

Meta Modeware

1.0.1

Generated by Doxygen 1.7.0

Fri Jun 18 2010 01:39:32

Contents

1	Main Page	1
1.1	Overview	1
1.2	Some differences in terminology	1
1.3	Source code comments	1
1.4	Software architecture	1
2	Namespace Index	3
2.1	Namespace List	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Namespace Documentation	9
5.1	Ui Namespace Reference	9
5.1.1	Detailed Description	9
6	Class Documentation	11
6.1	AddApplicationDialog Class Reference	11
6.1.1	Detailed Description	11
6.1.2	Constructor & Destructor Documentation	11
6.1.2.1	AddApplicationDialog	11
6.1.2.2	~AddApplicationDialog	12
6.1.3	Member Function Documentation	12
6.1.3.1	changeEvent	12
6.1.3.2	getCPI	12
6.1.3.3	getCPI_high	12
6.1.3.4	getCPI_low	12

6.1.3.5	getFileName	13
6.1.3.6	getJoystickMode	13
6.1.3.7	getName	13
6.1.3.8	getNewProfileNumber	13
6.1.3.9	getProfileForReplaced	13
6.1.3.10	getReplacedApplication	14
6.1.3.11	getWindowTitlePart	14
6.1.3.12	isOk	14
6.2	AddFunctionDialog Class Reference	14
6.2.1	Detailed Description	15
6.2.2	Constructor & Destructor Documentation	15
6.2.2.1	AddFunctionDialog	15
6.2.2.2	~AddFunctionDialog	15
6.2.3	Member Function Documentation	15
6.2.3.1	changeEvent	15
6.2.3.2	eventFilter	15
6.2.3.3	getFunctionName	16
6.2.3.4	getKeypresses	16
6.2.3.5	getType	16
6.2.3.6	isOk	16
6.2.3.7	keyPressEvent	16
6.2.3.8	keyReleaseEvent	16
6.3	ApplicationListModel Class Reference	17
6.3.1	Detailed Description	17
6.3.2	Constructor & Destructor Documentation	17
6.3.2.1	ApplicationListModel	17
6.3.2.2	~ApplicationListModel	17
6.3.3	Member Function Documentation	17
6.3.3.1	data	17
6.3.3.2	headerData	18
6.3.3.3	rowCount	18
6.4	ApplicationObject Class Reference	18
6.4.1	Detailed Description	19
6.4.2	Constructor & Destructor Documentation	20
6.4.2.1	ApplicationObject	20
6.4.2.2	ApplicationObject	20

6.4.2.3	~ApplicationObject	20
6.4.3	Member Function Documentation	20
6.4.3.1	getAutoswitchDisabled	20
6.4.3.2	getCategoryList	20
6.4.3.3	getCPI	20
6.4.3.4	getCPI_high	21
6.4.3.5	getCPI_low	21
6.4.3.6	getDoubleClickSpeed	21
6.4.3.7	getFileName	21
6.4.3.8	getGroupName	21
6.4.3.9	getJoystickMode	21
6.4.3.10	getName	22
6.4.3.11	getProfileNumber	22
6.4.3.12	getScrollWheelLines	22
6.4.3.13	getSensitivity	22
6.4.3.14	getWindowTitlePart	22
6.4.3.15	operator<	22
6.4.3.16	operator=	23
6.4.3.17	operator==	23
6.4.3.18	setAutoswitchDisabled	23
6.4.3.19	setCategoryListByNode	23
6.4.3.20	setCPI	23
6.4.3.21	setCPI_high	24
6.4.3.22	setCPI_low	24
6.4.3.23	setDoubleClickSpeed	24
6.4.3.24	setFileName	24
6.4.3.25	setFunctionBound	25
6.4.3.26	setFunctionListByNode	25
6.4.3.27	setGroupName	25
6.4.3.28	setJoystickMode	26
6.4.3.29	setName	26
6.4.3.30	setProfileNumber	26
6.4.3.31	setScrollWheelLines	26
6.4.3.32	setSensitivity	26
6.4.3.33	setWindowTitlePart	27
6.5	AutoswitchDetailsDialog Class Reference	27

6.5.1	Detailed Description	27
6.5.2	Constructor & Destructor Documentation	27
6.5.2.1	AutoswitchDetailsDialog	27
6.5.2.2	~AutoswitchDetailsDialog	28
6.5.3	Member Function Documentation	28
6.5.3.1	changeEvent	28
6.5.3.2	getAction	28
6.5.3.3	getFileName	28
6.5.3.4	getWindowTitlePart	28
6.6	ButtonBindingObject Class Reference	29
6.6.1	Detailed Description	29
6.6.2	Constructor & Destructor Documentation	29
6.6.2.1	ButtonBindingObject	29
6.6.2.2	ButtonBindingObject	29
6.6.3	Member Function Documentation	29
6.6.3.1	getFlags	29
6.6.3.2	getFunction	30
6.6.3.3	operator=	30
6.6.3.4	operator==	30
6.6.3.5	setFlags	30
6.6.3.6	setFunction	30
6.7	CategoryListModel Class Reference	31
6.7.1	Detailed Description	31
6.7.2	Constructor & Destructor Documentation	31
6.7.2.1	CategoryListModel	31
6.7.3	Member Function Documentation	31
6.7.3.1	data	31
6.7.3.2	headerData	32
6.7.3.3	rowCount	32
6.8	CategoryObject Class Reference	32
6.8.1	Detailed Description	33
6.8.2	Constructor & Destructor Documentation	33
6.8.2.1	CategoryObject	33
6.8.2.2	CategoryObject	33
6.8.2.3	~CategoryObject	33
6.8.3	Member Function Documentation	33

6.8.3.1	getFunctionList	33
6.8.3.2	getName	34
6.8.3.3	operator<	34
6.8.3.4	operator=	34
6.8.3.5	operator==	34
6.8.3.6	setFunctionBound	34
6.8.3.7	setFunctionListByNode	35
6.8.3.8	setName	35
6.9	charToKeyInfo Struct Reference	35
6.10	CopyApplicationDialog Class Reference	35
6.10.1	Detailed Description	36
6.10.2	Constructor & Destructor Documentation	36
6.10.2.1	CopyApplicationDialog	36
6.10.2.2	~CopyApplicationDialog	36
6.10.3	Member Function Documentation	36
6.10.3.1	changeEvent	36
6.10.3.2	getApplicationName	36
6.10.3.3	getSelectedGroup	37
6.10.3.4	isOk	37
6.11	CopyCategoryDialog Class Reference	37
6.11.1	Detailed Description	37
6.11.2	Constructor & Destructor Documentation	37
6.11.2.1	CopyCategoryDialog	37
6.11.2.2	~CopyCategoryDialog	38
6.11.3	Member Function Documentation	38
6.11.3.1	changeEvent	38
6.11.3.2	getApplicationName	38
6.11.3.3	getCategoryName	38
6.11.3.4	getGroupName	38
6.12	CopyFunctionDialog Class Reference	39
6.12.1	Detailed Description	39
6.12.2	Constructor & Destructor Documentation	39
6.12.2.1	CopyFunctionDialog	39
6.12.2.2	~CopyFunctionDialog	39
6.12.3	Member Function Documentation	39
6.12.3.1	changeEvent	39

6.12.3.2	getApplicationName	40
6.12.3.3	getCategoryName	40
6.12.3.4	getFunctionName	40
6.12.3.5	getGroupName	40
6.13	EditApplicationDialog Class Reference	40
6.13.1	Detailed Description	41
6.13.2	Constructor & Destructor Documentation	41
6.13.2.1	EditApplicationDialog	41
6.13.2.2	~EditApplicationDialog	42
6.13.3	Member Function Documentation	42
6.13.3.1	changeEvent	42
6.13.3.2	getAction	42
6.13.3.3	getCPI	42
6.13.3.4	getCPI_high	42
6.13.3.5	getCPI_low	42
6.13.3.6	getFileName	43
6.13.3.7	getJoystickMode	43
6.13.3.8	getName	43
6.13.3.9	getNewProfileNumber	43
6.13.3.10	getProfileForReplaced	43
6.13.3.11	getReplacedApplication	44
6.13.3.12	getWindowTitlePart	44
6.14	EditCategoryDialog Class Reference	44
6.14.1	Detailed Description	45
6.14.2	Constructor & Destructor Documentation	45
6.14.2.1	EditCategoryDialog	45
6.14.2.2	~EditCategoryDialog	45
6.14.3	Member Function Documentation	45
6.14.3.1	changeEvent	45
6.14.3.2	getAction	45
6.14.3.3	getCategoryName	45
6.14.3.4	getCopiedApplicationName	46
6.14.3.5	getCopiedCategoryName	46
6.14.3.6	getCopiedGroupName	46
6.15	EditFunctionDialog Class Reference	46
6.15.1	Detailed Description	47

6.15.2	Constructor & Destructor Documentation	47
6.15.2.1	EditFunctionDialog	47
6.15.2.2	~EditFunctionDialog	47
6.15.3	Member Function Documentation	47
6.15.3.1	changeEvent	47
6.15.3.2	eventFilter	48
6.15.3.3	getAction	48
6.15.3.4	getCopiedApplicationName	48
6.15.3.5	getCopiedCategoryName	48
6.15.3.6	getCopiedFunctionName	48
6.15.3.7	getCopiedGroupName	49
6.15.3.8	getFunctionName	49
6.15.3.9	getKeypresses	49
6.15.3.10	getType	49
6.15.3.11	keyPressEvent	49
6.15.3.12	keyReleaseEvent	49
6.15.3.13	setKeyPresses	50
6.16	FunctionDelegate Class Reference	50
6.16.1	Detailed Description	50
6.16.2	Constructor & Destructor Documentation	50
6.16.2.1	FunctionDelegate	50
6.16.3	Member Function Documentation	50
6.16.3.1	paint	50
6.17	FunctionListModel Class Reference	51
6.17.1	Detailed Description	51
6.17.2	Constructor & Destructor Documentation	51
6.17.2.1	FunctionListModel	51
6.17.3	Member Function Documentation	52
6.17.3.1	data	52
6.17.3.2	headerData	52
6.17.3.3	rowCount	52
6.18	FunctionObject Class Reference	53
6.18.1	Detailed Description	53
6.18.2	Constructor & Destructor Documentation	53
6.18.2.1	FunctionObject	53
6.18.2.2	FunctionObject	53

6.18.3	Member Function Documentation	54
6.18.3.1	getApplicationName	54
6.18.3.2	getData	54
6.18.3.3	getName	54
6.18.3.4	getType	54
6.18.3.5	isBound	54
6.18.3.6	operator<	54
6.18.3.7	operator=	55
6.18.3.8	operator==	55
6.18.3.9	operator>	55
6.18.3.10	setApplicationName	55
6.18.3.11	setBound	55
6.18.3.12	setData	56
6.18.3.13	setName	56
6.18.3.14	setType	56
6.19	GroupObject Class Reference	56
6.19.1	Detailed Description	57
6.19.2	Constructor & Destructor Documentation	57
6.19.2.1	GroupObject	57
6.19.2.2	~GroupObject	57
6.19.2.3	GroupObject	57
6.19.3	Member Function Documentation	57
6.19.3.1	getApplicationList	57
6.19.3.2	getName	58
6.19.3.3	operator<	58
6.19.3.4	operator=	58
6.19.3.5	operator==	58
6.19.3.6	setApplicationListByNode	58
6.19.3.7	setFunctionBound	59
6.19.3.8	setName	59
6.20	GroupOrAppWidget Class Reference	59
6.20.1	Detailed Description	60
6.20.2	Constructor & Destructor Documentation	60
6.20.2.1	GroupOrAppWidget	60
6.20.3	Member Function Documentation	60
6.20.3.1	setHaveNavigationTab	60

6.20.3.2	setNumButtons	60
6.21	HardwareThread Class Reference	61
6.21.1	Detailed Description	61
6.21.2	Constructor & Destructor Documentation	61
6.21.2.1	HardwareThread	61
6.21.3	Member Function Documentation	61
6.21.3.1	chooseTask	61
6.21.3.2	getLastDifferenceResult	61
6.21.3.3	run	62
6.22	IdentifiedPushButton Class Reference	62
6.22.1	Detailed Description	62
6.22.2	Constructor & Destructor Documentation	62
6.22.2.1	IdentifiedPushButton	62
6.22.3	Member Function Documentation	62
6.22.3.1	setColor	62
6.23	KeyObject Class Reference	63
6.23.1	Detailed Description	63
6.23.2	Constructor & Destructor Documentation	63
6.23.2.1	KeyObject	63
6.23.2.2	~KeyObject	64
6.23.2.3	KeyObject	64
6.23.3	Member Function Documentation	64
6.23.3.1	getDelayMs1	64
6.23.3.2	getDelayMs2	64
6.23.3.3	getKey	64
6.23.3.4	getModifiers	64
6.23.3.5	operator=	65
6.23.3.6	setDelayMs1	65
6.23.3.7	setDelayMs2	65
6.23.3.8	setKey	65
6.23.3.9	setModifiers	65
6.24	KeypressParser Class Reference	66
6.24.1	Detailed Description	66
6.24.2	Member Function Documentation	66
6.24.2.1	getMacroGUIString	66
6.24.2.2	getMacroInternalString	66

6.24.2.3	instance	67
6.24.2.4	keypressStringToHID	67
6.24.2.5	keypressStringToQt	67
6.24.2.6	macroStringToHID	68
6.24.2.7	macroStringToQt	68
6.25	MacroFunctionsDialog Class Reference	68
6.25.1	Constructor & Destructor Documentation	69
6.25.1.1	MacroFunctionsDialog	69
6.25.1.2	~MacroFunctionsDialog	69
6.25.2	Member Function Documentation	69
6.25.2.1	changeEvent	69
6.26	MainWindow Class Reference	69
6.26.1	Detailed Description	70
6.26.2	Constructor & Destructor Documentation	71
6.26.2.1	MainWindow	71
6.26.2.2	~MainWindow	71
6.26.3	Member Function Documentation	71
6.26.3.1	activateProfile	71
6.26.3.2	changeView	71
6.26.3.3	closeOtherView	71
6.26.3.4	flagsChanged	71
6.26.3.5	getActiveApplicationIndex	72
6.26.3.6	getActiveApplicationName	72
6.26.3.7	getActiveGroupIndex	72
6.26.3.8	getFlags	72
6.26.3.9	getGroupList	72
6.26.3.10	getHeaderSvg	72
6.26.3.11	initialized	72
6.26.3.12	isInitializedOK	73
6.26.3.13	mapClosed	73
6.26.3.14	paintEvent	73
6.26.3.15	processExternalMessage	73
6.26.3.16	profileEvent	73
6.26.3.17	quitModeware	73
6.26.3.18	refreshSimpleView	73
6.26.3.19	setCategoryIndex	74

6.26.3.20	setCategoryListFocus	74
6.26.3.21	setCheckBoxes	74
6.26.3.22	setFunctionIndex	74
6.26.3.23	setFunctionListFocus	74
6.26.3.24	setInactive	74
6.26.3.25	showThisView	75
6.26.3.26	stopAssigning	75
6.26.3.27	updateHeader	75
6.26.3.28	updateStatistics	75
6.27	MapWidget Class Reference	75
6.27.1	Detailed Description	76
6.27.2	Constructor & Destructor Documentation	76
6.27.2.1	MapWidget	76
6.27.3	Member Function Documentation	76
6.27.3.1	closing	76
6.28	mouseButtonAction Struct Reference	76
6.29	MouseButtonObject Class Reference	76
6.29.1	Detailed Description	77
6.29.2	Constructor & Destructor Documentation	77
6.29.2.1	MouseButtonObject	77
6.29.2.2	MouseButtonObject	77
6.29.3	Member Function Documentation	77
6.29.3.1	bindButton	77
6.29.3.2	getButtonBindings	78
6.29.3.3	getFunctionWithFlags	78
6.29.3.4	getName	78
6.29.3.5	isBoundInApplication	78
6.29.3.6	isBoundToFunction	78
6.29.3.7	isBoundToFunctionWithFlags	79
6.29.3.8	isBoundToThis	79
6.29.3.9	isOnlyPartiallyFreeInApplication	79
6.29.3.10	operator=	80
6.29.3.11	unbindAll	80
6.29.3.12	unbindButton	80
6.30	mouseButtonRepresentation Struct Reference	80
6.31	mouseButtonWithFlags Struct Reference	81

6.32 MouseHardware Class Reference	81
6.32.1 Detailed Description	82
6.32.2 Member Function Documentation	82
6.32.2.1 calculateChecksum	82
6.32.2.2 checkLibrary	82
6.32.2.3 checkMouseState	82
6.32.2.4 closeLibrary	83
6.32.2.5 enableFirmwareUpdate	83
6.32.2.6 eraseBlock	83
6.32.2.7 eraseFlash	83
6.32.2.8 getActiveProfile	83
6.32.2.9 getBlockChecksum	84
6.32.2.10 getFreeBlocks	84
6.32.2.11 getJoyCoords	84
6.32.2.12 getProfileCopy	84
6.32.2.13 getRevision	85
6.32.2.14 instance	85
6.32.2.15 isInitializedOK	85
6.32.2.16 pollProfileState	85
6.32.2.17 quit	86
6.32.2.18 readMouse	86
6.32.2.19 readProfileInfoBlock	86
6.32.2.20 setProfile	86
6.32.2.21 setProfileCopy	86
6.32.2.22 updateProfileCopy	87
6.32.2.23 writeMouse	87
6.32.2.24 writeProfileInfoBlock	87
6.32.2.25 writeProfileKeyMappingBlock	87
6.33 MouseObject Class Reference	88
6.33.1 Detailed Description	89
6.33.2 Member Function Documentation	89
6.33.2.1 buttonActivated	89
6.33.2.2 compareProfileInfoBlocks	89
6.33.2.3 configureButtons	90
6.33.2.4 createInfoBlockContents	90
6.33.2.5 createMappingBlockContents	90

6.33.2.6	enableFirmwareUpdate	90
6.33.2.7	eraseAll	91
6.33.2.8	flagsChanged	91
6.33.2.9	getActiveApplicationName	91
6.33.2.10	getActiveButton	91
6.33.2.11	getActiveProfileNumber	91
6.33.2.12	getAndResetStatistics	92
6.33.2.13	getButtonEnum	92
6.33.2.14	getButtonString	92
6.33.2.15	getFirmwareRevision	92
6.33.2.16	getFlags	93
6.33.2.17	getFreeButtonDisplay	93
6.33.2.18	getFreeButtons	93
6.33.2.19	getFunctionBindings	93
6.33.2.20	getJoyCoords	94
6.33.2.21	getMappingBlockDifferences	94
6.33.2.22	getMouseButtonPointer	94
6.33.2.23	getMouseState	94
6.33.2.24	getPartiallyFreeButtons	95
6.33.2.25	initHardware	95
6.33.2.26	instance	95
6.33.2.27	isMouseConnected	95
6.33.2.28	isProfileSynced	95
6.33.2.29	profileEvent	96
6.33.2.30	quit	96
6.33.2.31	refreshWidget	96
6.33.2.32	refreshWidgetSignal	96
6.33.2.33	setActiveButton	96
6.33.2.34	setActiveProfile	96
6.33.2.35	setButtonToFunction	97
6.33.2.36	setFlags	97
6.33.2.37	setFreeButtonDisplay	97
6.33.2.38	setMainWindowPtr	97
6.33.2.39	setMouseState	97
6.33.2.40	setMouseWidget	98
6.33.2.41	setProfileSynced	98

6.33.2.42	statisticsEvent	98
6.33.2.43	unbindAll	98
6.34	MouseWidget Class Reference	98
6.34.1	Detailed Description	99
6.34.2	Constructor & Destructor Documentation	99
6.34.2.1	MouseWidget	99
6.34.3	Member Function Documentation	99
6.34.3.1	getFreeButtonDisplay	99
6.34.3.2	mousePressEvent	99
6.34.3.3	paintEvent	99
6.34.3.4	refresh	100
6.34.3.5	setFreeButtonDisplay	100
6.35	MyAllStatisticsTable Class Reference	100
6.35.1	Detailed Description	100
6.35.2	Constructor & Destructor Documentation	100
6.35.2.1	MyAllStatisticsTable	100
6.35.3	Member Function Documentation	101
6.35.3.1	contextMenuEvent	101
6.35.3.2	paintEvent	101
6.35.3.3	recalculateColumnSizes	101
6.35.3.4	setShortcuts	101
6.36	MyKey Struct Reference	101
6.37	myKey Struct Reference	102
6.38	MyKeyPressHID Struct Reference	102
6.39	MyListView Class Reference	102
6.39.1	Detailed Description	102
6.39.2	Constructor & Destructor Documentation	102
6.39.2.1	MyListView	102
6.40	MyStatisticsTable Class Reference	103
6.40.1	Detailed Description	103
6.40.2	Constructor & Destructor Documentation	103
6.40.2.1	MyStatisticsTable	103
6.40.3	Member Function Documentation	103
6.40.3.1	contextMenuEvent	103
6.40.3.2	paintEvent	104
6.40.3.3	recalculateColumnSizes	104

6.40.3.4	setShortcuts	104
6.41	MyToolButton Class Reference	104
6.41.1	Detailed Description	104
6.41.2	Constructor & Destructor Documentation	104
6.41.2.1	MyToolButton	104
6.41.3	Member Function Documentation	105
6.41.3.1	paintEvent	105
6.41.3.2	setType	105
6.42	png_stream_to_byte_array_closure_t Struct Reference	105
6.43	PROFILE_INFO_BLOCK Struct Reference	105
6.44	SetupParser Class Reference	106
6.44.1	Detailed Description	107
6.44.2	Member Function Documentation	108
6.44.2.1	addApplication	108
6.44.2.2	addBinding	108
6.44.2.3	addBindingNoReplace	108
6.44.2.4	addCategory	109
6.44.2.5	addFunction	109
6.44.2.6	addGroup	109
6.44.2.7	addStatistics	110
6.44.2.8	deleteApplication	110
6.44.2.9	deleteCategory	110
6.44.2.10	deleteFunction	110
6.44.2.11	deleteGroup	111
6.44.2.12	getAllStatistics	111
6.44.2.13	getApplicationStatistics	112
6.44.2.14	getBindingsDomDocument	112
6.44.2.15	getGroupList	112
6.44.2.16	getLanguage	113
6.44.2.17	hasBinding	113
6.44.2.18	importApplication	113
6.44.2.19	instance	113
6.44.2.20	isFunctionBound	114
6.44.2.21	modifyApplication	114
6.44.2.22	modifyCategory	114
6.44.2.23	modifyFunction	115

