
- m_items_node
 - XmlDao, 122
- m_kit
 - KitListGui, 70
- m_max_category_id
 - XmlDao, 123
- m_max_item_id
 - XmlDao, 123
- m_model
 - FilterItem, 24
 - KitParser, 87
 - Service, 111
- m_name
 - Category, 21
- m_new
 - GuiState, 27
- m_page_breaks
 - KitPrintOperation, 91
- m_page_title
 - KitListGui, 70
 - KitPrintOperation, 90
- m_paste_menu_item
 - KitListGui, 72
- m_paste_tool_button
 - KitListGui, 72
- m_recent_files_menu_item
 - KitListGui, 72
- m_ref_category_list_store
 - KitListGui, 72
- m_ref_footers
 - KitPrintOperation, 91
- m_ref_headers
 - KitPrintOperation, 91
- m_ref_item_tree_model
 - KitListGui, 73
- m_ref_layout
 - KitPrintOperation, 91
- m_ref_page_setup
 - KitListGui, 74
- m_ref_printer_settings
 - KitListGui, 74
- m_removed_children
 - ModelCategory, 95
- m_service
 - KitListGui, 73
- m_state
 - KitListGui, 74
- m_status_bar
 - KitListGui, 73
- m_verbose_flag
 - KitListDao, 49
- m_window
 - KitListGui, 70
- m_window_add_category
 - KitListGui, 71
- m_window_add_item
 - KitListGui, 71
- m_window_preferences
 - KitListGui, 71
- main
 - main.cpp, 144
- main.cpp
 - html_flag, 145
 - main, 144
 - verbose_flag, 145
- ModelCategory, 92
 - ~ModelCategory, 93
 - add_item, 94
 - get_added_children, 94
 - get_items, 93
 - get_model_items, 93
 - get_removed_children, 94
 - KitModel, 95
 - m_added_children, 95
 - m_removed_children, 95
 - ModelCategory, 93
 - purge, 95
 - remove_item, 94
 - remove_items, 94
 - reset, 95
- ModelCategoryColumns, 97
 - m_col_num, 97
 - m_col_text, 97
 - ModelCategoryColumns, 97
- ModelCategoryContainer
 - kitmodel.hpp, 140
- ModelCategoryIter
 - kitmodel.hpp, 140
- ModelItem, 98
 - ModelItem, 98
 - ModelItemCompareId, 99
 - set_checked, 98
- ModelItemColumns, 100
 - m_col_checked, 100
 - m_col_num, 100
 - m_col_text, 100
 - ModelItemColumns, 100
- ModelItemCompareId, 102

- ModellItem, 99
- ModellItemCompareId, 102
- operator(), 102
- ModellItemContainer
 - kitmodel.hpp, 140
- ModellItemIter
 - kitmodel.hpp, 140
- new_category
 - KitList, 40
 - KitListDao, 47
 - XmlDao, 119
- new_footer
 - KitPrintOperation, 89
- new_header
 - KitPrintOperation, 89
- NYI
 - xmldao.hpp, 152
- on_begin_print
 - KitPrintOperation, 90
- on_category_change
 - KitListGui, 65
- on_cell_edit
 - KitListGui, 66
- on_characters
 - KitParser, 86
- on_clipboard_clear
 - KitListGui, 65
- on_clipboard_get
 - KitListGui, 65
- on_clipboard_received
 - KitListGui, 65
- on_comment
 - KitParser, 86
- on_delete_event
 - KitListGui, 59
- on_draw_page
 - KitPrintOperation, 90
- on_end_document
 - KitParser, 85
- on_end_element
 - KitParser, 85
- on_error
 - KitParser, 86
- on_fatal_error
 - KitParser, 86
- on_list_item
 - KitList, 38
- on_menu_add
 - KitListGui, 62
- on_menu_check_selected
 - KitListGui, 63
- on_menu_copy
 - KitListGui, 62
- on_menu_create_category
 - KitListGui, 64
- on_menu_cut
 - KitListGui, 62
- on_menu_delete
 - KitListGui, 62
- on_menu_delete_category
 - KitListGui, 64
- on_menu_export_to_pdf
 - KitListGui, 61
- on_menu_file_new
 - KitListGui, 60
- on_menu_file_open
 - KitListGui, 60
- on_menu_help_about
 - KitListGui, 64
- on_menu_help_contents
 - KitListGui, 65
- on_menu_paste
 - KitListGui, 62
- on_menu_preferences
 - KitListGui, 61
- on_menu_print
 - KitListGui, 61
- on_menu_quit
 - KitListGui, 59
- on_menu_recent_file
 - KitListGui, 61
- on_menu_rename_category
 - KitListGui, 64
- on_menu_save
 - KitListGui, 60
- on_menu_save_as
 - KitListGui, 60
- on_menu_select_all
 - KitListGui, 63
- on_menu_show_all
 - KitListGui, 63
- on_menu_show_checked
 - KitListGui, 63
- on_menu_show_unchecked
 - KitListGui, 63
- on_menu_uncheck_selected
 - KitListGui, 64
- on_printoperation_done
 - KitListGui, 60
- on_printoperation_status_changed
 - KitListGui, 61
- on_row_changed
 - KitListGui, 68
- on_start_document
 - KitParser, 85
- on_start_element