6.44.2.24	modifyGroup	115
6.44.2.25	parseBindings	115
6.44.2.26	parseSetup	116
6.44.2.27	parseStatistics	116
6.44.2.28	readBindingsFile	116
6.44.2.29	readSetupFile	117
6.44.2.30	readStatisticsFile	117
6.44.2.31	removeBinding	117
6.44.2.32	saveApplication	117
6.44.2.33	setFunctionBound	118
6.44.2.34	setGroupList	118
6.44.2.35	setLanguage	118
6.44.2.36	thisFunctionExists	118
6.44.2.37	updateStatistics	119
6.45	SimpleScreen Class Reference	119
6.45.1	Detailed Description	119
6.45.2	Constructor & Destructor Documentation	120
6.45.2.1	SimpleScreen	120
6.45.2.2	~SimpleScreen	120
6.45.3	Member Function Documentation	120
6.45.3.1	activateProfileOtherView	120
6.45.3.2	changeEvent	120
6.45.3.3	changeView	120
6.45.3.4	closeOtherView	120
6.45.3.5	parseConfig	121
6.45.3.6	quitModeware	121
6.45.3.7	refreshView	121
6.45.3.8	showThisView	121
6.45.3.9	updateActiveMode	121
6.46	statistic Struct Reference	121
6.47	statisticFunctionData Struct Reference	122
6.48	StatisticObject Class Reference	122
6.48.1	Detailed Description	122
6.48.2	Constructor & Destructor Documentation	123
6.48.2.1	StatisticObject	123
6.48.3	Member Function Documentation	123

6.48.3.1	getAllClicks	123
6.48.3.2	getApplicationName	123
6.48.3.3	getFunctionClicks	123
6.48.3.4	operator<	124
6.48.3.5	operator=	124
6.48.3.6	operator==	124
6.48.3.7	operator>	124
6.48.3.8	setApplicationName	124
6.48.3.9	setFunctionClicks	125
6.48.3.10	sortFunctions	125
6.49	StatisticsAllDialog Class Reference	125
6.49.1	Detailed Description	125
6.49.2	Constructor & Destructor Documentation	126
6.49.2.1	StatisticsAllDialog	126
6.49.2.2	~StatisticsAllDialog	126
6.49.3	Member Function Documentation	126
6.49.3.1	changeEvent	126
6.50	StatisticsAllModel Class Reference	127
6.50.1	Detailed Description	127
6.50.2	Constructor & Destructor Documentation	127
6.50.2.1	StatisticsAllModel	127
6.50.3	Member Function Documentation	127
6.50.3.1	columnCount	127
6.50.3.2	data	128
6.50.3.3	headerData	128
6.50.3.4	rowCount	128
6.51	StatisticsDialog Class Reference	129
6.51.1	Detailed Description	129
6.51.2	Constructor & Destructor Documentation	129
6.51.2.1	StatisticsDialog	129
6.51.2.2	~StatisticsDialog	129
6.51.3	Member Function Documentation	130
6.51.3.1	changeEvent	130
6.52	StatisticsModel Class Reference	130
6.52.1	Detailed Description	130
6.52.2	Constructor & Destructor Documentation	130

6.52.2.1	StatisticsModel	130
6.52.3	Member Function Documentation	131
6.52.3.1	columnCount	131
6.52.3.2	data	131
6.52.3.3	headerData	131
6.52.3.4	rowCount	131
6.53	stringToKeyInfo Struct Reference	132
6.54	stringToModifierInfo Struct Reference	132
7	File Documentation	133
7.1	src/addapplicationdialog.h File Reference	133
7.1.1	Detailed Description	133
7.2	src/addfunctiondialog.h File Reference	133
7.2.1	Detailed Description	134
7.3	src/applicationlistmodel.h File Reference	134
7.3.1	Detailed Description	134
7.4	src/applicationobject.h File Reference	134
7.4.1	Detailed Description	135
7.5	src/AtUsbHid.h File Reference	135
7.5.1	Detailed Description	136
7.5.2	Define Documentation	136
7.5.2.1	chBEGINTHREADEX	136
7.6	src/autoswitchdetailsdialog.h File Reference	136
7.6.1	Detailed Description	137
7.7	src/buttonbindingobject.h File Reference	137
7.7.1	Detailed Description	137
7.8	src/categorylistmodel.h File Reference	137
7.8.1	Detailed Description	137
7.9	src/categoryobject.h File Reference	137
7.9.1	Detailed Description	138
7.10	src/copyapplicationdialog.h File Reference	138
7.10.1	Detailed Description	138
7.11	src/copycategorydialog.h File Reference	138
7.11.1	Detailed Description	139
7.12	src/copyfunctiondialog.h File Reference	139
7.12.1	Detailed Description	139
7.13	src/editapplicationdialog.h File Reference	139

7.13.1 Detailed Description	139
7.14 src/editcategorydialog.h File Reference	140
7.14.1 Detailed Description	140
7.15 src/editfunctiondialog.h File Reference	140
7.15.1 Detailed Description	140
7.16 src/functiondelegate.h File Reference	141
7.16.1 Detailed Description	141
7.17 src/functionlistmodel.h File Reference	141
7.17.1 Detailed Description	141
7.18 src/functionobject.h File Reference	141
7.18.1 Detailed Description	141
7.19 src/groupobject.h File Reference	142
7.19.1 Detailed Description	142
7.20 src/grouporappwidget.h File Reference	142
7.20.1 Detailed Description	142
7.21 src/hardwarethread.h File Reference	142
7.21.1 Detailed Description	143
7.22 src/identifiedpushbutton.h File Reference	143
7.22.1 Detailed Description	143
7.23 src/keyobject.h File Reference	143
7.23.1 Detailed Description	143
7.24 src/keypress_common.h File Reference	144
7.24.1 Detailed Description	148
7.24.2 Function Documentation	149
7.24.2.1 internalStringFromQtKey	149
7.24.2.2 internalStringFromQtModifier	149
7.24.3 Variable Documentation	149
7.24.3.1 STRING_TO_MODIFIER_HID	149
7.25 src/keypressparser.h File Reference	150
7.25.1 Detailed Description	150
7.26 src/macروفunctionsdialog.h File Reference	150
7.26.1 Detailed Description	150
7.27 src/mainwindow.h File Reference	150
7.27.1 Detailed Description	151
7.28 src/mapwidget.h File Reference	151
7.28.1 Detailed Description	151

7.29	src/mousebuttonobject.h File Reference	152
7.29.1	Detailed Description	152
7.30	src/mousehardware.h File Reference	152
7.30.1	Detailed Description	153
7.31	src/mouseobject.h File Reference	153
7.31.1	Detailed Description	154
7.31.2	Variable Documentation	154
7.31.2.1	representation	154
7.32	src/mousewidget.h File Reference	154
7.32.1	Detailed Description	155
7.33	src/myallstatisticstable.h File Reference	155
7.33.1	Detailed Description	155
7.34	src/mydialogs_common.h File Reference	155
7.34.1	Detailed Description	155
7.34.2	Function Documentation	156
7.34.2.1	caseInsensitiveLessThan	156
7.34.2.2	getFreeProfileNumber	156
7.35	src/mylistview.h File Reference	156
7.35.1	Detailed Description	156
7.36	src/mystaticstable.h File Reference	156
7.36.1	Detailed Description	157
7.37	src/mytoolbarbutton.h File Reference	157
7.37.1	Detailed Description	157
7.38	src/setup_common.h File Reference	157
7.38.1	Detailed Description	164
7.38.2	Enumeration Type Documentation	164
7.38.2.1	functionType	164
7.38.3	Function Documentation	165
7.38.3.1	caseInsensitiveStatisticFunctionDataLessThan	165
7.38.3.2	deleteChildrenByAttribute	165
7.38.3.3	getChildByName	165
7.38.3.4	getGrandChildByName	166
7.38.3.5	giveParseError	166
7.38.3.6	hasChildrenWithAttribute	166
7.38.4	Variable Documentation	167
7.38.4.1	joystickModeTexts	167

7.38.4.2	mouseButtonActions	167
7.38.4.3	pdf_rectangles	167
7.38.4.4	pdf_rectangles_landscape	168
7.39	src/setup_objects_common.h File Reference	169
7.39.1	Detailed Description	169
7.39.2	Function Documentation	169
7.39.2.1	findApplicationByProfileNumber	169
7.40	src/setupparser.h File Reference	169
7.40.1	Detailed Description	170
7.41	src/simplescreen.h File Reference	170
7.41.1	Detailed Description	170
7.42	src/statisticobject.h File Reference	170
7.42.1	Detailed Description	171
7.43	src/statisticsalldialog.h File Reference	171
7.43.1	Detailed Description	171
7.44	src/statisticsallmodel.h File Reference	171
7.44.1	Detailed Description	171
7.45	src/statisticsdialog.h File Reference	172
7.45.1	Detailed Description	172
7.46	src/statisticsmodel.h File Reference	172
7.46.1	Detailed Description	172

Chapter 1

Main Page

1.1 Overview

This program controls and configures the Warmouse Meta computer mouse. It reads its data from xml-formatted setup files. User will then connect mouse buttons with different functions in applications, and the software will reconfigure the mouse based on the xml contents.

1.2 Some differences in terminology

What is called "mode" in the software is called "application" in the source code. The number of the mode is called "profile number".

1.3 Source code comments

All methods are fully commented in the header files, except some slots and signals created by Qt Creator, that are self-explanatory based on their names. Currently, the .cpp files are poorly commented. Check back at warmouse.com for updates. More comments will be added.

1.4 Software architecture

[MainWindow](#) is responsible for all communications with the mouse, even when it is hidden. When [SimpleScreen](#) is being used, it emits signals to [MainWindow](#), and [MainWindow](#) emits the mouse events back to [SimpleScreen](#) as signals. [MouseObject](#) abstracts the hardware implementation and the mouse graphic. For Linux and Macintosh, [MouseHardware](#) could later be changed with a completely different object that supports communications in those architectures, and it would still be used the same way by [MainWindow](#). [MouseObject](#) would be responsible for calling the proper methods, based on the current architecture.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Ui	9
--------------------------	---

Chapter 3

Class Index

3.1 Class List

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

AddApplicationDialog	11
AddFunctionDialog	14
ApplicationListModel	17
ApplicationObject	18
AutoswitchDetailsDialog	27
ButtonBindingObject	29
CategoryListModel	31
CategoryObject	32
charToKeyInfo	35
CopyApplicationDialog	35
CopyCategoryDialog	37
CopyFunctionDialog	39
EditApplicationDialog	40
EditCategoryDialog	44
EditFunctionDialog	46
FunctionDelegate	50
FunctionListModel	51
FunctionObject	53
GroupObject	56
GroupOrAppWidget	59
HardwareThread	61
IdentifiedPushButton	62
KeyObject	63
KeypressParser	66
MacroFunctionsDialog	68
MainWindow	69
MapWidget	75
mouseButtonAction	76
MouseButtonObject	76
mouseButtonRepresentation	80
mouseButtonWithFlags	81
MouseHardware	81
MouseObject	88

MouseWidget	98
MyAllStatisticsTable	100
MyKey	101
myKey	102
MyKeyPressHID	102
MyListView	102
MyStatisticsTable	103
MyToolButton	104
png_stream_to_byte_array_closure_t	105
PROFILE_INFO_BLOCK	105
SetupParser	106
SimpleScreen	119
statistic	121
statisticFunctionData	122
StatisticObject	122
StatisticsAllDialog	125
StatisticsAllModel	127
StatisticsDialog	129
StatisticsModel	130
stringToKeyInfo	132
stringToModifierInfo	132

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

src/addapplicationdialog.h (Defines the AddApplicationDialog class, which creates a dialog for adding an application)	133
src/addfunctiondialog.h (Defines the AddFunctionDialog class, which creates a dialog for adding a function)	133
src/applicationlistmodel.h (Defines the ApplicationListModel class, subclassed from QAbstractListModel)	134
src/applicationobject.h (Defines the ApplicationObject class)	134
src/AtUsbHid.h (Header file for using AtUsbHid DLL, from Atmel)	135
src/autoswitchdetailsdialog.h (Defines the AutoswitchDetailsDialog class, which creates a dialog for changing autoswitch details of an application)	136
src/buttonbindingobject.h (Defines the ButtonBindingObject class)	137
src/categorylistmodel.h (Defines the CategoryListModel class, subclassed from QAbstractListModel)	137
src/categoryobject.h (Defines the CategoryObject class)	137
src/copyapplicationdialog.h (Defines the CopyApplicationDialog class, which creates a dialog for copying an application)	138
src/copycategorydialog.h (Defines the CopyCategoryDialog class, which creates a dialog for copying a category of functions)	138
src/copyfunctiondialog.h (Defines the CopyFunctionDialog class, which creates a dialog for copying a functions)	139
src/editapplicationdialog.h (Defines the EditApplicationDialog class, which creates a dialog for editing an application)	139
src/editcategorydialog.h (Defines the EditCategoryDialog class, which creates a dialog for editing a category of functions)	140
src/editfunctiondialog.h (Defines the EditFunctionDialog class, which creates a dialog for editing a function)	140
src/functiondelegate.h (Defines the FunctionDelegate class, subclassed from QItemDelegate) . .	141
src/functionlistmodel.h (Defines the FunctionListModel class, subclassed from QAbstractListModel)	141
src/functionobject.h (Defines the FunctionObject class)	141
src/groupobject.h (Defines the GroupObject class)	142
src/grouporappwidget.h (Defines the GroupOrAppWidget class, which creates a header of buttons for choosing a group or application)	142

src/hardwarethread.h (Defines the HardwareThread class)	142
src/identifiedpushbutton.h (Defines the IdentifiedPushButton class, which creates a button widget for choosing a group or application)	143
src/keyobject.h (Defines the KeyObject class, for modeling a keypress, with its modifiers and delays)	143
src/keypress_common.h (Structures, constants and functions which are used by several classes, and which are related to keypresses)	144
src/keypressparser.h (Parser class for the keypress sequence format used by the xml-formatted setup file)	150
src/macrofunctionsdialog.h (Defines the MacroFunctionsDialog class, which creates a dialog for displaying the supported special keys and functions in macros)	150
src/mainwindow.h (Defines the MainWindow class, which creates the advanced view)	150
src/mapwidget.h (Defines the MapWidget class, which creates a widget for showing mode map)	151
src/mousebuttonobject.h (Defines the MouseButtonObject class)	152
src/mousehardware.h (Defines the MouseHardware class)	152
src/mouseobject.h (Defines the MouseObject class)	153
src/mousewidget.h (Defines the MouseWidget class, which creates a widget for showing the mouse graphic in advanced view)	154
src/myallstatisticstable.h (Defines the MyAllStatisticsTable class, which creates a table view for displaying statistics for all modes)	155
src/mydialogs_common.h (Structures, constants and functions which are used by several dialog classes)	155
src/mylistview.h (Defines the MyListView class, which has some custom features that default list views don't support)	156
src/mystatisticstable.h (Defines the MyStatisticsTable class, which creates a table view for displaying statistics for a specific mode)	156
src/mytoolbutton.h (Defines the MyToolButton class, which is a custom tool button for navigation tab)	157
src/setup_common.h (Structures, constants and functions which are used by several classes, and that are related to the setup file)	157
src/setup_objects_common.h (Functions which are used by several classes, and that are related to the setup file and that require including the group/application object headers)	169
src/setupparser.h (Parser class for the xml-formatted setup file)	169
src/simplescreen.h (Defines the SimpleScreen class, which creates the basic view, called simple screen in the code)	170
src/statisticobject.h (Defines the StatisticObject class)	170
src/statisticsalldialog.h (Defines the StatisticsAllDialog class, which creates a dialog for displaying click statistics for all modes)	171
src/statisticsallmodel.h (Defines the StatisticsAllModel class, subclassed from QAbstractTableModel)	171
src/statisticsdialog.h (Defines the StatisticsDialog class, which creates a dialog for displaying click statistics for a specific mode)	172
src/statisticsmodel.h (Defines the StatisticsModel class, subclassed from QAbstractTableModel)	172

Chapter 5

Namespace Documentation

5.1 Ui Namespace Reference

5.1.1 Detailed Description

A class which models dialog for displaying the supported special keys and functions in macros

Chapter 6

Class Documentation

6.1 AddApplicationDialog Class Reference

```
#include <addapplicationdialog.h>
```

Public Member Functions

- [AddApplicationDialog](#) (const QList< [GroupObject](#) > &groupList, QWidget *parent=0)
- virtual [~AddApplicationDialog](#) ()
- bool [getReplacedApplication](#) (QString &applicationToReplace) const
- unsigned int [getProfileForReplaced](#) (void) const
- QString [getName](#) (void) const
- unsigned int [getNewProfileNumber](#) (void) const
- QString [getFileName](#) (void) const
- QString [getWindowTitlePart](#) (void) const
- quint8 [getCPI](#) (void) const
- quint8 [getCPI_low](#) (void) const
- quint8 [getCPI_high](#) (void) const
- quint8 [getJoystickMode](#) (void) const
- bool [isOk](#) () const

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.1.1 Detailed Description

A class which models add application dialog.

6.1.2 Constructor & Destructor Documentation

6.1.2.1 AddApplicationDialog::AddApplicationDialog (const QList< [GroupObject](#) > & *groupList*, QWidget * *parent* = 0) [explicit]

Constructor

Parameters

groupList Reference to MainWindow's group list (required)

parent Parent of this QObject (optional)

6.1.2.2 AddApplicationDialog::~AddApplicationDialog () [virtual]

Destructor

6.1.3 Member Function Documentation**6.1.3.1 void AddApplicationDialog::changeEvent (QEvent * e) [protected]**

Reimplementation of QObject's changeEvent

Parameters

e Event given by Qt

6.1.3.2 quint8 AddApplicationDialog::getCPI (void) const

Get application's default CPI, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

CPI in internal format

6.1.3.3 quint8 AddApplicationDialog::getCPI_high (void) const

Get application's high CPI, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

High CPI in internal format

6.1.3.4 quint8 AddApplicationDialog::getCPI_low (void) const

Get application's low CPI, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

Low CPI in internal format

6.1.3.5 QString AddApplicationDialog::getFileName (void) const

Get file name associated with added application.

Returns empty string, if no file was chosen.

Returns

File name, or empty if none

6.1.3.6 quint8 AddApplicationDialog::getJoystickMode (void) const

Get application's joystick mode

Mode numbers are defined in [setup_common.h](#)

Returns

Joystick mode number

6.1.3.7 QString AddApplicationDialog::getName (void) const

Get name of added application.

Returns

Name of added application

6.1.3.8 unsigned int AddApplicationDialog::getNewProfileNumber (void) const

Get profile number of added application.

Returns

Profile number of added application

6.1.3.9 unsigned int AddApplicationDialog::getProfileForReplaced (void) const

Get new profile number that was given to replaced application.

Check first with `getReplacedApplication` that something was replaced, because return value 0 is ambiguous.

Returns

New profile number of replaced application

6.1.3.10 `bool AddApplicationDialog::getReplacedApplication (QString & applicationToReplace) const`

Get name of application replaced by new application in this profile number.

Parameters

applicationToReplace Reference to where the name of replaced app is placed

Returns

true if application was replaced, or false if no application was replaced

6.1.3.11 `QString AddApplicationDialog::getWindowTitlePart (void) const`

Get the part of window title associated with added application.

For example, openoffice.org adds the open file name to the window title. Only the static part of the title is given to this dialog, and it is matched as substring of the full title. Returns empty string, if no title was chosen.

Returns

Title part, or empty if none

6.1.3.12 `bool AddApplicationDialog::isOk () const`

Did user click OK?

Returns

true if user clicked OK, or false if dialog was canceled

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

- [src/addapplicationdialog.h](#)
- [src/addapplicationdialog.cpp](#)

6.2 AddFunctionDialog Class Reference

```
#include <addfunctiondialog.h>
```

Public Member Functions

- [AddFunctionDialog](#) (const QString applicationName, QWidget *parent=0)
- virtual [~AddFunctionDialog](#) ()
- [QString getFunctionName](#) () const
- [QString getKeypresses](#) () const
- [functionType getType](#) (void)
- [bool isOk](#) () const

Protected Member Functions

- bool [eventFilter](#) (QObject *target, QEvent *event)
- virtual void [changeEvent](#) (QEvent *e)
- void [keyPressEvent](#) (QKeyEvent *event)
- void [keyReleaseEvent](#) (QKeyEvent *event)

6.2.1 Detailed Description

A class which models add function dialog.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 AddFunctionDialog::AddFunctionDialog (const QString *applicationName*, QWidget **parent* = 0) [explicit]

Constructor

Parameters

- applicationName* Name of application where we are adding function (required)
parent Parent of this QObject (optional)

6.2.2.2 AddFunctionDialog::~AddFunctionDialog () [virtual]

Destructor

6.2.3 Member Function Documentation

6.2.3.1 void AddFunctionDialog::changeEvent (QEvent * *e*) [protected, virtual]

Reimplementation of QObject's changeEvent

Parameters

- e* Event given by Qt

6.2.3.2 bool AddFunctionDialog::eventFilter (QObject * *target*, QEvent * *event*) [protected]

Reimplementation of QObject's eventFilter

Parameters

- target* QObject that the event was directed to
event The event that occurred

Returns

- true if we stop event from being handled further, or false if some other target may handle the event

6.2.3.3 QString AddFunctionDialog::getFunctionName () const

Get name of added function.

Returns

Name of added function

6.2.3.4 QString AddFunctionDialog::getKeypresses () const

Get keypresses of added function in the internal representation of the xml file.

Assumes that string contents have been placed to keypresses_ using macroToData().

Returns

Keypresses

6.2.3.5 functionType AddFunctionDialog::getType (void)

Get function type.

Returns

Function type

6.2.3.6 bool AddFunctionDialog::isOk () const

Did user click OK?

Returns

true if user clicked OK, or false if dialog was canceled

6.2.3.7 void AddFunctionDialog::keyPressEvent (QKeyEvent * event) [protected]

Handler for keypress events

Parameters

event Event given by Qt

6.2.3.8 void AddFunctionDialog::keyReleaseEvent (QKeyEvent * event) [protected]

Handler for key release events

Parameters

event Event given by Qt

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

- [src/addfunctiondialog.h](#)
- [src/addfunctiondialog.cpp](#)

6.3 ApplicationListModel Class Reference

```
#include <applicationlistmodel.h>
```

Public Member Functions

- [ApplicationListModel](#) (const QList< [GroupObject](#) > &groupList, QObject *parent=0)
- virtual [~ApplicationListModel](#) ()
- int [rowCount](#) (const QModelIndex &parent=QModelIndex()) const
Implementation of the virtual function rowCount(...) in QAbstractListModel.
- QVariant [data](#) (const QModelIndex &index, int role) const
Implementation of the virtual function data(...) in QAbstractListModel.
- QVariant [headerData](#) (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const
Implementation of the virtual function headerData(...) in QAbstractListModel.

6.3.1 Detailed Description

A class which models a list of applications

6.3.2 Constructor & Destructor Documentation

6.3.2.1 ApplicationListModel::ApplicationListModel (const QList< GroupObject > &groupList, QObject * parent = 0) [explicit]

Constructor

Parameters

- groupList* Reference to the group list containing the applications we are modeling (required)
- parent* Parent of this QObject (optional)

6.3.2.2 ApplicationListModel::~~ApplicationListModel () [virtual]

Destructor

6.3.3 Member Function Documentation

6.3.3.1 QVariant ApplicationListModel::data (const QModelIndex & index, int role) const

Implementation of the virtual function data(...) in QAbstractListModel.

Returns the data of the desired application

Parameters

- index* Reference to the index, from which we get the application data

role See Qt documentation of views

Returns

The data of the application

6.3.3.2 QVariant ApplicationListModel::headerData (int section, Qt::Orientation orientation, int role = Qt::DisplayRole) const

Implementation of the virtual function headerData(...) in QAbstractListModel.

Gives the headers for those views that use them

Parameters

section See Qt documentation of views

orientation See Qt documentation of views

role See Qt documentation of views

Returns

The data for the headers

6.3.3.3 int ApplicationListModel::rowCount (const QModelIndex & parent = QModelIndex ()) const

Implementation of the virtual function rowCount(...) in QAbstractListModel.

Returns the number of applications in the list

Parameters

parent Parent, for which we return row count (optional)

Returns

The number of applications

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

- [src/applicationlistmodel.h](#)
- [src/applicationlistmodel.cpp](#)

6.4 ApplicationObject Class Reference

```
#include <applicationobject.h>
```

Public Member Functions

- [ApplicationObject](#) (QString name)
- [ApplicationObject](#) (const [ApplicationObject](#) &source)
- [~ApplicationObject](#) ()
- QString [getName](#) (void) const
- bool [setName](#) (const QString name)
- unsigned int [getProfileNumber](#) (void) const
- void [setProfileNumber](#) (const unsigned int profileNumber)
- QString [getFileName](#) (void) const
- bool [setFileName](#) (const QString fileName)
- QString [getWindowTitlePart](#) (void) const
- bool [setWindowTitlePart](#) (const QString windowTilePart)
- bool [getAutoswitchDisabled](#) (void) const
- void [setAutoswitchDisabled](#) (const bool autoswitchDisabled)
- quint8 [getCPI](#) (void) const
- bool [setCPI](#) (quint8 CPI)
- quint8 [getCPI_low](#) (void) const
- bool [setCPI_low](#) (quint8 CPI_low)
- quint8 [getCPI_high](#) (void) const
- bool [setCPI_high](#) (quint8 CPI_high)
- quint8 [getDoubleClickSpeed](#) (void) const
- bool [setDoubleClickSpeed](#) (quint8 newSpeed)
- quint8 [getScrollWheelLines](#) (void) const
- bool [setScrollWheelLines](#) (quint8 newLines)
- quint8 [getSensitivity](#) (void) const
- bool [setSensitivity](#) (quint8 newSensitivity)
- quint8 [getJoystickMode](#) (void) const
- bool [setJoystickMode](#) (quint8 joystickMode)
- QString [getGroupName](#) (void) const
- bool [setGroupName](#) (const QString &groupName)
- QList< [CategoryObject](#) > [getCategoryList](#) (void) const
- bool [setCategoryListByNode](#) (QDomNode applicationNode)
- bool [setFunctionListByNode](#) (QDomNode setupApplicationNode, QDomNode bindingsApplicationNode)
- bool [setFunctionBound](#) (const QString functionName, bool bound)
- bool [operator<](#) (const [ApplicationObject](#) &target) const
- [ApplicationObject](#) & [operator=](#) (const [ApplicationObject](#) &right)
- bool [operator==](#) (const [ApplicationObject](#) &target) const

6.4.1 Detailed Description

A class which models an application

6.4.2 Constructor & Destructor Documentation

6.4.2.1 `ApplicationObject::ApplicationObject (QString name)`

Constructor

Parameters

name Application's name (required)

6.4.2.2 `ApplicationObject::ApplicationObject (const ApplicationObject & source)`

Copy constructor, performs deep copy

Parameters

source [ApplicationObject](#) to be copied

6.4.2.3 `ApplicationObject::~~ApplicationObject ()`

Destructor

6.4.3 Member Function Documentation

6.4.3.1 `bool ApplicationObject::getAutoswitchDisabled (void) const`

Getter for autoswitch disabled

Returns

true if autoswitch disabled, or false if enabled

6.4.3.2 `QList< CategoryObject > ApplicationObject::getCategoryList (void) const`

Getter for list of categories in this application

Returns

category list

6.4.3.3 `quint8 ApplicationObject::getCPI (void) const`

Getter for CPI, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

Application's CPI

6.4.3.4 quint8 ApplicationObject::getCPI_high (void) const

Getter for CPI_high, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

Application's CPI_high

6.4.3.5 quint8 ApplicationObject::getCPI_low (void) const

Getter for CPI_low, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

Application's CPI_low

6.4.3.6 quint8 ApplicationObject::getDoubleClickSpeed (void) const

Getter for double click speed

Returns

Application's double click speed

6.4.3.7 QString ApplicationObject::getFileName (void) const

Getter for fileName

Returns

Application's fileName, or empty string if has none

6.4.3.8 QString ApplicationObject::getGroupName (void) const

Getter for group name

Returns

Application's parent group's name

6.4.3.9 quint8 ApplicationObject::getJoystickMode (void) const

Getter for joystick mode

Returns

Application's joystick mode

6.4.3.10 QString ApplicationObject::getName (void) const

Getter for name

Returns

Application's name

6.4.3.11 unsigned int ApplicationObject::getProfileNumber (void) const

Getter for profile number

Returns

Profile number, or 0 if has none

6.4.3.12 quint8 ApplicationObject::getScrollWheelLines (void) const

Getter for scroll wheel lines per one click forward/backward

Returns

Application's scroll wheel lines per click

6.4.3.13 quint8 ApplicationObject::getSensitivity (void) const

Getter for mouse sensitivity

Returns

Application's mouse sensitivity

6.4.3.14 QString ApplicationObject::getWindowTitlePart (void) const

Getter for window title part

Returns

Application's window title part, or empty string if has none

6.4.3.15 bool ApplicationObject::operator< (const ApplicationObject & *target*) const

Operator <, compares application's name

Parameters

target Application we compare to

Returns

as (application.getName() < target.getName())

6.4.3.16 ApplicationObject & ApplicationObject::operator= (const ApplicationObject & *right*)

Operator =, performs deep copy

Parameters

right Right side of assignment

Returns

[ApplicationObject](#) reference

6.4.3.17 bool ApplicationObject::operator==(const ApplicationObject & *target*) const

Operator ==

Parameters

target Application we compare to

Returns

true if the names are the same, or false otherwise

6.4.3.18 void ApplicationObject::setAutoswitchDisabled (const bool *autoswitchDisabled*)

Setter for autoswitch disabled

This function cannot fail, hence no return value

Parameters

autoswitchDisabled new value for autoswitch disabled

6.4.3.19 bool ApplicationObject::setCategoryListByNode (QDomNode *applicationNode*)

Receives a Dom node, and sets the categoryList_ according to it

Parameters

applicationNode Dom node to this application

Returns

true if set succeeded, or false if there was an error

6.4.3.20 bool ApplicationObject::setCPI (quint8 *CPI*)

Setter for CPI, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Parameters

CPI new CPI

Returns

true if CPI was set, or false if there was an error

6.4.3.21 bool ApplicationObject::setCPI_high (quint8 *CPI_high*)

Setter for *CPI_high*, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Parameters

CPI_high new *CPI_high*

Returns

true if *CPI_high* was set, or false if there was an error

6.4.3.22 bool ApplicationObject::setCPI_low (quint8 *CPI_low*)

Setter for *CPI_low*, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Parameters

CPI_low new *CPI_low*

Returns

true if *CPI_low* was set, or false if there was an error

6.4.3.23 bool ApplicationObject::setDoubleClickSpeed (quint8 *newSpeed*)

Setter for double click speed

Parameters

newSpeed new double click speed

Returns

true if speed was set, or false if there was an error

6.4.3.24 bool ApplicationObject::setFileName (const QString *fileName*)

Setter for *fileName*

Parameters

fileName new fileName

Returns

true if fileName was set, or false if there was an error

6.4.3.25 bool ApplicationObject::setFunctionBound (const QString *functionName*, bool *bound*)

Sets a particular function as bound or unbound

Parameters

functionName Name of the function

bound Do we set bound or unbound?

Returns

true if set succeeded, or false if there was an error

6.4.3.26 bool ApplicationObject::setFunctionListByNode (QDomNode *setupApplicationNode*, QDomNode *bindingsApplicationNode*)

Receives a Dom node, and sets the functionList_ according to it

Parameters

setupApplicationNode Dom node to this application in setup xml file

bindingsApplicationNode Dom node to this application in bindings xml file

Returns

true if set succeeded, or false if there was an error

6.4.3.27 bool ApplicationObject::setGroupName (const QString & *groupName*)

Setter for parent group's name

Parameters

groupName new parent group's name

Returns

true if group name was set, or false if there was an error

6.4.3.28 bool ApplicationObject::setJoystickMode (quint8 *joystickMode*)

Setter for joystick mode

Parameters

joystickMode new joystick mode

Returns

true if joystick mode was set, or false if there was an error

6.4.3.29 bool ApplicationObject::setName (const QString *name*)

Setter for name

Parameters

name new name

Returns

true if name was set, or false if there was an error

6.4.3.30 void ApplicationObject::setProfileNumber (const unsigned int *profileNumber*)

Setter for profile number

Parameters

profileNumber new profile number

6.4.3.31 bool ApplicationObject::setScrollWheelLines (quint8 *newLines*)

Setter for scroll wheel lines per one click forward/backward

Parameters

newLines new amount of lines

Returns

true if lines were set, or false if there was an error

6.4.3.32 bool ApplicationObject::setSensitivity (quint8 *newSensitivity*)

Setter for mouse sensitivity

Parameters

newSensitivity new sensitivity

Returns

true if sensitivity was set, or false if there was an error

6.4.3.33 bool ApplicationObject::setWindowTitlePart (const QString *windowTilePart*)

Setter for window title part

Parameters

windowTilePart new window title part

Returns

true if window title part was set, or false if there was an error

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

- [src/applicationobject.h](#)
- [src/applicationobject.cpp](#)

6.5 AutoswitchDetailsDialog Class Reference

```
#include <autoswitchdetailsdialog.h>
```

Public Member Functions

- [AutoswitchDetailsDialog](#) (const QString oldFileName, const QString oldWindowTitlePart, QWidget *parent=0)
- [~AutoswitchDetailsDialog](#) ()
- action [getAction](#) (void) const
- QString [getFileName](#) (void) const
- QString [getWindowTitlePart](#) (void) const

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.5.1 Detailed Description

A class which models dialog for changing autoswitch details of an application

6.5.2 Constructor & Destructor Documentation**6.5.2.1 AutoswitchDetailsDialog::AutoswitchDetailsDialog (const QString *oldFileName*, const QString *oldWindowTitlePart*, QWidget * *parent* = 0)**

Constructor

Parameters

oldFileName Previous file name of the binary (required)

oldWindowTitlePart Previous, static part of window title (required)

parent Parent of this QObject (optional)

6.5.2.2 AutoswitchDetailsDialog::~~AutoswitchDetailsDialog ()

Destructor

6.5.3 Member Function Documentation

6.5.3.1 void AutoswitchDetailsDialog::changeEvent (QEvent * *e*) [protected]

Reimplementation of QObject's changeEvent

Parameters

e Event given by Qt

6.5.3.2 action AutoswitchDetailsDialog::getAction (void) const

Was data modified, or was dialog canceled

Returns

MODIFY_ACTION if modified, or CANCEL_ACTION if canceled

6.5.3.3 QString AutoswitchDetailsDialog::getFileName (void) const

Get file name associated with this application.

Returns empty string, if no file is chosen.

Returns

File name, or empty if none

6.5.3.4 QString AutoswitchDetailsDialog::getWindowTitlePart (void) const

Get the part of window title associated with application.

For example, openoffice.org adds the open file name to the window title. Only the static part of the title is given to this dialog, and it is matched as substring of the full title. Returns empty string, if no title was chosen.

Returns

Title part, or empty if none

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

- [src/autoswitchdetailsdialog.h](#)
- [src/autoswitchdetailsdialog.cpp](#)

6.6 ButtonBindingObject Class Reference

```
#include <buttonbindingobject.h>
```

Public Member Functions

- [ButtonBindingObject](#) ([FunctionObject](#) function, quint32 flags=0)
- [ButtonBindingObject](#) (const [ButtonBindingObject](#) &source)
- [FunctionObject](#) [getFunction](#) (void) const
- bool [setFunction](#) (const [FunctionObject](#) function)
- quint32 [getFlags](#) (void) const
- bool [setFlags](#) (quint32 flags)
- [ButtonBindingObject](#) & [operator=](#) (const [ButtonBindingObject](#) &right)
- bool [operator==](#) (const [ButtonBindingObject](#) &target) const

6.6.1 Detailed Description

A class which models a mouse button binding to a function

6.6.2 Constructor & Destructor Documentation

6.6.2.1 [ButtonBindingObject::ButtonBindingObject](#) ([FunctionObject](#) *function*, quint32 *flags* = 0)

Constructor

Parameters

function Function associated with this binding (required)

flags Flags associated with this binding (optional)

6.6.2.2 [ButtonBindingObject::ButtonBindingObject](#) (const [ButtonBindingObject](#) & *source*)

Copy constructor, performs deep copy

Parameters

source [ButtonBindingObject](#) to be copied

6.6.3 Member Function Documentation

6.6.3.1 quint32 [ButtonBindingObject::getFlags](#) (void) const

Getter for flags

Returns

Flags associated with this binding

6.6.3.2 `FunctionObject ButtonBindingObject::getFunction (void) const`

Getter for function

Returns

Function associated with this binding

6.6.3.3 `ButtonBindingObject & ButtonBindingObject::operator= (const ButtonBindingObject & right)`

Operator =, performs deep copy

Parameters

right Right side of assignment

Returns

[ButtonBindingObject](#) reference

6.6.3.4 `bool ButtonBindingObject::operator== (const ButtonBindingObject & target) const`

Operator ==

Parameters

target Binding we compare to

Returns

true if the bindings were the same, or false otherwise

6.6.3.5 `bool ButtonBindingObject::setFlags (quint32 flags)`

Setter for flags

Parameters

flags new flags

Returns

true if flags were set, or false if there was an error

6.6.3.6 `bool ButtonBindingObject::setFunction (const FunctionObject function)`

Setter for function

Parameters

function new function

Returns

true if function was set, or false if there was an error

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

- [src/buttonbindingobject.h](#)
- [src/buttonbindingobject.cpp](#)

6.7 CategoryListModel Class Reference

```
#include <categorylistmodel.h>
```

Public Member Functions

- [CategoryListModel](#) (const QList< [CategoryObject](#) > &categoryList, QObject *parent=0)
- int [rowCount](#) (const QModelIndex &parent=QModelIndex()) const
Implementation of the virtual function rowCount(...) in QAbstractListModel.
- QVariant [data](#) (const QModelIndex &index, int role) const
Implementation of the virtual function data(...) in QAbstractListModel.
- QVariant [headerData](#) (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const
Implementation of the virtual function headerData(...) in QAbstractListModel.

6.7.1 Detailed Description

A class which models a list of categories

6.7.2 Constructor & Destructor Documentation

6.7.2.1 [CategoryListModel::CategoryListModel](#) (const QList< [CategoryObject](#) > & *categoryList*, QObject * *parent* = 0)

Constructor

Parameters

- categoryList* Reference to the category list we are modeling (required)
parent Parent of this QObject (optional)

6.7.3 Member Function Documentation

6.7.3.1 [QVariant CategoryListModel::data](#) (const QModelIndex & *index*, int *role*) const

Implementation of the virtual function data(...) in QAbstractListModel.

Returns the data of the desired category

Parameters

index Reference to the index, from which we get the category data

role See Qt documentation of views

Returns

The data of the category

6.7.3.2 QVariant CategoryListModel::headerData (int section, Qt::Orientation orientation, int role = Qt::DisplayRole) const

Implementation of the virtual function headerData(...) in QAbstractListModel.

Gives the headers for those views that use them

Parameters

section See Qt documentation of views

orientation See Qt documentation of views

role See Qt documentation of views

Returns

The data for the headers

6.7.3.3 int CategoryListModel::rowCount (const QModelIndex & parent = QModelIndex ()) const

Implementation of the virtual function rowCount(...) in QAbstractListModel.

Returns the number of categories in the list

Parameters

parent Parent, for which we return row count (optional)

Returns

The number of categories

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

- [src/categorylistmodel.h](#)
- [src/categorylistmodel.cpp](#)

6.8 CategoryObject Class Reference

```
#include <categoryobject.h>
```

Public Member Functions

- [CategoryObject](#) (QString name)
- [CategoryObject](#) (const [CategoryObject](#) &source)
- [~CategoryObject](#) ()
- QString [getName](#) (void) const
- bool [setName](#) (QString name)
- QList< [FunctionObject](#) > [getFunctionList](#) (void) const
- bool [setFunctionListByNode](#) (QDomNode categoryNode, QList< [FunctionObject](#) > functionList)
- bool [setFunctionBound](#) (const QString functionName, bool bound)
- bool [operator<](#) (const [CategoryObject](#) &target) const
- [CategoryObject](#) & [operator=](#) (const [CategoryObject](#) &right)
- bool [operator==](#) (const [CategoryObject](#) &target) const

6.8.1 Detailed Description

A class which models a category of functions in an application

6.8.2 Constructor & Destructor Documentation

6.8.2.1 [CategoryObject::CategoryObject](#) ([QString](#) *name*)

Constructor

Parameters

name Category's name (required)

6.8.2.2 [CategoryObject::CategoryObject](#) ([const](#) [CategoryObject](#) & *source*)

Copy constructor, performs deep copy

Parameters

source [CategoryObject](#) to be copied

6.8.2.3 [CategoryObject::~~CategoryObject](#) ()

Destructor

6.8.3 Member Function Documentation

6.8.3.1 [QList](#)< [FunctionObject](#) > [CategoryObject::getFunctionList](#) ([void](#)) const

Getter for function list

Returns

function list

6.8.3.2 `QString CategoryObject::getName (void) const`

Getter for name

Returns

Category's name

6.8.3.3 `bool CategoryObject::operator< (const CategoryObject & target) const`

Operator <, compares category's name

Parameters

target Category we compare to

Returns

as (category.getName() < target.getName())

6.8.3.4 `CategoryObject & CategoryObject::operator= (const CategoryObject & right)`

Operator =, performs deep copy

Parameters

right Right side of assignment

Returns

[CategoryObject](#) reference

6.8.3.5 `bool CategoryObject::operator== (const CategoryObject & target) const`

Operator ==

Parameters

target Category we compare to

Returns

true if the names are the same, or false otherwise

6.8.3.6 `bool CategoryObject::setFunctionBound (const QString functionName, bool bound)`

Sets a particular function as bound or unbound

Parameters

functionName Name of the function

bound Do we set bound or unbound?

Returns

true if set succeeded, or false if there was an error

6.8.3.7 bool CategoryObject::setFunctionListByNode (QDomNode *categoryNode*, QList<FunctionObject > *functionList*)

Receives a Dom node, and sets the function list according to it

Parameters

categoryNode Dom node to this category

functionList The parent ApplicationObject's list to FunctionObjects.

Returns

true if set succeeded, or false if there was an error

6.8.3.8 bool CategoryObject::setName (QString *name*)

Setter for name

Parameters

name new name

Returns

true if name was set, or false if there was an error

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

- [src/categoryobject.h](#)
- [src/categoryobject.cpp](#)

6.9 charToKeyInfo Struct Reference

Public Attributes

- const char **character**
- const char * **GUIText**
- int **keyQt**
- quint8 **HID**
- int **forcedModifier**
- bool **numPad**

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

- [src/keypress_common.h](#)

6.10 CopyApplicationDialog Class Reference

```
#include <copyapplicationdialog.h>
```

Public Member Functions

- [CopyApplicationDialog](#) (QList< [GroupObject](#) > groupList, int selectedGroup, QWidget *parent=0)
- QString [getApplicationName](#) (void) const
- int [getSelectedGroup](#) (void) const
- [~CopyApplicationDialog](#) ()
- bool [isOk](#) (void) const

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.10.1 Detailed Description

A class which models copy application dialog.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 CopyApplicationDialog::CopyApplicationDialog (QList< GroupObject > *groupList*, int *selectedGroup*, QWidget * *parent* = 0)

Constructor

Parameters

- groupList* Reference to MainWindow's group list (required)
- selectedGroup* Which group index is selected for copying
- parent* Parent of this QObject (optional)

6.10.2.2 CopyApplicationDialog::~~CopyApplicationDialog ()

Destructor

6.10.3 Member Function Documentation

6.10.3.1 void CopyApplicationDialog::changeEvent (QEvent * *e*) [protected]

Reimplementation of QObject's changeEvent

Parameters

- e* Event given by Qt

6.10.3.2 QString CopyApplicationDialog::getApplicationName (void) const

Get new name of application.