- KitParser, 85
- on_warning
 - KitParser, 86
- open_as_xml
 - Service, 108
- open_file
 - KitListGui, 69
- operator<<
 - Item, 30
- operator()
 - CategoryCompareId, 22
 - CategoryCompareName, 23
 - FilterItem, 24
 - ItemCompareId, 32
 - ItemCompareName, 33
 - ItemFunctor, 34
 - ModelItemCompareId, 102
 - TickItem, 112
- PAGE_TOLERANCE
 - printing.cpp, 147
- paste_from_xml
 - KitListGui, 67
- paste_status_received
 - KitListGui, 67
- PDF_FILENAME_EXTENSION
 - anonymous_namespace{kitlistgui.cpp}, 14
- printing.cpp
 - BORDER_SPACING, 146
 - FOOTER_SPACING, 146
 - FOOTER_TEXT, 146
 - HEADER_SPACING, 146
 - PAGE_TOLERANCE, 147
- printing.hpp
 - layout_refptr, 148
- process_category
 - KitParser, 85
- process_category_item
 - KitParser, 85
- process_item
 - KitParser, 84
- purge
 - KitModel, 81
 - ModelCategory, 95
- raise
 - KitListGui, 69
- refresh_category_list
 - KitListGui, 67
- refresh_item_list
 - KitListGui, 67
- remove_item
 - Category, 19
 - ModelCategory, 94
- remove_item_from_category
 - KitList, 41
 - KitListDao, 47
 - XmlDao, 120
- remove_items
 - ModelCategory, 94
- RENAME_CATEGORY
 - kitlistgui.hpp, 135
- require_filename
 - KitListDao, 49
 - Service, 111
 - XmlDao, 122
- reset
 - GuiState, 26
 - KitModel, 80
 - ModelCategory, 95
- run
 - KitListGui, 69
- safe_open_file
 - KitListGui, 69
- save
 - Service, 108
- save_as_xml
 - Service, 108
- save_model
 - KitListDao, 45
 - XmlDao, 117
- SB_ITEM_COUNT
 - anonymous_namespace{kitlistgui.cpp}, 14
- SB_MSG
 - anonymous_namespace{kitlistgui.cpp}, 14
- SB_PRINT
 - anonymous_namespace{kitlistgui.cpp}, 14
- SB_SAVE
 - anonymous_namespace{kitlistgui.cpp}, 14
- select_items
 - Service, 110
- selected_row_callback
 - KitListGui, 68
- Service, 103
 - ~Service, 105
 - copy_items, 106
 - create_category, 107
 - create_default_model, 108
 - create_item, 106
 - delete_category, 106
 - delete_item, 106
 - filter, 107
 - find_category, 105
 - find_item, 105
 - get_categories, 109
 - get_filtered_items, 109
 - get_items, 109