Returns

- Name of new application

6.10.3.3 int CopyApplicationDialog::getSelectedGroup (void) const

Get group index to where we place new application

Returns

Group index

6.10.3.4 bool CopyApplicationDialog::isOk (void) const

Did user click OK?

Returns

true if user clicked OK, or false if dialog was canceled

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

- [src/copyapplicationdialog.h](#)
- [src/copyapplicationdialog.cpp](#)

6.11 CopyCategoryDialog Class Reference

```
#include <copycategorydialog.h>
```

Public Member Functions

- [CopyCategoryDialog](#) (const QList< [GroupObject](#) > &groupList, int activeGroupIndex, int activeApplicationIndex, QWidget *parent=0)
- [~CopyCategoryDialog](#) ()
- [QString](#) [getGroupName](#) (void) const
- [QString](#) [getApplicationName](#) (void) const
- [QString](#) [getCategoryName](#) (void) const

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.11.1 Detailed Description

A class which models copy category dialog.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 CopyCategoryDialog::CopyCategoryDialog (const QList< GroupObject > & groupList, int activeGroupIndex, int activeApplicationIndex, QWidget * parent = 0)

Constructor

Parameters

- groupList* Reference to MainWindow's group list (required)
- activeGroupIndex* MainWindow's active group index (required)
- activeApplicationIndex* MainWindow's active application index (required)
- parent* Parent of this QObject (optional)

6.11.2.2 CopyCategoryDialog::~~CopyCategoryDialog ()

Destructor

6.11.3 Member Function Documentation

6.11.3.1 void CopyCategoryDialog::changeEvent (QEvent * *e*) [protected]

Reimplementation of QObject's changeEvent

Parameters

- e* Event given by Qt

6.11.3.2 QString CopyCategoryDialog::getApplicationName (void) const

Getter for application name, for the newly copied category

Returns

Application name

6.11.3.3 QString CopyCategoryDialog::getCategoryName (void) const

Getter for the newly copied category's name

Returns

Category name

6.11.3.4 QString CopyCategoryDialog::getGroupName (void) const

Getter for group name, for the newly copied category

Returns

Group name

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

- [src/copycategorydialog.h](#)
- [src/copycategorydialog.cpp](#)

6.12 CopyFunctionDialog Class Reference

```
#include <copyfunctiondialog.h>
```

Public Member Functions

- [CopyFunctionDialog](#) (const QList< [GroupObject](#) > &groupList, int activeGroupIndex, int activeApplicationIndex, int activeCategoryIndex, QWidget *parent)
- [~CopyFunctionDialog](#) ()
- QString [getGroupName](#) (void) const
- QString [getApplicationName](#) (void) const
- QString [getCategoryName](#) (void) const
- QString [getFunctionName](#) (void) const

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.12.1 Detailed Description

A class which models copy function dialog.

6.12.2 Constructor & Destructor Documentation

6.12.2.1 CopyFunctionDialog::CopyFunctionDialog (const QList< GroupObject > & groupList, int activeGroupIndex, int activeApplicationIndex, int activeCategoryIndex, QWidget * parent)

Constructor

Parameters

- groupList* Reference to MainWindow's group list (required)
- activeGroupIndex* MainWindow's active group index (required)
- activeApplicationIndex* MainWindow's active application index (required)
- activeCategoryIndex* MainWindow's active category index (required)
- parent* Parent of this QObject (optional)

6.12.2.2 CopyFunctionDialog::~CopyFunctionDialog ()

Destructor

6.12.3 Member Function Documentation

6.12.3.1 void CopyFunctionDialog::changeEvent (QEvent * e) [protected]

Reimplementation of QObject's changeEvent

Parameters

e Event given by Qt

6.12.3.2 QString CopyFunctionDialog::getApplicationName (void) const

Getter for application name, for the newly copied function

Returns

Application name

6.12.3.3 QString CopyFunctionDialog::getCategoryName (void) const

Getter for category name, for the newly copied function

Returns

Category name

6.12.3.4 QString CopyFunctionDialog::getFunctionName (void) const

Getter for newly copied function's name

Returns

Function name

6.12.3.5 QString CopyFunctionDialog::getGroupName (void) const

Getter for group name, for the newly copied function

Returns

Group name

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

- [src/copyfunctiondialog.h](#)
- [src/copyfunctiondialog.cpp](#)

6.13 EditApplicationDialog Class Reference

```
#include <editapplicationdialog.h>
```

Public Member Functions

- [EditApplicationDialog](#) (const QList< [GroupObject](#) > &groupList, const QString oldName, unsigned int oldProfile, const QString oldFileName, const QString oldWindowTitlePart, quint8 oldCPI, quint8 oldCPI_low, quint8 oldCPI_high, quint8 oldJoystickMode, QWidget *parent=0)
- virtual [~EditApplicationDialog](#) ()
- bool [getReplacedApplication](#) (QString &applicationToReplace) const
- unsigned int [getProfileForReplaced](#) (void) const
- action [getAction](#) (void) const
- QString [getName](#) (void) const
- unsigned int [getNewProfileNumber](#) (void) const
- QString [getFileName](#) (void) const
- QString [getWindowTitlePart](#) (void) const
- quint8 [getCPI](#) (void) const
- quint8 [getCPI_low](#) (void) const
- quint8 [getCPI_high](#) (void) const
- quint8 [getJoystickMode](#) (void) const

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.13.1 Detailed Description

A class which models edit application dialog.

6.13.2 Constructor & Destructor Documentation

- 6.13.2.1 [EditApplicationDialog::EditApplicationDialog](#) (const QList< [GroupObject](#) > &groupList, const QString oldName, unsigned int oldProfile, const QString oldFileName, const QString oldWindowTitlePart, quint8 oldCPI, quint8 oldCPI_low, quint8 oldCPI_high, quint8 oldJoystickMode, QWidget * parent = 0) [explicit]**

Constructor

Parameters

- groupList* Reference to MainWindow's group list (required)
- oldName* Previous name of application being edited (required)
- oldProfile* Previous profile number of application being edited (required)
- oldFileName* Previous file name associated with application being edited (required)
- oldWindowTitlePart* Previous window title part associated with application being edited (required)
- oldCPI* Previous CPI, in mouse's internal format (required)
- oldCPI_low* Previous low CPI, in mouse's internal format (required)
- oldCPI_high* Previous high CPI, in mouse's internal format (required)
- oldJoystickMode* Previous joystick mode (required)
- parent* Parent of this QObject (optional)

6.13.2.2 `EditApplicationDialog::~~EditApplicationDialog () [virtual]`

Destructor

6.13.3 Member Function Documentation

6.13.3.1 `void EditApplicationDialog::changeEvent (QEvent * e) [protected]`

Reimplementation of `QObject`'s `changeEvent`

Parameters

e Event given by Qt

6.13.3.2 `action EditApplicationDialog::getAction (void) const`

Was application removed, modified, or was dialog canceled?

Returns

`MODIFY_ACTION` if modified, `CANCEL_ACTION` if canceled, or `REMOVE_ACTION` if removed

6.13.3.3 `quint8 EditApplicationDialog::getCPI (void) const`

Get edited application's new default CPI, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

CPI in internal format

6.13.3.4 `quint8 EditApplicationDialog::getCPI_high (void) const`

Get edited application's new high CPI, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

CPI in internal format

6.13.3.5 `quint8 EditApplicationDialog::getCPI_low (void) const`

Get edited application's new low CPI, in mouse's internal format

$((\text{actual CPI} - 100) / 25) = \text{internal CPI}$, and $((\text{internal CPI} * 25) + 100) = \text{actual CPI}$

Returns

CPI in internal format

6.13.3.6 QString EditApplicationDialog::getFileName (void) const

Get new file name associated with edited application.

Returns empty string, if no file was chosen.

Returns

File name, or empty if none

6.13.3.7 quint8 EditApplicationDialog::getJoystickMode (void) const

Get edited application's new joystick mode

Mode numbers are defined in [setup_common.h](#)

Returns

Joystick mode number

6.13.3.8 QString EditApplicationDialog::getName (void) const

Get new name of edited application.

Returns

New name

6.13.3.9 unsigned int EditApplicationDialog::getNewProfileNumber (void) const

Get new profile number of edited application.

Returns

New profile number

6.13.3.10 unsigned int EditApplicationDialog::getProfileForReplaced (void) const

Get new profile number of application replaced by moving this application to its profile number.

Check first with `getReplacedApplication` that something was replaced, because return value 0 is ambiguous.

Returns

New profile number of replaced application

6.13.3.11 bool EditApplicationDialog::getReplacedApplication (QString & *applicationToReplace*) const

Get name of application replaced when moving this application to another profile number.

Parameters

applicationToReplace Reference to where the name of replaced app is placed

Returns

true if application was replaced, or false if no application was replaced

6.13.3.12 QString EditApplicationDialog::getWindowTitlePart (void) const

Get the new part of window title associated with edited application.

For example, openoffice.org adds the open file name to the window title. Only the static part of the title is given to this dialog, and it is matched as substring of the full title. Returns empty string, if no title was chosen.

Returns

Title part, or empty if none

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

- [src/editapplicationdialog.h](#)
- [src/editapplicationdialog.cpp](#)

6.14 EditCategoryDialog Class Reference

```
#include <editcategorydialog.h>
```

Public Member Functions

- [EditCategoryDialog](#) (const QList< [GroupObject](#) > &groupList, const QString categoryName, int activeGroupIndex, int activeApplicationIndex, QWidget *parent=0)
- virtual [~EditCategoryDialog](#) ()
- QString [getCategoryName](#) (void) const
- QString [getCopiedGroupName](#) (void) const
- QString [getCopiedApplicationName](#) (void) const
- QString [getCopiedCategoryName](#) (void) const
- action [getAction](#) (void) const

Protected Member Functions

- virtual void [changeEvent](#) (QEvent *e)

6.14.1 Detailed Description

A class which models edit category dialog.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 EditCategoryDialog::EditCategoryDialog (const QList< GroupObject > & *groupList*, const QString *categoryName*, int *activeGroupIndex*, int *activeApplicationIndex*, QWidget * *parent* = 0) [explicit]

Constructor

Parameters

groupList Reference to MainWindow's group list (required)
categoryName Previous name of category being edited (required)
activeGroupIndex Active group index from [MainWindow](#) (required)
activeApplicationIndex Active application index from [MainWindow](#) (required)
parent Parent of this QObject (optional)

6.14.2.2 EditCategoryDialog::~EditCategoryDialog () [virtual]

Destructor

6.14.3 Member Function Documentation

6.14.3.1 void EditCategoryDialog::changeEvent (QEvent * *e*) [protected, virtual]

Reimplementation of QObject's changeEvent

Parameters

e Event given by Qt

6.14.3.2 action EditCategoryDialog::getAction (void) const

Was category removed, modified, copied or was dialog canceled?

Returns

MODIFY_ACTION if modified, CANCEL_ACTION if canceled, REMOVE_ACTION if removed, or COPY_ACTION if copied

6.14.3.3 QString EditCategoryDialog::getCategoryName (void) const

Get name of modified category.

Returns

Name of modified category

6.14.3.4 QString EditCategoryDialog::getCopiedApplicationName (void) const

Name of application to which we copied category, if any

Returns

Name of application, or empty if category not copied

6.14.3.5 QString EditCategoryDialog::getCopiedCategoryName (void) const

New name for copied category, if any

Returns

Name of new category, or empty if category not copied

6.14.3.6 QString EditCategoryDialog::getCopiedGroupName (void) const

Name of group to which we copied category, if any

Returns

Name of group, or empty if category not copied

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

- [src/editcategorydialog.h](#)
- [src/editcategorydialog.cpp](#)

6.15 EditFunctionDialog Class Reference

```
#include <editfunctiondialog.h>
```

Public Member Functions

- [EditFunctionDialog](#) (const QString applicationName, const QString functionName, const QString data, const [functionType](#) type, const QList< [GroupObject](#) > &groupList, int activeGroupIndex, int activeApplicationIndex, int activeCategoryIndex, QWidget *parent=0)
- virtual [~EditFunctionDialog](#) (void)
- QString [getKeypresses](#) (void) const
- bool [setKeypresses](#) (const QString newKeypresses)
- [functionType](#) [getType](#) (void)
- action [getAction](#) (void) const
- QString [getFunctionName](#) (void) const
- QString [getCopiedGroupName](#) (void) const
- QString [getCopiedApplicationName](#) (void) const
- QString [getCopiedCategoryName](#) (void) const
- QString [getCopiedFunctionName](#) (void) const

Protected Member Functions

- bool [eventFilter](#) (QObject *target, QEvent *event)
- virtual void [changeEvent](#) (QEvent *e)
- void [keyPressEvent](#) (QKeyEvent *event)
- void [keyReleaseEvent](#) (QKeyEvent *event)

6.15.1 Detailed Description

A class which models edit function dialog.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 `EditFunctionDialog::EditFunctionDialog (const QString applicationName, const QString functionName, const QString data, const functionType type, const QList< GroupObject > & groupList, int activeGroupIndex, int activeApplicationIndex, int activeCategoryIndex, QWidget * parent = 0) [explicit]`

Constructor

Parameters

applicationName Name of application where we are editing function (required)

functionName Name of function we are editing (required)

data String representation of macro/key/keypress (required)

type Type of function we are editing (required)

groupList Reference to MainWindow's group list (required)

activeGroupIndex Active group index from [MainWindow](#) (required)

activeApplicationIndex Active application index from [MainWindow](#) (required)

activeCategoryIndex Active application index from [MainWindow](#) (required)

parent Parent of this QObject (optional)

6.15.2.2 `EditFunctionDialog::~EditFunctionDialog (void) [virtual]`

Destructor

6.15.3 Member Function Documentation

6.15.3.1 `void EditFunctionDialog::changeEvent (QEvent * e) [protected, virtual]`

Reimplementation of QObject's changeEvent

Parameters

e Event given by Qt

6.15.3.2 `bool EditFunctionDialog::eventFilter (QObject * target, QEvent * event)` `[protected]`

Reimplementation of QObject's eventFilter

Parameters

target QObject that the event was directed to

event The event that occurred

Returns

true if we stop event from being handled further, or false if some other target may handle the event

6.15.3.3 `action EditFunctionDialog::getAction (void) const`

Was function removed, modified, copied or was dialog canceled?

Returns

MODIFY_ACTION if modified, CANCEL_ACTION if canceled, REMOVE_ACTION if removed, or COPY_ACTION if copied

6.15.3.4 `QString EditFunctionDialog::getCopiedApplicationName (void) const`

Name of application to which we copied function, if any

Returns

Name of application, or empty if function not copied

6.15.3.5 `QString EditFunctionDialog::getCopiedCategoryName (void) const`

Name of category to which we copied function, if any

Returns

Name of category, or empty if function not copied

6.15.3.6 `QString EditFunctionDialog::getCopiedFunctionName (void) const`

New name for copied function, if any

Returns

Name of new function, or empty if function not copied

6.15.3.7 QString EditFunctionDialog::getCopiedGroupName (void) const

Name of group to which we copied function, if any

Returns

Name of group, or empty if function not copied

6.15.3.8 QString EditFunctionDialog::getFunctionName (void) const

Get name of modified function.

Returns

Name of modified function

6.15.3.9 QString EditFunctionDialog::getKeypresses (void) const

Get keypresses of modified function in the internal representation of the xml file.

Assumes that string contents have been placed to keypresses_ using macroToData().

Returns

Keypresses

See also

[setKeypresses](#)

6.15.3.10 functionType EditFunctionDialog::getType (void)

Get function type.

Returns

Function type

6.15.3.11 void EditFunctionDialog::keyPressEvent (QKeyEvent * event) [protected]

Handler for keypress events

Parameters

event Event given by Qt

6.15.3.12 void EditFunctionDialog::keyReleaseEvent (QKeyEvent * event) [protected]

Handler for key release events

Parameters

event Event given by Qt

6.15.3.13 bool EditFunctionDialog::setKeypresses (const QString *newKeypresses*)

Set keypresses to modified function in the internal representation of the xml file.

Parameters

newKeypresses Keypresses to set

Returns

true if keypresses were accepted, or false if data was incorrect

See also

[getKeypresses](#)

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

- [src/editfunctiondialog.h](#)
- [src/editfunctiondialog.cpp](#)

6.16 FunctionDelegate Class Reference

```
#include <functiondelegate.h>
```

Public Member Functions

- [FunctionDelegate](#) (QObject *parent=0)
- void [paint](#) (QPainter *painter, const QStyleOptionViewItem &option, const QModelIndex &index) const

6.16.1 Detailed Description

A class which defines how to edit and display functions in an editor or view

6.16.2 Constructor & Destructor Documentation

6.16.2.1 FunctionDelegate::FunctionDelegate (QObject * *parent* = 0)

Constructor

Parameters

parent Parent of this QObject (optional)

6.16.3 Member Function Documentation

6.16.3.1 void FunctionDelegate::paint (QPainter * *painter*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) const

Reimplementation of QItemDelegate's paint function

Parameters

painter See Qt's documentation of QItemDelegate

option See Qt's documentation of QItemDelegate

index See Qt's documentation of QItemDelegate

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

- [src/functiondelegate.h](#)
- [src/functiondelegate.cpp](#)

6.17 FunctionListModel Class Reference

```
#include <functionlistmodel.h>
```

Public Member Functions

- **FunctionListModel** (const QList< **FunctionObject** > &functionList, QObject *parent=0)
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const
Implementation of the virtual function rowCount(...) in QAbstractListModel.
- QVariant **data** (const QModelIndex &index, int role) const
Implementation of the virtual function data(...) in QAbstractListModel.
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const
Implementation of the virtual function headerData(...) in QAbstractListModel.

6.17.1 Detailed Description

A class which models a list of functions

6.17.2 Constructor & Destructor Documentation**6.17.2.1 FunctionListModel::FunctionListModel (const QList< FunctionObject > &functionList, QObject * parent = 0)**

Constructor

Parameters

functionList Reference to the function list we are modeling (required)

parent Parent of this QObject (optional)

6.17.3 Member Function Documentation

6.17.3.1 QVariant FunctionListModel::data (const QModelIndex & *index*, int *role*) const

Implementation of the virtual function data(...) in QAbstractListModel.

Returns the data of the desired function

Parameters

index Reference to the index, from which we get the function data

role See Qt documentation of views

Returns

The data of the function

6.17.3.2 QVariant FunctionListModel::headerData (int *section*, Qt::Orientation *orientation*, int *role* = Qt::DisplayRole) const

Implementation of the virtual function headerData(...) in QAbstractListModel.

Gives the headers for those views that use them

Parameters

section See Qt documentation of views

orientation See Qt documentation of views

role See Qt documentation of views

Returns

The data for the headers

6.17.3.3 int FunctionListModel::rowCount (const QModelIndex & *parent* = QModelIndex()) const

Implementation of the virtual function rowCount(...) in QAbstractListModel.

Returns the number of functions in the list

Parameters

parent Parent, for which we return row count (optional)

Returns

The number of functions

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

- [src/functionlistmodel.h](#)
- [src/functionlistmodel.cpp](#)

6.18 FunctionObject Class Reference

```
#include <functionobject.h>
```

Public Member Functions

- [FunctionObject](#) (QString name="", QString applicationName="", [functionType](#) type=NONE, QString data="")
- [FunctionObject](#) (const [FunctionObject](#) &source)
- QString [getName](#) (void) const
- bool [setName](#) (QString name)
- QString [getApplicationName](#) (void) const
- bool [setApplicationName](#) (QString applicationName)
- [functionType](#) [getType](#) (void) const
- bool [setType](#) ([functionType](#) type)
- QString [getData](#) (void) const
- bool [setData](#) (QString data)
- bool [isBound](#) (void) const
- void [setBound](#) (bool bound)
- bool [operator<](#) (const [FunctionObject](#) &target) const
- bool [operator>](#) (const [FunctionObject](#) &target) const
- bool [operator==](#) (const [FunctionObject](#) &right) const
- [FunctionObject](#) & [operator=](#) (const [FunctionObject](#) &right)

6.18.1 Detailed Description

A class which models a function of an application

6.18.2 Constructor & Destructor Documentation

6.18.2.1 [FunctionObject::FunctionObject](#) ([QString](#) *name* = "", [QString](#) *applicationName* = "", [functionType](#) *type* = *NONE*, [QString](#) *data* = "")

Constructor

Parameters

- name* Function name (optional)
- applicationName* Parent application's name (optional)
- type* How this is implemented. See [functionType](#). (optional)
- data* Key/keypress/macro in internal string format (optional)

6.18.2.2 [FunctionObject::FunctionObject](#) (const [FunctionObject](#) & *source*)

Copy constructor

Parameters

- source* [FunctionObject](#) to be copied

6.18.3 Member Function Documentation

6.18.3.1 QString FunctionObject::getApplicationName (void) const

Getter for parent application's name

Returns

Application's name

6.18.3.2 QString FunctionObject::getData (void) const

Getter for data

Returns

Function's data

6.18.3.3 QString FunctionObject::getName (void) const

Getter for name

Returns

Function's name

6.18.3.4 functionType FunctionObject::getType (void) const

Getter for type

Returns

Function's type

6.18.3.5 bool FunctionObject::isBound (void) const

Getter for bound

Returns

Is function bound to a button?

6.18.3.6 bool FunctionObject::operator< (const FunctionObject & *target*) const

Operator <, compares function's name

Parameters

target Function we compare to

Returns

as (function.getName() < target.getName())

6.18.3.7 FunctionObject & FunctionObject::operator= (const FunctionObject & *right*)

Operator =, performs deep copy

Parameters

right Right side of assignment

Returns

FunctionObject reference

6.18.3.8 bool FunctionObject::operator==(const FunctionObject & *right*) const

Operator ==, compares function's name

Parameters

right Function we compare to

Returns

as (function.getName() == target.getName())

6.18.3.9 bool FunctionObject::operator> (const FunctionObject & *target*) const

Operator >, compares function's name

Parameters

target Function we compare to

Returns

as (function.getName() > target.getName())

6.18.3.10 bool FunctionObject::setApplicationName (QString *applicationName*)

Setter for parent application's name

Parameters

applicationName new application name

Returns

true if application name was set, or false if there was an error

6.18.3.11 void FunctionObject::setBound (bool *bound*)

Setter for bound

Parameters

bound new value for bound

6.18.3.12 bool FunctionObject::setData (QString *data*)

Setter for data

Parameters

data new data

Returns

true if data was set, or false if there was an error

6.18.3.13 bool FunctionObject::setName (QString *name*)

Setter for name

Parameters

name new name

Returns

true if name was set, or false if there was an error

6.18.3.14 bool FunctionObject::setType (functionType *type*)

Setter for type

Parameters

type new type

Returns

true if type was set, or false if there was an error

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

- [src/functionobject.h](#)
- [src/functionobject.cpp](#)

6.19 GroupObject Class Reference

```
#include <groupobject.h>
```

Public Member Functions

- [GroupObject](#) (QString name)
- [~GroupObject](#) ()
- [GroupObject](#) (const [GroupObject](#) &source)
- QString [getName](#) (void) const

- bool `setName` (QString name)
- QList< ApplicationObject > `getApplicationList` (bool reverse=false, bool byProfileNumber=false) const
- bool `setApplicationListByNode` (QDomNode groupNode, QDomNode bindingsNode, bool profileNumberUsed[])
- bool `setFunctionBound` (const QString applicationName, const QString functionName, bool bound)
- bool `operator<` (const GroupObject &target) const
- GroupObject & `operator=` (const GroupObject &right)
- bool `operator==` (const GroupObject &target) const

6.19.1 Detailed Description

A class which models a group of applications

6.19.2 Constructor & Destructor Documentation

6.19.2.1 GroupObject::GroupObject (QString *name*)

Constructor

Parameters

name Group's name (required)

6.19.2.2 GroupObject::~~GroupObject ()

Destructor

6.19.2.3 GroupObject::GroupObject (const GroupObject & *source*)

Copy constructor, performs deep copy

Parameters

source GroupObject to be copied

6.19.3 Member Function Documentation

6.19.3.1 QList< ApplicationObject > GroupObject::getApplicationList (bool *reverse* = *false*, bool *byProfileNumber* = *false*) const

Getter for list of applications in this group

Parameters

reverse True if we get reversed list, or false otherwise

byProfileNumber True if we get list ordered by profile number, or false otherwise

Returns

application list

6.19.3.2 QString GroupObject::getName (void) const

Getter for name

Returns

Group's name

6.19.3.3 bool GroupObject::operator< (const GroupObject & *target*) const

Operator <, compares group's name

Parameters

target Group we compare to

Returns

as (group.getName() < target.getName())

6.19.3.4 GroupObject & GroupObject::operator= (const GroupObject & *right*)

Operator =, performs deep copy

Parameters

right Right side of assignment

Returns

[GroupObject](#) reference

6.19.3.5 bool GroupObject::operator== (const GroupObject & *target*) const

Operator ==

Parameters

target Group we compare to

Returns

true if the names are the same, or false otherwise

6.19.3.6 bool GroupObject::setApplicationListByNode (QDomNode *groupNode*, QDomNode *bindingsNode*, bool *profileNumberUsed*[])

Receives a Dom node, and sets the applicationList_ according to it

Parameters

groupNode Dom node to this group

bindingsNode Dom node to bindings

profileNumberUsed Which profile numbers are taken? Profile number 1 is index 0. Will update.

Returns

true if set succeeded, or false if there was an error

6.19.3.7 bool GroupObject::setFunctionBound (const QString *applicationName*, const QString *functionName*, bool *bound*)

Sets a particular function of given application as bound or unbound

Parameters

applicationName Name of the application where the function is

functionName Name of the function

bound Do we set bound or unbound?

Returns

true if set succeeded, or false if there was an error

6.19.3.8 bool GroupObject::setName (QString *name*)

Setter for name

Parameters

name new name

Returns

true if name was set, or false if there was an error

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

- [src/groupobject.h](#)
- [src/groupobject.cpp](#)

6.20 GroupOrAppWidget Class Reference

```
#include <grouporappwidget.h>
```

Signals

- void **buttonPressed** (const unsigned int button)

Public Member Functions

- [GroupOrAppWidget](#) (QWidget *parent=0)
- void [setNumButtons](#) (unsigned int numButtons)
- void [setHaveNavigationTab](#) (bool haveNavigationTab)

Protected Member Functions

- void [mousePressEvent](#) (QMouseEvent *event)

6.20.1 Detailed Description

A class which models a header of buttons for choosing a group or application
The top rows of tabs in advanced view are done with this class

6.20.2 Constructor & Destructor Documentation

6.20.2.1 GroupOrAppWidget::GroupOrAppWidget (QWidget * *parent* = 0)

Constructor

Parameters

parent Parent of this QObject (optional)

6.20.3 Member Function Documentation

6.20.3.1 void GroupOrAppWidget::setHaveNavigationTab (bool *haveNavigationTab*)

Set, whether or not we have a navigation tab in widget

Parameters

haveNavigationTab True if have navigation tab, false otherwise

6.20.3.2 void GroupOrAppWidget::setNumButtons (unsigned int *numButtons*)

Set number of group or application buttons in the widget

Parameters

numButtons New number of buttons.

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

- [src/grouporappwidget.h](#)
- [src/grouporappwidget.cpp](#)

6.21 HardwareThread Class Reference

```
#include <hardwarethread.h>
```

Public Member Functions

- [HardwareThread](#) (QList< [GroupObject](#) > &groupList)
- void [chooseTask](#) (tasks task)
- quint64 [getLastDifferenceResult](#) (void)
- void [run](#) ()

6.21.1 Detailed Description

A class which does those mouse hardware related operations that take too long to do in main loop
If main loop stays in a method too long, the window becomes unresponsive, hence the need for threads.

6.21.2 Constructor & Destructor Documentation

6.21.2.1 HardwareThread::HardwareThread (QList< GroupObject > & *groupList*)

Constructor

Parameters

groupList Reference to MainWindow's group list (required)

6.21.3 Member Function Documentation

6.21.3.1 void HardwareThread::chooseTask (tasks *task*)

Choose, which task the thread does

Parameters

task The task to do, see structure tasks.

6.21.3.2 quint64 HardwareThread::getLastDifferenceResult (void)

Differences between mouse and software contents in last check

Returns

64 bit number, the bits of which mean if there was a difference. 1 = yes, 0 = no

6.21.3.3 void HardwareThread::run (void)

Run the thread

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

- [src/hardwarethread.h](#)
- [src/hardwarethread.cpp](#)

6.22 IdentifiedPushButton Class Reference

```
#include <identifiedpushbutton.h>
```

Signals

- void **clicked** (QString id_)

Public Member Functions

- [IdentifiedPushButton](#) (const QString &id, const QString &text, QWidget *parent=0)
- void [setColor](#) (const QString &color)

6.22.1 Detailed Description

A class which models a button widget for choosing a group or application

This button has an identifier that is given when it is activated, and it can be unrelated to the button text

6.22.2 Constructor & Destructor Documentation

6.22.2.1 IdentifiedPushButton::IdentifiedPushButton (const QString & *id*, const QString & *text*, QWidget * *parent* = 0)

Constructor

Parameters

- id* Internal identifier string of this button (required)
- text* Button text, shown to user (required)
- parent* Parent of this QObject (optional)

6.22.3 Member Function Documentation

6.22.3.1 void IdentifiedPushButton::setColor (const QString & *color*)

Set color of button text

Parameters

- color* String representation of text color

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

- [src/identifiedpushbutton.h](#)
- [src/identifiedpushbutton.cpp](#)

6.23 KeyObject Class Reference

```
#include <keyobject.h>
```

Public Member Functions

- [KeyObject](#) (const [MyKey](#) key, const [QList](#)< [MyKey](#) > modifiers, unsigned int delayMs1=DEFAULT_DELAY_MS_1, unsigned int delayMs2=DEFAULT_DELAY_MS_2)
- virtual [~KeyObject](#) (void)
- [KeyObject](#) (const [KeyObject](#) &source)

Copy constructor.

- [MyKey](#) [getKey](#) (void) const
- void [setKey](#) (const [MyKey](#) key)
- [QList](#)< [MyKey](#) > [getModifiers](#) (void) const
- void [setModifiers](#) (const [QList](#)< [MyKey](#) > modifiers)
- unsigned int [getDelayMs1](#) (void) const
- void [setDelayMs1](#) (unsigned int delayMs1)
- unsigned int [getDelayMs2](#) (void) const
- void [setDelayMs2](#) (unsigned int delayMs2)
- [KeyObject](#) & [operator=](#) (const [KeyObject](#) &right)

Operator =, performs deep copy.

6.23.1 Detailed Description

A class which models a keypress, with its modifiers and delays

6.23.2 Constructor & Destructor Documentation

6.23.2.1 [KeyObject::KeyObject](#) (const [MyKey](#) key, const [QList](#)< [MyKey](#) > modifiers, unsigned int delayMs1 = **DEFAULT_DELAY_MS_1**, unsigned int delayMs2 = **DEFAULT_DELAY_MS_2**)

Constructor

Parameters

- key* Key (HID and Qt representation in [MyKey](#)) (required)
- modifiers* List of modifiers associated with this key (required)
- delayMs1* Delay before key, in milliseconds (optional)
- delayMs2* How many milliseconds key is down (optional)

6.23.2.2 `KeyObject::~~KeyObject (void) [virtual]`

Destructor

6.23.2.3 `KeyObject::KeyObject (const KeyObject & source)`

Copy constructor.

Parameters

source [KeyObject](#) to be copied

6.23.3 Member Function Documentation**6.23.3.1** `unsigned int KeyObject::getDelayMs1 (void) const`

Getter for pre-delay

Returns

Delay before key, in milliseconds

6.23.3.2 `unsigned int KeyObject::getDelayMs2 (void) const`

Getter for down-delay

Returns

Number of milliseconds this key is down

6.23.3.3 `MyKey KeyObject::getKey (void) const`

Getter for key

Returns

The key associated with this object

6.23.3.4 `QList< MyKey > KeyObject::getModifiers (void) const`

Getter for modifier list

Returns

The modifier list associated with this object

6.23.3.5 KeyObject & KeyObject::operator= (const KeyObject & *right*)

Operator =, performs deep copy.

Parameters

right Right side of assignment

Returns

[KeyObject](#) reference

6.23.3.6 void KeyObject::setDelayMs1 (unsigned int *delayMs1*)

Setter for pre-delay

Parameters

delayMs1 New delay before key, in milliseconds

6.23.3.7 void KeyObject::setDelayMs2 (unsigned int *delayMs2*)

Setter for down-delay

Parameters

delayMs2 New number of milliseconds this key is down

6.23.3.8 void KeyObject::setKey (const MyKey *key*)

Setter for key

Parameters

key New key to be placed in this object

6.23.3.9 void KeyObject::setModifiers (const QList< MyKey > *modifiers*)

Setter for modifier list

Parameters

modifiers New modifier list to be placed in this object

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

- [src/keyobject.h](#)
- [src/keyobject.cpp](#)

6.24 KeypressParser Class Reference

```
#include <keypressparser.h>
```

Public Member Functions

- bool [keypressStringToHID](#) (const QString keypressString, QList< [MyKeypressHID](#) > &keypressesHID, QString &errorDetails)
- bool [keypressStringToQt](#) (const QString keypressString, [KeyObject](#) &keyObjectQt, QString &errorDetails, bool acceptRepresentation=true)
- bool [macroStringToHID](#) (const QString macroString, QList< [MyKeypressHID](#) > &keypressesHID, QString &errorDetails)
- bool [macroStringToQt](#) (const QString macroString, QList< [KeyObject](#) > &keyObjects, QString &errorDetails, bool acceptRepresentation=true)
- bool [getMacroInternalString](#) (const QList< [KeyObject](#) > macro, QString &representation) const
- bool [getMacroGUIString](#) (const QList< [KeyObject](#) > macro, QString &representation) const

Static Public Member Functions

- static [KeypressParser](#) & [instance](#) ()

6.24.1 Detailed Description

A Parser class for the keypress sequence

Conforms to the Singleton design pattern

6.24.2 Member Function Documentation

6.24.2.1 bool KeypressParser::getMacroGUIString (const QList< [KeyObject](#) > *macro*, QString & *representation*) const

Convert macro as list of KeyObjects to GUI string representation.

Parameters

macro List of keyobjects

representation Reference to where the string is placed

Returns

true if converted ok, or false if there was an error in input

6.24.2.2 bool KeypressParser::getMacroInternalString (const QList< [KeyObject](#) > *macro*, QString & *representation*) const

Convert macro as list of KeyObjects to internal string representation.

Parameters

macro List of keyobjects

representation Reference to where the string is placed

Returns

true if converted ok, or false if there was an error in input

6.24.2.3 static KeypressParser& KeypressParser::instance () [inline, static]

Gives a reference to the Singleton object

Returns

reference to the object

The Singleton instance

6.24.2.4 bool KeypressParser::keypressStringToHID (const QString *keypressString*, QList< MyKeypressHID > & *keypressesHID*, QString & *errorDetails*)

Convert keypress string to list of HID codes

Parameters

keypressString String to convert

keypressesHID Reference to where the list of HID codes is placed

errorDetails Reference to where a possible error string is placed, if function returns false

Returns

true if converted ok, or false if there was an error in input

6.24.2.5 bool KeypressParser::keypressStringToQt (const QString *keypressString*, KeyObject & *keyObjectQt*, QString & *errorDetails*, bool *acceptRepresentation = true*)

Convert keypress string to Qt representation of key

Parameters

keypressString String to convert

keyObjectQt Reference to where the Qt representation is placed, as a [KeyObject](#)

errorDetails Reference to where a possible error string is placed, if function returns false

acceptRepresentation True if we accept both GUI and internal representation, false if only internal

Returns

true if converted ok, or false if there was an error in input

6.24.2.6 `bool KeypressParser::macroStringToHID (const QString macroString, QList< MyKeypressHID > & keypressesHID, QString & errorDetails)`

Convert macro string to list of HID codes

Parameters

macroString String to convert

keypressesHID Reference to where the list of HID codes is placed

errorDetails Reference to where a possible error string is placed, if function returns false

Returns

true if converted ok, or false if there was an error in input

6.24.2.7 `bool KeypressParser::macroStringToQt (const QString macroString, QList< KeyObject > & keyObjects, QString & errorDetails, bool acceptRepresentation = true)`

Convert macro string to list of Qt representations of keys

Parameters

macroString String to convert

keyObjects Reference to where the list of Qt representation is placed

errorDetails Reference to where a possible error string is placed, if function returns false

acceptRepresentation True if we accept both GUI and internal representation, false if only internal

Returns

true if converted ok, or false if there was an error in input

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

- [src/keypressparser.h](#)
- [src/keypressparser.cpp](#)

6.25 MacroFunctionsDialog Class Reference

Public Member Functions

- [MacroFunctionsDialog](#) (QString contents, QWidget *parent=0)
- [~MacroFunctionsDialog](#) ()

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.25.1 Constructor & Destructor Documentation

6.25.1.1 MacroFunctionsDialog::MacroFunctionsDialog (QString *contents*, QWidget * *parent* = 0)

Constructor

Parameters

- contents* HTML contents to display in widget (required)
- parent* Parent of this QObject (optional)

6.25.1.2 MacroFunctionsDialog::~~MacroFunctionsDialog ()

Destructor

6.25.2 Member Function Documentation

6.25.2.1 void MacroFunctionsDialog::changeEvent (QEvent * *e*) [protected]

Reimplementation of QObject's changeEvent

Parameters

- e* Event given by Qt

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

- [src/macrofuctionsdialog.h](#)
- [src/macrofuctionsdialog.cpp](#)

6.26 MainWindow Class Reference

```
#include <mainwindow.h>
```

Public Slots

- void **groupButtonPressed** (unsigned int button)
- void **applicationButtonPressed** (unsigned int button)
- void **addCategoryButtonClicked** (void)
- void **editCategoryButtonClicked** (void)
- void **addFunctionButtonClicked** (void)
- void **editFunctionButtonClicked** (void)
- void **assignButtonClicked** (void)
- void **unassignButtonClicked** (void)
- void **menuBarTriggered** (QAction *action)
- void **categoryActivated** (const QModelIndex &index)
- void **functionActivated** (const QModelIndex &index)

- void **mouseButtonActivated** (int mouseButton)
- void **checkboxToggled** (void)
- void **updateModeButtonClicked** (void)
- void **activateModeButtonClicked** (void)
- void **setInactive** (const objectType typeEnum, bool recursive=false)
- quint32 **getFlags** (void) const
- void **setCheckBoxes** (const quint32 flags)
- void **stopAssigning** (void)
- void **processExternalMessage** (const QString &message)
- void **profileEvent** (int event)
- void **showThisView** (QString groupName, QString applicationName, QPoint center)
- void **mapClosed** (void)
- void **activateProfile** (unsigned int profileNum, bool reload)
- void **updateStatistics** (void)
- QString **getActiveApplicationName** (void)
- void **quitModeware** ()

Signals

- void **setCategoryIndex** (const QModelIndex &index)
- void **setFunctionIndex** (const QModelIndex &index)
- void **setFunctionListFocus** ()
- void **setCategoryListFocus** ()
- void **flagsChanged** (quint32 flags)
- void **changeView** (QString groupName, QString applicationName, QPoint center)
- void **closeOtherView** ()
- void **refreshSimpleView** (QString groupName, QString applicationName)
- void **initialized** ()

Public Member Functions

- **MainWindow** (QWidget *parent=0)
- **~MainWindow** ()
- bool **isInitializedOK** (void)
- const QByteArray **getHeaderSvg** (void) const
- void **updateHeader** (bool groupNavigation, bool appNavigation)
- const QList< **GroupObject** > & **getGroupList** (void) const
- const int **getActiveGroupIndex** (void) const
- const int **getActiveApplicationIndex** (void) const

Protected Member Functions

- void **paintEvent** (QPaintEvent *)

6.26.1 Detailed Description

Qt main window

6.26.2 Constructor & Destructor Documentation

6.26.2.1 MainWindow::MainWindow (QWidget * *parent* = 0)

Constructor

Parameters

parent Parent of this QWidget (optional)

6.26.2.2 MainWindow::~MainWindow ()

Destructor

6.26.3 Member Function Documentation

6.26.3.1 void MainWindow::activateProfile (unsigned int *profileNum*, bool *reload*) [slot]

Activate mode with this profile number

Parameters

profileNum Profile number to activate

reload If setup contents may have been changed in simple screen, also reload setup

6.26.3.2 void MainWindow::changeView (QString *groupName*, QString *applicationName*, QPoint *center*) [signal]

Change to simple screen

Can also change simple screen's active group and application

Parameters

groupName Name of the active group, or empty if no change

applicationName Name of the active application, or empty if no change

center Coordinates of the center of shown window

6.26.3.3 void MainWindow::closeOtherView () [signal]

Close simple screen

6.26.3.4 void MainWindow::flagsChanged (quint32 *flags*) [signal]

Set active mouse widget flags

Currently only double-click is supported. Another mouse hardware might have toggle buttons that would be modelled as flags.

Parameters

flags Flags to set, individual bits of the number represent the flags.

6.26.3.5 `const int MainWindow::getActiveApplicationIndex (void) const`

Gets active application index of [MainWindow](#)

Returns

The application index

6.26.3.6 `QString MainWindow::getActiveApplicationName (void) [slot]`

Get active application's name

6.26.3.7 `const int MainWindow::getActiveGroupIndex (void) const`

Gets active group index of [MainWindow](#)

Returns

The group index

6.26.3.8 `quint32 MainWindow::getFlags (void) const [slot]`

Get active mouse widget flags.

Currently, only double-click is a supported flag.

Returns

A number, the bits of which represent the flags.

6.26.3.9 `const QList< GroupObject > & MainWindow::getGroupList (void) const`

Gets constant reference to the list of group objects owned by [MainWindow](#)

Returns

Reference to the list

6.26.3.10 `const QByteArray MainWindow::getHeaderSvg (void) const`

Gets header svg contents as byte array

Returns

The svg contents

6.26.3.11 `void MainWindow::initialized () [signal]`

Tell simple view that [MainWindow](#) is initialized, so that it may start to initialize itself

6.26.3.12 bool MainWindow::isInitializedOK (void)

Returns whether or not the program been successfully initialized

Returns

true if initialized successfully, or false if not

6.26.3.13 void MainWindow::mapClosed (void) [slot]

User requests the mode map to be closed

6.26.3.14 void MainWindow::paintEvent (QPaintEvent *) [protected]

Reimplementation of QWidget's paint event

6.26.3.15 void MainWindow::processExternalMessage (const QString & message) [slot]

Process a message that comes through the domain socket.

Currently, only the wake-up message that the single instance library sends, is supported.

Parameters

message Command that arrived in the domain socket

6.26.3.16 void MainWindow::profileEvent (int event) [slot]

Something regarding active profile happened in mouse hardware

Positive number means a profile was chosen with mouse buttons, and the number is the number of the chosen profile. Negative numbers are special events: CHOICE_EVENT, DISCONNECTED_EVENT and MAP_EVENT, for user entering profile choosing mode, for mouse being disconnected and for user requesting mode map with a mouse button.

Parameters

event Event that happened in mouse hardware

6.26.3.17 void MainWindow::quitModeware () [slot]

Quit Modeware

6.26.3.18 void MainWindow::refreshSimpleView (QString groupName, QString applicationName) [signal]

Assuming we are currently showing simple view, change its active application

Parameters

groupName The application's group

applicationName The application's name

6.26.3.19 void MainWindow::setCategoryIndex (const QModelIndex & *index*) [signal]

Set category with given index as active

Parameters

index Index to set active

6.26.3.20 void MainWindow::setCategoryListFocus () [signal]

Set keyboard focus to category list

6.26.3.21 void MainWindow::setCheckBoxes (const quint32 *flags*) [slot]

Set mouse widget checkboxes according to given flags

Currently we only have double-click checkbox

Parameters

flags Number, the bits of which represent the flags

6.26.3.22 void MainWindow::setFunctionIndex (const QModelIndex & *index*) [signal]

Set function with given index as active

Parameters

index Index to set active

6.26.3.23 void MainWindow::setFunctionListFocus () [signal]

Set keyboard focus to function list

6.26.3.24 void MainWindow::setInactive (const objectType *typeEnum*, bool *recursive* = *false*) [slot]

Reset active index of group/application/category/function.