- get_next_category_id, 105
- get_next_item_id, 105
- is_model_dirty, 107
- load_model, 105
- m_dao, 111
- m_model, 111
- open_as_xml, 108
- require_filename, 111
- save, 108
- save_as_xml, 108
- select_items, 110
- Service, 105
- set_model_dirty, 107
- show_all, 110
- show_checked_only, 110
- show_unchecked_only, 110
- toggle_selected_items, 110
- update_item, 108
- service.hpp
 - SERVICE_H, 150
- SERVICE_H
 - service.hpp, 150
- set_all_flags
 - KitList, 42
 - KitListDao, 49
 - XmlDao, 122
- set_category_flag
 - KitList, 41
 - KitListDao, 48
 - XmlDao, 121
- set_checked
 - Item, 29
 - ModelItem, 98
- set_deleted
 - GuiState, 26
- set_description
 - Item, 29
- set_dirty
 - GuiState, 26
 - KitModel, 79
- set_filename
 - XmlDao, 122
- set_id
 - Category, 18
 - Item, 29
- set_item_flag
 - KitList, 41
 - KitListDao, 48
 - XmlDao, 121
- set_items
 - KitPrintOperation, 89
- set_model_dirty
 - Service, 107
- set_name
 - Category, 19
- set_new_flag
 - GuiState, 26
- set_page_title
 - KitListGui, 56
 - KitPrintOperation, 89
- set_selected
 - KitListGui, 68
- set_verbose
 - KitListDao, 45
- show_all
 - KitModel, 80
 - Service, 110
- show_checked_only
 - KitModel, 80
 - Service, 110
- show_unchecked_only
 - KitModel, 80
 - Service, 110
- SlotForeachCategory
 - kitmodel.hpp, 140
- SlotForeachCategoryIter
 - kitmodel.hpp, 140
- SlotForeachItem
 - item.hpp, 128
- SlotForeachModelItem
 - kitmodel.hpp, 140
- SlotForeachModelItemIter
 - kitmodel.hpp, 141
- tick_items
 - KitList, 39, 40
- TickItem, 112
 - m_changed, 112
 - operator(), 112
 - TickItem, 112
- toggle_selected
 - KitListGui, 68
- toggle_selected_items
 - Service, 110
- type_children
 - kitlistgui.cpp, 133
- UNCHECKED
 - kitmodel.hpp, 141
- UNCHECKED_ITEMS
 - kitlistdao.hpp, 129
- unset_all_flags
 - KitList, 42
 - KitListDao, 49
 - XmlDao, 122
- unset_category_flag
 - KitList, 41
 - KitListDao, 48

- XmlDao, 121
- unset_item_flag
 - KitList, 41
 - KitListDao, 48
 - XmlDao, 121
- update_item
 - Service, 108
- update_item_checked_state
 - KitListDao, 47
 - XmlDao, 120
- update_item_count
 - KitListGui, 69
- update_paste_status
 - KitListGui, 66
- update_recent_files
 - KitListGui, 59
- update_recent_files_menu
 - KitListGui, 59

- verbose_flag
 - main.cpp, 145

- XML_ELEMENT_ID
 - anonymous_namespace{kitlistgui.cpp}, 15
- XmlDao, 114
 - add_category_item_to_dom, 116
 - add_category_to_dom, 116
 - add_item, 118
 - add_item_to_dom, 116
 - append_items_to_category, 119
 - associate_item_with_category, 119
 - delete_category, 121
 - delete_item, 120
 - get_all_items, 118
 - get_categories, 119
 - get_category, 118
 - get_model, 116, 117
 - get_next_category_id, 120
 - get_next_item_id, 120
 - m_cat_items_node, 123
 - m_categories_node, 123
 - m_filename, 122
 - m_items_node, 122
 - m_max_category_id, 123
 - m_max_item_id, 123
 - new_category, 119
 - remove_item_from_category, 120
 - require_filename, 122
 - save_model, 117
 - set_all_flags, 122
 - set_category_flag, 121
 - set_filename, 122
 - set_item_flag, 121
 - unset_all_flags, 122
 - unset_category_flag, 121
 - unset_item_flag, 121
 - update_item_checked_state, 120
 - XmlDao, 116
- xmldao.hpp
 - NYI, 152
 - XMLDAO_H, 152
- XMLDAO_H
 - xmldao.hpp, 152