Parameters

typeEnum Which of the four options to set inactive

recursive Do we recursively reset everything below the chosen option? For example, group resets all.

6.26.3.25 void `MainWindow::showThisView` (`QString groupName`, `QString applicationName`, `QPoint center`) [`slot`]

Unhide [MainWindow](#)

Parameters

groupName Name of active group to change to, or empty string if no change

applicationName Name of active application to change to, or empty string if no change

center Coordinates of the center of shown window

6.26.3.26 void `MainWindow::stopAssigning` (`void`) [`slot`]

Stop assigning to a mouse button

6.26.3.27 void `MainWindow::updateHeader` (`bool groupNavigation`, `bool appNavigation`)

Updates header graphic to reflect the latest data

Parameters

groupNavigation True if include group navigation tab, false if not

appNavigation True if include application navigation tab, false if not

6.26.3.28 void `MainWindow::updateStatistics` (`void`) [`slot`]

Reload the statistics xml file.

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

- [src/mainwindow.h](#)
- [src/mainwindow.cpp](#)

6.27 MapWidget Class Reference

```
#include <mapwidget.h>
```

Signals

- void [closing](#) ()

Public Member Functions

- [MapWidget](#) (const `QPixmap &pixmap`, `QWidget *parent=0`)

6.27.1 Detailed Description

A class which models a widget that shows the mode map

Widget contents are created by the caller

6.27.2 Constructor & Destructor Documentation

6.27.2.1 `MapWidget::MapWidget (const QPixmap & pixmap, QWidget * parent = 0)`

Constructor

Parameters

pixmap Bitmap containing the widget contents (required)

parent Parent of this QObject (optional)

6.27.3 Member Function Documentation

6.27.3.1 `void MapWidget::closing () [signal]`

Widget is closing

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

- [src/mapwidget.h](#)
- [src/mapwidget.cpp](#)

6.28 mouseButtonAction Struct Reference

Public Attributes

- `const char *` **internalString**
- `const char *` **GUIText**
- `quint8` **code**

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

- [src/setup_common.h](#)

6.29 MouseButtonObject Class Reference

```
#include <mousebuttonobject.h>
```

Public Member Functions

- [MouseButtonObject](#) (QString name)
- [MouseButtonObject](#) (const [MouseButtonObject](#) &source)

- QString [getName](#) (void) const
- QList< [ButtonBindingObject](#) > [getButtonBindings](#) (void) const
- bool [isBoundToThis](#) (const [ButtonBindingObject](#) &binding) const
- bool [isBoundToFunction](#) (const [FunctionObject](#) &function, quint32 &flags) const
- bool [isBoundToFunctionWithFlags](#) (const [FunctionObject](#) &function, quint32 flags) const
- bool [isBoundInApplication](#) (const QString applicationName) const
- bool [isOnlyPartiallyFreeInApplication](#) (const QString applicationName) const
- bool [getFunctionWithFlags](#) ([FunctionObject](#) &function, const QString applicationName, quint32 flags) const
- bool [bindButton](#) ([ButtonBindingObject](#) &binding)
- bool [unbindButton](#) (const [ButtonBindingObject](#) &binding)
- void [unbindAll](#) (void)
- [MouseButtonObject](#) & [operator=](#) (const [MouseButtonObject](#) &right)

6.29.1 Detailed Description

A class which models a mouse button

6.29.2 Constructor & Destructor Documentation

6.29.2.1 MouseButtonObject::MouseButtonObject (QString *name*)

Constructor

Parameters

name Button's name (required)

6.29.2.2 MouseButtonObject::MouseButtonObject (const MouseButtonObject & *source*)

Copy constructor, performs deep copy

Parameters

source [MouseButtonObject](#) to be copied

6.29.3 Member Function Documentation

6.29.3.1 bool MouseButtonObject::bindButton (ButtonBindingObject & *binding*)

Sets one more binding

Parameters

binding What to bind

Returns

true if bound successfully, or false if there was an error

6.29.3.2 `QList< ButtonBindingObject > MouseButtonObject::getButtonBindings (void) const`

Get's list of button's bindings in all the different applications

Returns

List of bindings

6.29.3.3 `bool MouseButtonObject::getFunctionWithFlags (FunctionObject & function, const QString applicationName, quint32 flags) const`

If there is a function bound in given application, with given flags, get the function.

Parameters

function Reference to where the function is placed if it is found

applicationName Application, the functions of which we are checking.

flags The flags that have to be associated with the function, if it is to be found.

Returns

true if function is found, or false if it is not found.

6.29.3.4 `QString MouseButtonObject::getName (void) const`

Getter for name

Returns

Button's name

6.29.3.5 `bool MouseButtonObject::isBoundInApplication (const QString applicationName) const`

Is this button bound to any function in given application?

Parameters

applicationName Name of the application to check

Returns

true if button is bound to something in given application, or false if is not bound

6.29.3.6 `bool MouseButtonObject::isBoundToFunction (const FunctionObject & function, quint32 & flags) const`

Is this function among the bindings?

Also shows what flags the function is associated with. Only double-click flag is currently supported.

Parameters

function The function to seek in the list of bindings
flags Reference to where the associated flags are written

Returns

true if is bound to given function, or false if is not bound

6.29.3.7 bool MouseButtonObject::isBoundToFunctionWithFlags (const FunctionObject & function, quint32 flags) const

Is this function, with the given flags, among the bindings?
Only double-click flag is currently supported.

Parameters

function The function to seek in the list of bindings
flags The flags that have to be associated with the function, if it is to be found.

Returns

true if is bound to given function with given flags, or false if is not bound

6.29.3.8 bool MouseButtonObject::isBoundToThis (const ButtonBindingObject & binding) const

Does the given binding exist in the list of bindings?

Parameters

binding The binding to seek in the list

Returns

true if exists, or false if doesn't exist

6.29.3.9 bool MouseButtonObject::isOnlyPartiallyFreeInApplication (const QString applicationName) const

Is this button bound in given application with some, but not all, supported flag combinations?
Only double-click flag is currently supported, giving two possible combinations.

Parameters

applicationName Name of the application to check

Returns

true if button is partially bound to something in given application, or false if is not bound at all, or is fully bound

6.29.3.10 MouseButtonObject & MouseButtonObject::operator= (const MouseButtonObject & *right*)

Operator =, performs deep copy

Parameters

right Right side of assignment

Returns

[MouseButtonObject](#) reference

6.29.3.11 void MouseButtonObject::unbindAll (void)

Remove all bindings from button

6.29.3.12 bool MouseButtonObject::unbindButton (const ButtonBindingObject & *binding*)

If there is a binding that matches the given binding, unbind it

Parameters

binding The binding to check against

Returns

true if binding was found and unbound, or false if there was no such binding

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

- [src/mousebuttonobject.h](#)
- [src/mousebuttonobject.cpp](#)

6.30 mouseButtonRepresentation Struct Reference

Public Attributes

- QString **buttonString**
- QString **buttonGUIString**
- mouseButtons **buttonEnum**

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

- [src/mouseobject.h](#)

6.31 mouseButtonWithFlags Struct Reference

Public Attributes

- mouseButtons **mouseButton**
- quint32 **flags**

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

- [src/mouseobject.h](#)

6.32 MouseHardware Class Reference

```
#include <mousehardware.h>
```

Public Member Functions

- void [quit](#) ()
- bool [updateProfileCopy](#) ()
- int [checkLibrary](#) ()
- void [closeLibrary](#) ()
- int [checkMouseState](#) ()
- bool [writeMouse](#) ()
- bool [readMouse](#) ()
- bool [getRevision](#) (quint8 &major, quint8 &middle, quint8 &minor)
- bool [getActiveProfile](#) (quint8 &profile)
- bool [eraseFlash](#) ()
- bool [eraseBlock](#) (quint8 block4k)
- bool [writeProfileInfoBlock](#) (const [PROFILE_INFO_BLOCK](#) &profile)
- bool [setProfile](#) (quint8 profileNumber)
- bool [enableFirmwareUpdate](#) (void)
- bool [getProfileCopy](#) (quint8 profileNumber, [PROFILE_INFO_BLOCK](#) &profile) const
- bool [setProfileCopy](#) (quint8 profileNumber, const [PROFILE_INFO_BLOCK](#) profile)
- bool [writeProfileKeyMappingBlock](#) (quint8 addr, unsigned int size, quint8 *data)
- bool [readProfileInfoBlock](#) (quint8 profileNumber, quint8 *profileBlock)
- void [getFreeBlocks](#) (bool freeArray[], unsigned int skipProfile=0) const
- int [pollProfileState](#) (bool updateStatistics, quint8 *singleClickData=0, quint8 *doubleClickData=0, quint8 *unprogrammableData=0)
- bool [getBlockChecksum](#) (quint8 addr, quint32 &checksum)
- quint32 [calculateChecksum](#) (quint8 *data, unsigned int length, quint32 checksum) const
- bool [getJoyCoords](#) (quint8 &X, quint8 &Y)
- bool [isInitializedOK](#) (void)

Static Public Member Functions

- static [MouseHardware](#) & [instance](#) ()

Public Attributes

- quint8 **reportOut_** [9]
- quint8 **reportIn_** [9]
- quint8 **revision_** [3]

6.32.1 Detailed Description

A class which is responsible for using AtUsbHid DLL for communicating with the mouse

Conforms to the Singleton design pattern

6.32.2 Member Function Documentation

6.32.2.1 quint32 MouseHardware::calculateChecksum (quint8 * *data*, unsigned int *length*, quint32 *checksum*) const

Calculate the checksum of given data

Does not communicate with mouse.

Parameters

data The data for which we calculate checksum

length Length of data

checksum Initial value for calculation. Usually 0.

Returns

The calculated checksum

6.32.2.2 int MouseHardware::checkLibrary ()

Check that the Atmel library was correctly loaded

Returns

WM_OK if loaded ok, WM_BAD_PATH if library path was wrong, WM_DLL_ERROR if DLL returned error, or WM_ERROR if other error happened

6.32.2.3 int MouseHardware::checkMouseState ()

Check mouse's state

Returns

WM_OK if mouse is ok, WM_COULD_NOT_OPEN_DEVICE if mouse not found, WM_NO_USB_DEVICE_CAPABILITIES if USB not supported, or WM_ERROR if there is something else wrong with mouse

6.32.2.4 void MouseHardware::closeLibrary ()

Close the Atmel library

6.32.2.5 bool MouseHardware::enableFirmwareUpdate (void)

Set mouse in firmware update mode

Mouse will become unresponsive when this is done

Returns

true if firmware update mode set successfully, or false if there was an error

6.32.2.6 bool MouseHardware::eraseBlock (quint8 *block4k*)

Erase a specific 4k block in flash

Parameters

block4k Block to erase, starting from 0

Returns

true if erased successfully, or false if there was an error

6.32.2.7 bool MouseHardware::eraseFlash ()

Erase all profiles from mouse

Returns

true if erased successfully, or false if there was an error

6.32.2.8 bool MouseHardware::getActiveProfile (quint8 & *profile*)

Get the currently active profile number in the mouse

Parameters

profile Reference to where profile number is placed

Returns

true if received number successfully, or false if there was an error

6.32.2.9 bool MouseHardware::getBlockChecksum (quint8 addr, quint32 & checksum)

Get the checksum of a block, calculated by the mouse

Parameters

addr Which block to get, 0 - 127
checksum Reference to where the checksum is placed

Returns

true if received checksum successfully, or false if there was an error

6.32.2.10 void MouseHardware::getFreeBlocks (bool freeArray[], unsigned int skipProfile = 0) const

Finds out which blocks are free, based on local copy

Does not communicate with mouse. It is possible to skip a specific profile in checking, so that it is marked as free. In this parameter, first profile number is 1, so that 0 is left for no skipping.

Parameters

freeArray Array of booleans, where true means free and false means used. There must be space for 128 booleans.
skipProfile If nonzero, skip the profile with this number. Profiles start from 1.

6.32.2.11 bool MouseHardware::getJoyCoords (quint8 & X, quint8 & Y)

Get the current, raw coordinates of mouse's joystick

Parameters

X Reference to where X-coordinate is placed
Y Reference to where Y-coordinate is placed

Returns

true if received coordinates successfully, or false if there was an error

6.32.2.12 bool MouseHardware::getProfileCopy (quint8 profileNumber, PROFILE_INFO_BLOCK & profile) const

Get a specific profile from our local copy of profiles

Does not communicate with the mouse. In the HW API, profile numbers start from 0.

Parameters

profileNumber The profile number to get, 0 being the first profile
profile Reference to where the profile contents are copied to

Returns

true if profile received successfully, or false if there was an error

6.32.2.13 `bool MouseHardware::getRevision (quint8 & major, quint8 & middle, quint8 & minor)`

Get firmware revision, for example 1.0.0

Parameters

major Reference to where the major number of version is placed
middle Reference to where the middle number of version is placed
minor Reference to where the minor number of version is placed

Returns

true if received version successfully, or false if there was an error

6.32.2.14 `static MouseHardware& MouseHardware::instance () [inline, static]`

Gives a reference to the Singleton object

Returns

reference to the object

The Singleton instance

6.32.2.15 `bool MouseHardware::isInitializedOK (void)`

Returns whether or not the program been successfully initialized

Returns

true if initialized successfully, or false if not

6.32.2.16 `int MouseHardware::pollProfileState (bool updateStatistics, quint8 * singleClickData = 0, quint8 * doubleClickData = 0, quint8 * unprogrammableData = 0)`

Polls the profile state from mouse.

User might have, for example, entered into profile choosing mode with mouse buttons, or chosen another profile with the mouse.

Parameters

updateStatistics In this poll, also get statistics from mouse.
singleClickData If pointer not null, place single click statistics here.
doubleClickData If pointer not null, place double-click statistics here.
unprogrammableData If pointer not null, place statistics regarding unprogrammable buttons (L1 etc.) here.

Returns

-1 if unable to communicate with mouse, 128 if we had to skip this poll because of other communications, or otherwise whatever the mouse responded to the poll

6.32.2.17 void MouseHardware::quit ()

Close hardware library and set mouse as disconnected

6.32.2.18 bool MouseHardware::readMouse ()

Read what the mouse is sending and place it in reportIn_

Returns

true if read successfully, or false if there was an error

6.32.2.19 bool MouseHardware::readProfileInfoBlock (quint8 *profileNumber*, quint8 **profileBlock*)

Gets a specific profile info block from mouse

In the HW API, profile numbers start from 0.

Parameters

profileNumber The profile number to get, 0 being the first profile

profileBlock Where to copy the block contents

Returns

true if successfully received contents, or false if there was an error

6.32.2.20 bool MouseHardware::setProfile (quint8 *profileNumber*)

Set given profile number as active

In the HW API, profile numbers start from 0

Parameters

profileNumber The profile number to activate, 0 being the first profile

Returns

true if profile set successfully, or false if there was an error

6.32.2.21 bool MouseHardware::setProfileCopy (quint8 *profileNumber*, const PROFILE_INFO_BLOCK *profile*)

Sets a specific profile in our local copy of profiles, to the given contents.

Does not communicate with the mouse. In the HW API, profile numbers start from 0.

Parameters

profileNumber The profile number to set, 0 being the first profile

profile New profile contents

Returns

true if profile copied successfully, or false if there was an error

6.32.2.22 bool MouseHardware::updateProfileCopy ()

Get the profile info blocks from mouse and store them in the local array

Returns

true if successfully received profile infos, or false if there was an error

6.32.2.23 bool MouseHardware::writeMouse ()

Write to mouse whatever is in reportOut_.

Returns

true if written successfully, or false if there was an error

6.32.2.24 bool MouseHardware::writeProfileInfoBlock (const PROFILE_INFO_BLOCK & *profile*)

Write the given profile info block into mouse

Parameters

profile The block to write

Returns

true if block written successfully, or false if there was an error

6.32.2.25 bool MouseHardware::writeProfileKeyMappingBlock (quint8 *addr*, unsigned int *size*, quint8 * *data*)

Write given data to given key mapping block

Parameters

addr Which block to write, 0 - 127?

size Length of data

data Data to write

Returns

true if written successfully, or false if there was an error

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

- [src/mousehardware.h](#)
- [src/mousehardware.cpp](#)

6.33 MouseObject Class Reference

```
#include <mouseobject.h>
```

Public Slots

- void [setFlags](#) (const quint32 flags)
- void [refreshWidget](#) (void)

Signals

- void [buttonActivated](#) (int)
- void [flagsChanged](#) (quint32 flags)
- void [refreshWidgetSignal](#) (void)
- void [profileEvent](#) (int event)
- void [statisticsEvent](#) (void)

Public Member Functions

- void [quit](#) ()
- bool [getButtonString](#) (mouseButtons buttonEnum, QString &buttonString, bool GUIText=false)
- bool [getButtonEnum](#) (QString buttonString, mouseButtons &button)
- bool [getMouseButtonPointer](#) (mouseButtons buttonEnum, [MouseButtonObject](#) **ppMouseButton)
- bool [setButtonToFunction](#) (const mouseButtons buttonEnum, const [FunctionObject](#) function, const quint32 flags=0)
- bool [getActiveButton](#) (mouseButtons &buttonEnum) const
- bool [setActiveButton](#) (int mouseButton)
- quint32 [getFlags](#) (void) const
- void [setMouseWidget](#) ([MouseWidget](#) *pMouseWidget)
- cmdResult [getFirmwareRevision](#) (QString &revision)
- mouseState [getMouseState](#) (void) const
- void [setMouseState](#) (mouseState newState)
- bool [isMouseConnected](#) (void) const
- bool [compareProfileInfoBlocks](#) (const [PROFILE_INFO_BLOCK](#) &block1, const [PROFILE_INFO_BLOCK](#) &block2, bool ignoreMappingAddresses) const
- bool [createInfoBlockContents](#) (const [ApplicationObject](#) &application, [PROFILE_INFO_BLOCK](#) *pProfileInfo)
- unsigned int [createMappingBlockContents](#) (const [ApplicationObject](#) &application, quint8 dataBlocks[][4096])
- quint64 [getMappingBlockDifferences](#) (const QList< [GroupObject](#) > &groups)
- bool [configureButtons](#) (const QList< [ApplicationObject](#) > applications)
- bool [enableFirmwareUpdate](#) (void)
- bool [initHardware](#) (void)
- bool [setActiveProfile](#) (unsigned int profileNumber, bool forceNumber=false)
- void [unbindAll](#) ()
- bool [eraseAll](#) ()
- cmdResult [getJoyCoords](#) (quint8 &X, quint8 &Y)
- quint8 [getActiveProfileNumber](#) (void) const

- bool `getFunctionBindings` (`FunctionObject` function, `QList< struct mouseButtonWithFlags >` &bindings) const
- bool `isProfileSynced` (`quint8` profileNumber) const
- bool `setProfileSynced` (`quint8` profileNumber, bool synced)
- void `getAndResetStatistics` (`unsigned int` *singleClickData, `unsigned int` *doubleClickData, `unsigned int` *unprogrammableData)
- void `getFreeButtons` (`const QString` applicationName, `QList< mouseButtons >` &freeButtons) const
- void `getPartiallyFreeButtons` (`const QString` applicationName, `QList< mouseButtons >` &freeButtons) const
- bool `getFreeButtonDisplay` (`void`) const
- void `setFreeButtonDisplay` (bool displayFreeButtons)
- `QString` `getActiveApplicationName` (`void`)
- void `setMainWindowPtr` (`QObject` *ptr)

Static Public Member Functions

- static `MouseObject` & `instance` ()

6.33.1 Detailed Description

A class which models the mouse

Conforms to the Singleton design pattern

6.33.2 Member Function Documentation

6.33.2.1 void MouseObject::buttonActivated (int) [signal]

A mouse button was activated with mouse widget

6.33.2.2 bool MouseObject::compareProfileInfoBlocks (const PROFILE_INFO_BLOCK & block1, const PROFILE_INFO_BLOCK & block2, bool ignoreMappingAddresses) const

Compares two profile info blocks to see if they have same contents

Parameters

block1 The first block to compare

block2 The second block to compare

ignoreMappingAddresses True if we ignore address fields, false if we take them into account

Returns

true if contents are different, or false if they are the same

6.33.2.3 bool MouseObject::configureButtons (const QList< ApplicationObject > *applications*)

Set mouse flash contents based on a list of application objects

Parameters

applications List of modes to change in the mouse flash, to what is in application objects

Returns

true if successfully wrote mouse flash contents, or false if there was an error

6.33.2.4 bool MouseObject::createInfoBlockContents (const ApplicationObject & *application*, PROFILE_INFO_BLOCK * *pProfileInfo*)

Creates profile info block contents based on given application object

Parameters

application The application, based on which we create contents

pProfileInfo Pointer to where the contents are placed. Caller is responsible for memory allocation.

Returns

true if contents were created successfully, or false if there was something wrong with application object

6.33.2.5 unsigned int MouseObject::createMappingBlockContents (const ApplicationObject & *application*, quint8 *dataBlocks*[[4096]])

Creates key mapping block contents based on given application object

Parameters

application The application, based on which we create contents

dataBlocks Two dimensional array of four 4K blocks of data, where we place contents

Returns

true if contents were created successfully, or false if there was something wrong with application object

6.33.2.6 bool MouseObject::enableFirmwareUpdate (void)

Set the mouse in firmware update mode

The mouse will become unresponsive.

Returns

true if set successfully, or false if there was an error

6.33.2.7 bool MouseObject::eraseAll ()

Erase everything in mouse flash

Returns

true if erased successfully, or false if there was an error

6.33.2.8 void MouseObject::flagsChanged (quint32 *flags*) [signal]

User changed mouse flags with checkboxes

Currently, only double-click flag is supported.

Parameters

flags New flags to be set.

6.33.2.9 QString MouseObject::getActiveApplicationName (void)

Gets the name of the currently active application in the mouse.

This might be different than what the software thinks is active.

Returns

The name of the active application

6.33.2.10 bool MouseObject::getActiveButton (mouseButtons & *buttonEnum*) const

Which button is currently active, as represented by glow around it in the mouse widget

Parameters

buttonEnum Reference to where the enum-representation of the active button is placed

Returns

true if there was an active button, or false if no button was active

6.33.2.11 quint8 MouseObject::getActiveProfileNumber (void) const

Get the currently active profile number in the mouse, or 0 if no profile is active

We get 0 if mouse is disconnected

Returns

The active profile number, or 0 if failed to get number

6.33.2.12 void MouseObject::getAndResetStatistics (unsigned int * *singleClickData*, unsigned int * *doubleClickData*, unsigned int * *unprogrammableData*)

Gets and resets the local copy of mouse button statistics

Does not communicate with mouse. The numbers mean number of clicks, and the indices represent the different buttons.

Parameters

singleClickData Pointer to an array where we place single click data. Must have NUM_BUTTONS indices.

doubleClickData Pointer to an array where we place double-click data. Must have NUM_DOUBLECLICK_BUTTONS indices.

unprogrammableData Pointer to an array where we place data for unprogrammable buttons. Must have NUM_UNPROGRAMMABLE_BUTTONS indices.

6.33.2.13 bool MouseObject::getButtonEnum (QString *buttonString*, mouseButtons & *button*)

Get mouseButtons enum representation of a mouse button based on internal string representation

Parameters

buttonString Internal string representation of button we are interested in

button Reference to where the enum-representation is placed

Returns

true if successfully converted, or false if there is no such button

6.33.2.14 bool MouseObject::getButtonString (mouseButtons *buttonEnum*, QString & *buttonString*, bool *GUIText* = *false*)

Get string representation of a mouse button based on mouseButtons enum

Parameters

buttonEnum The enum-representation of the button that we are interested in

buttonString Reference to where the string representation is placed

GUIText False if we need internal representation, true if human-readable

Returns

true if successfully converted, or false if there is no such button

6.33.2.15 cmdResult MouseObject::getFirmwareRevision (QString & *revision*)

Get firmware version from the mouse

Parameters

revision Reference to where the firmware version is placed as string

Returns

CMD_OK if successfully received version, CMD_DISCONNECTED if mouse is currently disconnected, or CMD_ERROR if there was a communication error

6.33.2.16 quint32 MouseObject::getFlags (void) const

Get current mouse flags.

Only double-click flag is currently supported.

Returns

A number, the bits of which represent the flags

6.33.2.17 bool MouseObject::getFreeButtonDisplay (void) const

Finds out if the mouse widget is currently displaying the unbound and partially bound buttons with different colors

Returns

true if is displaying, or false if is not displaying

6.33.2.18 void MouseObject::getFreeButtons (const QString *applicationName*, QList< mouseButtons > & *freeButtons*) const

Gets a list of buttons for given application that are completely unbound

Parameters

applicationName Name of application for which to get the list

freeButtons Reference to list where the unassigned buttons are placed

6.33.2.19 bool MouseObject::getFunctionBindings (FunctionObject *function*, QList< struct mouseButtonWithFlags > & *bindings*) const

Get all button bindings to given function

Parameters

function The function, the bindings to which we want

bindings Reference to where the list of bindings is placed

Returns

true if there were some bindings, or false if there were none

6.33.2.20 `cmdResult MouseObject::getJoyCoords (quint8 & X, quint8 & Y)`

Get the current, raw coordinates of mouse's joystick

Parameters

X Reference to where X-coordinate is placed

Y Reference to where Y-coordinate is placed

Returns

CMD_OK if received coordinates successfully, CMD_DISCONNECTED if mouse is currently disconnected, or CMD_ERROR if there was a communication error

6.33.2.21 `quint64 MouseObject::getMappingBlockDifferences (const QList< GroupObject > & groups)`

Find out what profiles are different in mouse than in current setup xml

Parameters

groups The current group list, based on setup xml file

Returns

64 bit number, the bits of which represent profile differences. 1 for difference, 0 for no difference.

6.33.2.22 `bool MouseObject::getMouseButtonPointer (mouseButtons buttonEnum, MouseButtonObject ** ppMouseButton)`

Get pointer's pointer to a mouse button object based on its mouseButtons enum representation

Parameters

buttonEnum The enum-representation of the button that we are interested in

ppMouseButton Pointer behind this pointer will point to the mouse button object if successful

Returns

true if mouse button found, or false if there is no such button

6.33.2.23 `mouseState MouseObject::getMouseState (void) const`

Get the current state of mouse

Returns

MOUSE_OBJECT_NOT_INITIALIZED if mouse object was not initialized yet, MOUSE_HARDWARE_NOT_FOUND if the mouse cannot be found, MOUSE_HARDWARE_UNINITIALIZED if the mouse is found but uninitialized, or MOUSE_READY if the mouse is operative

6.33.2.24 void MouseObject::getPartiallyFreeButtons (const QString *applicationName*, QList< mouseButtons > & *freeButtons*) const

Gets a list of buttons for given application that are only partially bound.

Partially means that only some of the possible combinations of flags are used. Currently, only double-click flag is used, giving two combinations.

Parameters

applicationName Name of application for which to get the list

freeButtons Reference to list where the only partially assigned buttons are placed

6.33.2.25 bool MouseObject::initHardware (void)

Initialize mouse hardware

Returns

true if initialized successfully, or false if there was an error

6.33.2.26 static MouseObject& MouseObject::instance () [inline, static]

Gives a reference to the Singleton object

Returns

reference to the object

The Singleton instance

6.33.2.27 bool MouseObject::isMouseConnected (void) const

Checks if the mouse is currently connected and operative

Returns

true if mouse is operative, or false if mouse not found, or is not initialized

6.33.2.28 bool MouseObject::isProfileSynced (quint8 *profileNumber*) const

Checks if the mouse has the same contents for this profile number as the setup xml file based on local information

Does not communicate with mouse, but assumes that the local information has been updated when mouse was plugged in.

Parameters

profileNumber The profile number to check, starting from 1

Returns

true if the mouse has current contents for profile, or false if the contents are not current

6.33.2.29 void MouseObject::profileEvent (int *event*) [signal]

Something happened in the mouse regarding profile

CHOICE_EVENT means that user entered to profile choosing mode, DISCONNECTED_EVENT means that mouse was disconnected, MAP_EVENT means that user pressed mouse button to display or close mode map, and 0 or higher means that user changed profile with mouse buttons. In this case, the profile numbers start from 0, not 1, so subtract one from the profile number before sending the signal.

Parameters

event The event that occurred

6.33.2.30 void MouseObject::quit ()

Command hardware interface to shut down

6.33.2.31 void MouseObject::refreshWidget (void) [slot]

Forces repaint of mouse widget

6.33.2.32 void MouseObject::refreshWidgetSignal (void) [signal]

Forces repaint of mouse widget

6.33.2.33 bool MouseObject::setActiveButton (int *mouseButton*)

Set specific button as active, as represented by glow around it in the mouse widget

Parameters

mouseButton The enum-representation of the button casted as int, or -1 if set all inactive.

Returns

true if we set successfully, or false if parameter had illegal value

6.33.2.34 bool MouseObject::setActiveProfile (unsigned int *profileNumber*, bool *forceNumber* = *false*)

Set the active profile in the mouse to this new profile number

Profile numbers start from 1.

Parameters

profileNumber The profile number to set active

forceNumber Set the number to the mouse even if it appears to be active, based on RAM contents

Returns

true if profile set successfully, or false if there was an error

6.33.2.35 `bool MouseObject::setButtonToFunction (const mouseButtons buttonEnum, const FunctionObject function, const quint32 flags = 0)`

Bind a specific mouse button to given function

Parameters

buttonEnum The enum-representation of the button that we are interested in

function The function we are binding to

flags Flags for the binding. Only double-click is currently supported.

Returns

true if bound successfully, or false if unable to bind

6.33.2.36 `void MouseObject::setFlags (const quint32 flags) [slot]`

Sets the current mouse flags.

Currently, only double-click flag is supported.

Parameters

flags New flags to be set

6.33.2.37 `void MouseObject::setFreeButtonDisplay (bool displayFreeButtons)`

Sets the mouse widget as displaying, or not displaying, the unbound and partially bound buttons with different colors

Parameters

displayFreeButtons True for display, false for not display

6.33.2.38 `void MouseObject::setMainWindowPtr (QObject * ptr)`

[MouseObject](#) must have a pointer to [MainWindow](#), and this sets the pointer.

Parameters

ptr Pointer to [MainWindow](#)

6.33.2.39 `void MouseObject::setMouseState (mouseState newState)`

Set the current mouse state

Parameters

newState The state to set

6.33.2.40 void MouseObject::setMouseWidget (MouseWidget * *pMouseWidget*)

[MouseObject](#) owns the mouse widget, and we set the widget with this function

Parameters

pMouseWidget Pointer to the mouse widget

6.33.2.41 bool MouseObject::setProfileSynced (quint8 *profileNumber*, bool *synced*)

Set the local information to have the given profile number as having current, or not current, profile information

Does not communicate with mouse, only changes local data

Parameters

profileNumber The profile number to set current, starting from 1

synced True to set information current, or false to set it not current

Returns

true if set successfully, or false if there was something wrong with given parameters

6.33.2.42 void MouseObject::statisticsEvent (void) [signal]

Mouse just sent statistics that had some clicks.

If all buttons have zero clicks, this event is not emitted.

6.33.2.43 void MouseObject::unbindAll (void)

Unbind all mouse buttons

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

- [src/mouseobject.h](#)
- [src/mouseobject.cpp](#)

6.34 MouseWidget Class Reference

```
#include <mousewidget.h>
```

Public Slots

- void [refresh](#) (void)

Public Member Functions

- [QWidget](#) (QWidget *parent)
- bool [getFreeButtonDisplay](#) (void)
- void [setFreeButtonDisplay](#) (bool displayFreeButtons)

Protected Member Functions

- void [mousePressEvent](#) (QMouseEvent *event)
- void [paintEvent](#) (QPaintEvent *e)

6.34.1 Detailed Description

A class which models a widget for showing the mouse graphic in advanced view

6.34.2 Constructor & Destructor Documentation

6.34.2.1 QWidget::QWidget (QWidget * *parent*)

Constructor

Parameters

parent Parent of this QObject (required)

6.34.3 Member Function Documentation

6.34.3.1 bool QWidget::getFreeButtonDisplay (void)

Finds out if the mouse widget is currently displaying the unbound and partially bound buttons with different colors

Returns

true if is displaying, or false if is not displaying

6.34.3.2 void QWidget::mousePressEvent (QMouseEvent * *event*) [protected]

Reimplementation of QWidget's mousePressEvent

Parameters

event The mouse press event

6.34.3.3 void QWidget::paintEvent (QPaintEvent * *e*) [protected]

Reimplementation of QWidget's paint event

Parameters

e The paint event

6.34.3.4 void MouseWidget::refresh (void) [slot]

Forces repaint of mouse widget

6.34.3.5 void MouseWidget::setFreeButtonDisplay (bool *displayFreeButtons*)

Sets the mouse widget as displaying, or not displaying, the unbound and partially bound buttons with different colors

Parameters

displayFreeButtons True for display, false for not display

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

- [src/mousewidget.h](#)
- [src/mousewidget.cpp](#)

6.35 MyAllStatisticsTable Class Reference

```
#include <myallstatisticstable.h>
```

Public Member Functions

- [MyAllStatisticsTable](#) (QWidget *parent)
- void [recalculateColumnSizes](#) ()
- void [setShortcuts](#) (void)

Protected Member Functions

- void [paintEvent](#) (QPaintEvent *e)
- void [contextMenuEvent](#) (QContextMenuEvent *event)

6.35.1 Detailed Description

A class which models a table view for displaying statistics for all modes

Table contents are created by the caller

6.35.2 Constructor & Destructor Documentation

6.35.2.1 MyAllStatisticsTable::MyAllStatisticsTable (QWidget * *parent*)

Constructor

Parameters

parent Parent of this QObject (required)

6.35.3 Member Function Documentation

6.35.3.1 void MyAllStatisticsTable::contextMenuEvent (QContextMenuEvent * *event*) [protected]

Reimplementation of QWidget's contextMenuEvent

Parameters

event The context menu event

6.35.3.2 void MyAllStatisticsTable::paintEvent (QPaintEvent * *e*) [protected]

Reimplementation of QWidget's paintEvent

Parameters

e The paint event

6.35.3.3 void MyAllStatisticsTable::recalculateColumnSizes ()

Recalculate column sizes, because something has changed

6.35.3.4 void MyAllStatisticsTable::setShortcuts (void)

Set keyboard shortcuts.

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

- [src/myallstatisticstable.h](#)
- [src/myallstatisticstable.cpp](#)

6.36 MyKey Struct Reference

Public Attributes

- int **key**
- bool **right**
- bool **numPad**

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

- [src/keypress_common.h](#)

6.37 myKey Struct Reference

Public Attributes

- int **key**
- bool **right**

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

- [src/editfunctiondialog.cpp](#)

6.38 MyKeyPressHID Struct Reference

Public Attributes

- quint8 **keyHID**
- quint8 **modifiersHID**
- unsigned int **delayMs1**
- unsigned int **delayMs2**

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

- [src/keypress_common.h](#)

6.39 MyListView Class Reference

```
#include <mylistview.h>
```

Public Member Functions

- [MyListView](#) (QWidget *parent)

6.39.1 Detailed Description

A class which models a custom list view

Currently, the only difference to default list view is that items are activated with a single click

6.39.2 Constructor & Destructor Documentation

6.39.2.1 MyListView::MyListView (QWidget * *parent*)

Constructor

Parameters

parent Parent of this QObject (required)

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

- [src/mylistview.h](#)
- [src/mylistview.cpp](#)

6.40 MyStatisticsTable Class Reference

```
#include <mystaticstable.h>
```

Public Member Functions

- [MyStatisticsTable](#) (QWidget *parent)
- void [recalculateColumnSizes](#) ()
- void [setShortcuts](#) (void)

Protected Member Functions

- void [paintEvent](#) (QPaintEvent *e)
- void [contextMenuEvent](#) (QContextMenuEvent *event)

6.40.1 Detailed Description

A class which models a table view for displaying statistics for a specific mode
Table contents are created by the caller

6.40.2 Constructor & Destructor Documentation

6.40.2.1 MyStatisticsTable::MyStatisticsTable (QWidget * *parent*)

Constructor

Parameters

parent Parent of this QObject (required)

6.40.3 Member Function Documentation

6.40.3.1 void MyStatisticsTable::contextMenuEvent (QContextMenuEvent * *event*) [protected]

Reimplementation of QWidget's contextMenuEvent

Parameters

event The context menu event

6.40.3.2 void MyStatisticsTable::paintEvent (QPaintEvent * e) [protected]

Reimplementation of QWidget's paintEvent

Parameters

- e* The paint event

6.40.3.3 void MyStatisticsTable::recalculateColumnSizes ()

Recalculate column sizes, because something has changed

6.40.3.4 void MyStatisticsTable::setShortcuts (void)

Set keyboard shortcuts.

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

- [src/mystatisticstable.h](#)
- [src/mystatisticstable.cpp](#)

6.41 MyToolButton Class Reference

```
#include <mytoolbutton.h>
```

Public Member Functions

- [MyToolButton](#) (QWidget *parent=0, const MyToolButtonType type=MY_TOOLBUTTON_TYPE_NONE)
- void [setType](#) (MyToolButtonType type)

Protected Member Functions

- void [paintEvent](#) (QPaintEvent *e)

6.41.1 Detailed Description

A class which models a custom tool button for navigation tab

The button knows which control it represents, and chooses the right graphics, both for being enabled and for being disabled

6.41.2 Constructor & Destructor Documentation

6.41.2.1 MyToolButton::MyToolButton (QWidget * parent = 0, const MyToolButtonType type = MY_TOOLBUTTON_TYPE_NONE)

Constructor

Parameters

- parent* Parent of this QObject (optional)
- type* Type of this tool button

6.41.3 Member Function Documentation**6.41.3.1 void MyToolButton::paintEvent (QPaintEvent * e) [protected]**

Reimplementation of QWidget's paintEvent

Parameters

- e* The paint event

6.41.3.2 void MyToolButton::setType (MyToolButtonType type)

Set the type of this tool button

Parameters

- type* The new type for this tool button

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

- [src/mytoolbutton.h](#)
- [src/mytoolbutton.cpp](#)

6.42 png_stream_to_byte_array_closure_t Struct Reference**Public Attributes**

- unsigned char * **current_position**
- unsigned char * **end_of_array**

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

- [src/mainwindow.cpp](#)

6.43 PROFILE_INFO_BLOCK Struct Reference**Public Attributes**

- quint8 **profile_valid**
- quint8 **profile_number**
- quint8 **cpi_setting**
- quint8 **joystick_key**
- quint8 **addr_profile_block1**

- quint8 **addr_profile_block2**
- quint8 **addr_profile_block3**
- quint8 **addr_profile_block4**
- quint8 **J** [4][2]
- quint8 **cpi_low**
- quint8 **cpi_high**
- quint8 **unused** [84]

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

- [src/mousehardware.h](#)

6.44 SetupParser Class Reference

```
#include <setupparser.h>
```

Public Member Functions

- bool [readSetupFile](#) (QIODevice *device)
- bool [readBindingsFile](#) (QIODevice *device)
- bool [readStatisticsFile](#) (QIODevice *device)
- int [parseSetup](#) (QIODevice *deviceSetup, bool ignoreDuplicates=false)
- int [parseBindings](#) (QIODevice *deviceBindings)
- bool [parseStatistics](#) (QList< [StatisticObject](#) > &statisticsTimeAll, QList< [StatisticObject](#) > &statisticsTime365, QList< [StatisticObject](#) > &statisticsTime30, QList< [StatisticObject](#) > &statisticsTime7, QList< [StatisticObject](#) > &statisticsTimeToday, QList< [StatisticObject](#) > &statisticsTimeSession) const
- bool [getGroupList](#) (QList< [GroupObject](#) > &groupList) const
- void [setGroupList](#) (const QList< [GroupObject](#) > &groupList)
- bool [addGroup](#) (QIODevice *device, const [GroupObject](#) group)
- bool [deleteGroup](#) (QIODevice *deviceSetup, QIODevice *deviceBindings, const [GroupObject](#) group)
- bool [modifyGroup](#) (QIODevice *deviceSetup, const [GroupObject](#) group, const QString oldGroupName)
- bool [addApplication](#) (QIODevice *deviceSetup, QIODevice *deviceBindings, const QString groupName, const [ApplicationObject](#) application, const QString oldApplicationName="")
- bool [modifyApplication](#) (QIODevice *deviceSetup, QIODevice *deviceBindings, const QString oldApplicationName, const [ApplicationObject](#) application)
- bool [saveApplication](#) (QIODevice *device, const QString groupName, const [ApplicationObject](#) application)
- bool [importApplication](#) (QIODevice *deviceSource, QIODevice *deviceSetup, QIODevice *deviceBindings, unsigned int profileNumber=0)
- bool [deleteApplication](#) (QIODevice *deviceSetup, QIODevice *deviceBindings, const [ApplicationObject](#) application)
- bool [addCategory](#) (QIODevice *device, const QString applicationName, const [CategoryObject](#) category)
- bool [deleteCategory](#) (QIODevice *deviceSetup, QIODevice *deviceBindings, const QString applicationName, const [CategoryObject](#) category)

- bool [modifyCategory](#) (QIODevice *device, const QString applicationName, const [CategoryObject](#) category, const QString oldCategoryName)
- bool [addFunction](#) (QIODevice *deviceSetup, const QString categoryName, const [FunctionObject](#) function)
- bool [deleteFunction](#) (QIODevice *deviceSetup, QIODevice *deviceBindings, const QString categoryName, const [FunctionObject](#) function)
- bool [modifyFunction](#) (QIODevice *deviceSetup, QIODevice *deviceBindings, const QString categoryName, const [FunctionObject](#) function, const QString newFunctionName="")
- bool [thisFunctionExists](#) (const QString applicationName, const QString functionName) const
- bool [addBinding](#) (const mouseButtons buttonEnum, const [ButtonBindingObject](#) binding, bool &foundBindingToRemove, [ButtonBindingObject](#) &bindingToRemove)
- bool [addBindingNoReplace](#) (const mouseButtons buttonEnum, const [ButtonBindingObject](#) binding)
- bool [removeBinding](#) (const mouseButtons buttonEnum, const [ButtonBindingObject](#) binding)
- bool [hasBinding](#) (const mouseButtons buttonEnum, const QString applicationName, const quint32 flags, [ButtonBindingObject](#) &oldBinding)
- bool [isFunctionBound](#) (const QString applicationName, const QString functionName) const
- bool [setFunctionBound](#) (const QString groupName, const QString applicationName, const QString functionName, bool bound)
- bool [updateStatistics](#) (QIODevice *device)
- bool [addStatistics](#) (QIODevice *device, const QList< struct [statistic](#) > &statistics)
- void [getAllStatistics](#) (quint32 *statisticsSingleClickTimeAll, quint32 *statisticsDoubleClickTimeAll, quint32 *statisticsSingleClickTime365, quint32 *statisticsDoubleClickTime365, quint32 *statisticsSingleClickTime30, quint32 *statisticsDoubleClickTime30, quint32 *statisticsSingleClickTime7, quint32 *statisticsDoubleClickTime7, quint32 *statisticsSingleClickTimeToday, quint32 *statisticsDoubleClickTimeToday, quint32 *statisticsSingleClickTimeSession, quint32 *statisticsDoubleClickTimeSession) const
- void [getApplicationStatistics](#) (const QString applicationName, QList< struct [statisticFunctionData](#) > &statisticsTimeAll, QList< struct [statisticFunctionData](#) > &statisticsTime365, QList< struct [statisticFunctionData](#) > &statisticsTime30, QList< struct [statisticFunctionData](#) > &statisticsTime7, QList< struct [statisticFunctionData](#) > &statisticsTimeToday, QList< struct [statisticFunctionData](#) > &statisticsTimeSession, bool isGUI) const
- QDomDocument * [getBindingsDomDocument](#) (void)
- QString [getLanguage](#) (void)
- bool [setLanguage](#) (QIODevice *deviceSetup, const QString newLanguage)

Static Public Member Functions

- static [SetupParser](#) & [instance](#) ()

6.44.1 Detailed Description

A Parser class for the setup file.

Conforms to the Singleton design pattern

6.44.2 Member Function Documentation

6.44.2.1 `bool SetupParser::addApplication (QIODevice * deviceSetup, QIODevice * deviceBindings, const QString groupName, const ApplicationObject application, const QString oldApplicationName = "")`

Adds an application to the setup and bindings files

This is also used for copying application, in which case, give the `oldApplicationName` parameter. The parser will copy the bindings from the application with that name.

Parameters

deviceSetup The setup file (required)

deviceBindings The bindings file (required)

groupName Name of the group in which we are adding application (required)

application The application to add (required)

oldApplicationName If we want to copy bindings, the application we are copying from (optional)

Returns

true if application was successfully added, or false if there was an error

6.44.2.2 `bool SetupParser::addBinding (const mouseButtons buttonEnum, const ButtonBindingObject binding, bool & foundBindingToRemove, ButtonBindingObject & bindingToRemove)`

Adds a binding to the bindings node tree, even if we need to replace another binding

It is caller's responsibility to save the node three into bindings file. This will be changed later to reflect the behavior of other add methods.

Parameters

buttonEnum Mouse button to which we bind function

binding The binding to add

foundBindingToRemove Reference to where we write, whether or not some other binding must be removed to add this

bindingToRemove Reference to where we place the removed binding, if such a binding was found

Returns

true if binding was successfully added to node tree, or false if there was an error

6.44.2.3 `bool SetupParser::addBindingNoReplace (const mouseButtons buttonEnum, const ButtonBindingObject binding)`

Adds a binding only if we don't need to replace another binding

It is caller's responsibility to save the node three into bindings file. This will be changed later to reflect the behavior of other add methods.

Parameters

buttonEnum Mouse button to which we bind function

binding The binding to add

Returns

true if binding was successfully added to node tree, or false if there was another binding, or there was some other error

6.44.2.4 bool SetupParser::addCategory (QIODevice * *device*, const QString *applicationName*, const CategoryObject *category*)

Adds a category to the setup file

Parameters

device The setup file

applicationName Name of the application into which we add this category

category The category to add

Returns

true if category was successfully added, or false if there was an error

6.44.2.5 bool SetupParser::addFunction (QIODevice * *deviceSetup*, const QString *categoryName*, const FunctionObject *function*)

Adds a function to the setup file

Parameters

deviceSetup The setup file

categoryName Name of the category into which we add this function

function The function to add

Returns

true if function was successfully added, or false if there was an error

6.44.2.6 bool SetupParser::addGroup (QIODevice * *device*, const GroupObject *group*)

Adds a group to the setup file

Parameters

device The setup file

group The group to add

Returns

true if group was successfully added, or false if there was an error

6.44.2.7 bool SetupParser::addStatistics (QIODevice * *device*, const QList< struct statistic > & *statistics*)

Adds some statistics to the statistics file

Parameters

device The statistics file

statistics List of statistics to add

Returns

true if statistics were successfully added, or false if there was an error

6.44.2.8 bool SetupParser::deleteApplication (QIODevice * *deviceSetup*, QIODevice * *deviceBindings*, const ApplicationObject *application*)

Deletes an application from setup and bindings file

Parameters

deviceSetup The setup file

deviceBindings The bindings file

application The application to delete

Returns

true if application was successfully deleted, or false if there was an error

6.44.2.9 bool SetupParser::deleteCategory (QIODevice * *deviceSetup*, QIODevice * *deviceBindings*, const QString *applicationName*, const CategoryObject *category*)

Deletes a category from setup and bindings file

Parameters

deviceSetup The setup file

deviceBindings The bindings file

applicationName Name of the application from which we delete this category

category The category to delete

Returns

true if category was successfully deleted, or false if there was an error

6.44.2.10 bool SetupParser::deleteFunction (QIODevice * *deviceSetup*, QIODevice * *deviceBindings*, const QString *categoryName*, const FunctionObject *function*)

Deletes a function from setup and bindings file

Parameters

deviceSetup The setup file
deviceBindings The bindings file
categoryName Name of the category from which we delete this function
function The function to delete

Returns

true if function was successfully deleted, or false if there was an error

6.44.2.11 bool SetupParser::deleteGroup (QIODevice * *deviceSetup*, QIODevice * *deviceBindings*, const GroupObject *group*)

Deletes a group from setup and bindings file

Parameters

deviceSetup The setup file
deviceBindings The bindings file
group The group to delete

Returns

true if group was successfully deleted, or false if there was an error

6.44.2.12 void SetupParser::getAllStatistics (quint32 * *statisticsSingleClickTimeAll*, quint32 * *statisticsDoubleClickTimeAll*, quint32 * *statisticsSingleClickTime365*, quint32 * *statisticsDoubleClickTime365*, quint32 * *statisticsSingleClickTime30*, quint32 * *statisticsDoubleClickTime30*, quint32 * *statisticsSingleClickTime7*, quint32 * *statisticsDoubleClickTime7*, quint32 * *statisticsSingleClickTimeToday*, quint32 * *statisticsDoubleClickTimeToday*, quint32 * *statisticsSingleClickTimeSession*, quint32 * *statisticsDoubleClickTimeSession*) const

Gets click statistics for all applications, in all the different time frames

Arrays for all single click statistics must have (NUM_BUTTONS + NUM_UNPROGRAMMABLE_BUTTONS) indices, and for all double-click statistics, NUM_DOUBLECLICK_BUTTONS indices.

Parameters

statisticsSingleClickTimeAll Array for single click statistics for clicks older than 365 days
statisticsDoubleClickTimeAll Array for double-click statistics for clicks older than 365 days
statisticsSingleClickTime365 Array for single click statistics for 365...31 days old clicks
statisticsDoubleClickTime365 Array for double-click statistics for 365...31 days old clicks
statisticsSingleClickTime30 Array for single click statistics for 30...8 days old clicks
statisticsDoubleClickTime30 Array for double-click statistics for 30...8 days old clicks
statisticsSingleClickTime7 Array for single click statistics for 7...1 days old clicks
statisticsDoubleClickTime7 Array for double-click statistics for 7...1 days old clicks
statisticsSingleClickTimeToday Array for single click statistics for today's clicks

statisticsDoubleClickTimeToday Array for double-click statistics for today's clicks

statisticsSingleClickTimeSession Array for single click statistics that happened after the program was launched

statisticsDoubleClickTimeSession Array for double-click statistics that happened after the program was launched

6.44.2.13 `void SetupParser::getApplicationStatistics (const QString applicationName, QList< struct statisticFunctionData > & statisticsTimeAll, QList< struct statisticFunctionData > & statisticsTime365, QList< struct statisticFunctionData > & statisticsTime30, QList< struct statisticFunctionData > & statisticsTime7, QList< struct statisticFunctionData > & statisticsTimeToday, QList< struct statisticFunctionData > & statisticsTimeSession, bool isGUI) const`

Gets click statistics for given applications, in all the different time frames

Gives more information than the statistics for all applications. The lists contain both the mouse button that the clicks are for, and the function that is bound to the mouse button. The structs in the list contain the single- and double-click statistics.

Parameters

applicationName Name of the application for which we get statistics

statisticsTimeAll Reference to where statistics for clicks older than 365 days are placed

statisticsTime365 Reference to where statistics for 365...31 days old clicks are placed

statisticsTime30 Reference to where statistics for 30...8 days old clicks are placed

statisticsTime7 Reference to where statistics for 7...1 days old clicks are placed

statisticsTimeToday Reference to where statistics for today's clicks are placed

statisticsTimeSession Reference to where statistics that happened after the program was launched are placed

isGUI True if we want information in human-readable format, or false if internal format

6.44.2.14 `QDomDocument * SetupParser::getBindingsDomDocument (void)`

Gives a pointer to the bindings node tree, for writing into file

Returns

The pointer to node tree

6.44.2.15 `bool SetupParser::getGroupList (QList< GroupObject > & groupList) const`

Getter for list of groups according to the setup file

Parameters

groupList Reference to the list to be filled with [GroupObject](#) instances

Returns

true if list was given successfully, or false if there was an error (for example, setup file was not parsed yet)

6.44.2.16 QString SetupParser::getLanguage (void)

Gets the country code for current language

Returns

The two character country code

6.44.2.17 bool SetupParser::hasBinding (const mouseButtons *buttonEnum*, const QString *applicationName*, const quint32 *flags*, ButtonBindingObject & *oldBinding*)

Checks if there is a binding in given mouse button, with given flags, in given application

Parameters

buttonEnum Mouse button in which we seek the given binding

applicationName Name of the application in which we seek the given binding

flags The combination of flags that the binding must have if it is to be found

oldBinding Reference to where the binding is placed, if it was found

Returns

true if the binding exists, or false if there is no such binding

6.44.2.18 bool SetupParser::importApplication (QIODevice * *deviceSource*, QIODevice * *deviceSetup*, QIODevice * *deviceBindings*, unsigned int *profileNumber* = 0)

Imports an application from an exported xml file into the setup xml file

Parameters

deviceSource The file to be imported

deviceSetup The setup file

deviceBindings The bindings file

profileNumber Profile number for the new application

Returns

true if application was successfully imported, or false if there was an error

6.44.2.19 static SetupParser& SetupParser::instance () [inline, static]

Gives a reference to the Singleton object

Returns

reference to the object

The Singleton instance

6.44.2.20 `bool SetupParser::isFunctionBound (const QString applicationName, const QString functionName) const`

Checks if the given function is bound to any button in given application

Parameters

applicationName Name of the application in which we seek the function

functionName Name of the function we are seeking

Returns

true if the function is bound, or false if it isn't

6.44.2.21 `bool SetupParser::modifyApplication (QIODevice * deviceSetup, QIODevice * deviceBindings, const QString oldApplicationName, const ApplicationObject application)`

Changes an application in setup and bindings files to reflect the given application's data

Parameters

deviceSetup The setup file

deviceBindings The bindings file

oldApplicationName The old name of the application, since the name may have changed.

application The application that we are changing

Returns

true if application was successfully modified, or false if there was an error

6.44.2.22 `bool SetupParser::modifyCategory (QIODevice * device, const QString applicationName, const CategoryObject category, const QString oldCategoryName)`

Changes a category in setup and bindings files to reflect the given category's data

Parameters

device The setup file

applicationName Name of the application in which we modify this category

category The category that we are changing

oldCategoryName The old name of the category, since the name may have changed.

Returns

true if category was successfully modified, or false if there was an error

6.44.2.23 `bool SetupParser::modifyFunction (QIODevice * deviceSetup, QIODevice * deviceBindings, const QString categoryName, const FunctionObject function, const QString newFunctionName = "")`

Changes a function in setup and bindings files to reflect the given function's data

There is an exception that will be fixed later: In this method, the function object must have the old name, and *newFunctionName* contains the new one, if we are changing function's name. This will be changed to reflect the behavior of other methods in which the string contains the old name and the object's name is changed to the new one.

Parameters

deviceSetup The setup file (required)

deviceBindings The bindings file (required)

categoryName Name of the category in which we modify this function (required)

function The function that we are changing, except for the name (required)

newFunctionName If we wish to change function, new name for it (optional)

Returns

true if function was successfully modified, or false if there was an error

6.44.2.24 `bool SetupParser::modifyGroup (QIODevice * deviceSetup, const GroupObject group, const QString oldGroupName)`

Changes a group in setup file to reflect the given group's data

Parameters

deviceSetup The setup file

group The group that we are changing

oldGroupName The old name of the group, since the name may have changed.

Returns

true if group was successfully modified, or false if there was an error

6.44.2.25 `int SetupParser::parseBindings (QIODevice * deviceBindings)`

Parse the previously read bindings file

Parameters

deviceBindings The bindings file

Returns

true if file was successfully parsed, or false if there was an error

6.44.2.26 `int SetupParser::parseSetup (QIODevice * deviceSetup, bool ignoreDuplicates = false)`

Parse the previously read setup file

Parameters

deviceSetup The setup file

ignoreDuplicates Do not check if there are several profiles with same number

Returns

true if file was successfully parsed, or false if there was an error

6.44.2.27 `bool SetupParser::parseStatistics (QList< StatisticObject > & statisticsTimeAll, QList< StatisticObject > & statisticsTime365, QList< StatisticObject > & statisticsTime30, QList< StatisticObject > & statisticsTime7, QList< StatisticObject > & statisticsTimeToday, QList< StatisticObject > & statisticsTimeSession) const`

Parse the previously read statistics file

Parameters

statisticsTimeAll Reference to where statistics older than 365 days are placed as a list

statisticsTime365 Reference to where statistics that are 365...31 days old are placed as a list

statisticsTime30 Reference to where statistics that are 30...8 days old are placed as a list

statisticsTime7 Reference to where statistics that are 7...1 days old are placed as a list

statisticsTimeToday Reference to where today's statistics are placed as a list

statisticsTimeSession Reference to where the statistics starting from when this program was launched, are placed as a list

Returns

true if statistics file was successfully parsed, or false if there was an error

6.44.2.28 `bool SetupParser::readBindingsFile (QIODevice * device)`

Receives the bindings file, for later parsing

Parameters

device The bindings file

Returns

true if file was read successfully, or false if there was an error

6.44.2.29 bool SetupParser::readSetupFile (QIODevice * *device*)

Receives the setup file, for later parsing

Parameters

device The setup file

Returns

true if file was read successfully, or false if there was an error

6.44.2.30 bool SetupParser::readStatisticsFile (QIODevice * *device*)

Receives the statistics file, for later parsing

Parameters

device The statistics file

Returns

true if file was read successfully, or false if there was an error

6.44.2.31 bool SetupParser::removeBinding (const mouseButtons *buttonEnum*, const ButtonBindingObject *binding*)

Removes a binding from the bindings node tree

It is caller's responsibility to save the node three into bindings file. This will be changed later to reflect the behavior of other delete methods.

Parameters

buttonEnum Mouse button from which we remove binding

binding The binding to remove

Returns

true if binding was successfully removed from node tree, or false if there was an error

6.44.2.32 bool SetupParser::saveApplication (QIODevice * *device*, const QString *groupName*, const ApplicationObject *application*)

Exports an application as an importable xml file

Parameters

device The setup file

groupName The exported application's group's name

application The application to export

Returns

true if application was successfully exported, or false if there was an error

6.44.2.33 `bool SetupParser::setFunctionBound (const QString groupName, const QString applicationName, const QString functionName, bool bound)`

Updates the local group list so that the given function is marked bound or unbound

Does not read the files. This function is for optimization, so that we don't need to read the entire files after doing a small change.

Parameters

groupName Name of the group in which we are modifying function

applicationName Name of the application in which we are modifying function

functionName Name of the function to modify

bound True if we are setting function as bound, or false if as unbound

6.44.2.34 `void SetupParser::setGroupList (const QList< GroupObject > & groupList)`

Setter for list of groups, owned by parser

Parameters

groupList New group list

6.44.2.35 `bool SetupParser::setLanguage (QIODevice * deviceSetup, const QString newLanguage)`

Sets the current language

Parameters

deviceSetup The setup file

newLanguage The two character country code of new language

Returns

true if language was successfully set, or false if there was an error

6.44.2.36 `bool SetupParser::thisFunctionExists (const QString applicationName, const QString functionName) const`

Checks if the setup file has a function in given application, with given function name

Parameters

applicationName Name of the application in which we are searching

functionName Name of the function that we are searching

Returns

true if function exists, or false if it doesn't

6.44.2.37 bool SetupParser::updateStatistics (QIODevice * *device*)

Updates the statistics file to reflect the information in local node tree

Parameters

device The statistics file

Returns

true if file was successfully updated, or false if there was an error

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

- [src/setupparser.h](#)
- [src/setupparser.cpp](#)

6.45 SimpleScreen Class Reference

```
#include <simplescreen.h>
```

Public Slots

- void [showThisView](#) (QString groupName, QString applicationName, QPoint center)
- void [refreshView](#) (QString groupName, QString applicationName, bool fromOtherView=true)
- void [quitModeware](#) ()
- void [parseConfig](#) ()

Signals

- void [changeView](#) (QString groupName, QString applicationName, QPoint center)
- void [closeOtherView](#) ()
- void [activateProfileOtherView](#) (unsigned int profileNum, bool reload)
- void [updateActiveMode](#) (void)

Public Member Functions

- [SimpleScreen](#) (QWidget *parent=0)
- [~SimpleScreen](#) ()

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.45.1 Detailed Description

A class which models the simple screen window

6.45.2 Constructor & Destructor Documentation

6.45.2.1 SimpleScreen::SimpleScreen (QWidget * *parent* = 0)

Constructor

Parameters

parent Parent of this QObject (optional)

6.45.2.2 SimpleScreen::~SimpleScreen ()

Destructor

6.45.3 Member Function Documentation

6.45.3.1 void SimpleScreen::activateProfileOtherView (unsigned int *profileNum*, bool *reload*) [signal]

Activate mode with this profile number in advanced view

Parameters

profileNum Profile number to activate

reload If setup contents may have been changed in advanced view, also reload setup

6.45.3.2 void SimpleScreen::changeEvent (QEvent * *e*) [protected]

Reimplementation of QObject's changeEvent

Parameters

e Event given by Qt

6.45.3.3 void SimpleScreen::changeView (QString *groupName*, QString *applicationName*, QPoint *center*) [signal]

Switch to advanced view

Can also change advanced view's active group and application

Parameters

groupName Name of the active group, or empty if no change

applicationName Name of the active application, or empty if no change

center Coordinates of the center of shown window

6.45.3.4 void SimpleScreen::closeOtherView () [signal]

Close advanced view

6.45.3.5 void SimpleScreen::parseConfig () [slot]

Re-read the setup and bindings xml files to local group list

6.45.3.6 void SimpleScreen::quitModeware () [slot]

Exit the software

6.45.3.7 void SimpleScreen::refreshView (QString groupName, QString applicationName, bool fromOtherView = true) [slot]

Change active application based on group and application name

Can also command [MainWindow](#) to refresh, if this command doesn't already come from there

Parameters

groupName The application's group

applicationName The application's name

fromOtherView True if this command comes from [MainWindow](#) (optional)

6.45.3.8 void SimpleScreen::showThisView (QString groupName, QString applicationName, QPoint center) [slot]

Unhide this window

Can also change active group and application

Parameters

groupName Name of the active group, or empty if no change

applicationName Name of the active application, or empty if no change

center Coordinates of the center of shown window

6.45.3.9 void SimpleScreen::updateActiveMode (void) [signal]

Update the mouse flash contents for the active application

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

- [src/simplescreen.h](#)
- [src/simplescreen.cpp](#)

6.46 statistic Struct Reference

Public Attributes

- **QString application**

- QString **function**
- mouseButtons **mouseButton**
- bool **doubleclick**
- quint32 **clicks**

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

- [src/setupparser.h](#)

6.47 statisticFunctionData Struct Reference

Public Attributes

- QString **function**
- mouseButtons **mouseButton**
- bool **doubleclick**
- quint32 **clicks**

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

- [src/setup_common.h](#)

6.48 StatisticObject Class Reference

```
#include <statisticobject.h>
```

Public Member Functions

- **StatisticObject** (const QString applicationName)
- **StatisticObject** (const **StatisticObject** &source)
- bool **operator<** (const **StatisticObject** &target) const
- bool **operator>** (const **StatisticObject** &target) const
- bool **operator==** (const **StatisticObject** &right) const
- **StatisticObject** & **operator=** (const **StatisticObject** &right)
- QString **getApplicationName** (void) const
- void **setApplicationName** (const QString newApplicationName)
- bool **getFunctionClicks** (const QString functionName, quint32 &clicks, bool &doubleClick, mouseButtons &button)
- void **setFunctionClicks** (const QString functionName, quint32 clicks, bool doubleClick, mouseButtons button)
- void **getAllClicks** (QList< struct **statisticFunctionData** > &functionClicks) const
- void **sortFunctions** (void)

6.48.1 Detailed Description

A class which models mouse button click statistics for a specific application

6.48.2 Constructor & Destructor Documentation

6.48.2.1 `StatisticObject::StatisticObject (const StatisticObject & source)`

Copy constructor

Parameters

source `StatisticObject` to be copied

6.48.3 Member Function Documentation

6.48.3.1 `void StatisticObject::getAllClicks (QList< struct statisticFunctionData > & functionClicks) const`

Get a list of statistics that contains clicks and double-clicks for all functions

Parameters

functionClicks Reference to where the list is placed

6.48.3.2 `QString StatisticObject::getApplicationName (void) const`

Get name of the application that these statistics are for

Returns

Name of the application

6.48.3.3 `bool StatisticObject::getFunctionClicks (const QString functionName, quint32 & clicks, bool & doubleClick, mouseButtons & button)`

Get the number of clicks or double-clicks for given function

Parameters

functionName Name of the function, for which we get the clicks

clicks Reference to where the number of clicks is placed

doubleClick True if we want number of double-clicks, false if single clicks

button Reference to where we place, what button the function is bound to in this particular statistic object

Returns

true if clicks were successfully given, or false if there is no such function in the statistics

6.48.3.4 bool StatisticObject::operator< (const StatisticObject & *target*) const

Operator <, compares statistic object's function

Parameters

target Statistic object we compare to

Returns

as (statistic.getName() < target.getName())

6.48.3.5 StatisticObject & StatisticObject::operator= (const StatisticObject & *right*)

Operator =, performs deep copy

Parameters

right Right side of assignment

Returns

[StatisticObject](#) reference

6.48.3.6 bool StatisticObject::operator== (const StatisticObject & *right*) const

Operator ==, compares statistic object's function

Parameters

right Statistic object we compare to

Returns

as (statistic.getName() == target.getName())

6.48.3.7 bool StatisticObject::operator> (const StatisticObject & *target*) const

Operator >, compares statistic object's function

Parameters

target Statistic object we compare to

Returns

as (statistic.getName() > target.getName())

6.48.3.8 void StatisticObject::setApplicationName (const QString *newApplicationName*)

Set name of the application that these statistics are for

Parameters

newApplicationName New name of the application

6.48.3.9 void `StatisticObject::setFunctionClicks` (const `QString` *functionName*, `quint32` *clicks*, `bool` *doubleClick*, `mouseButtons` *button*)

Set the number of clicks or double-clicks for given function

Both function and mouse button must be given, to uniquely identify the statistic, because a function can be bound to several buttons

Parameters

- functionName* Name of the function, for which we set the clicks
- clicks* New number of clicks
- doubleClick* True if we set number of double-clicks, false if single clicks
- button* The button for which we set statistics

6.48.3.10 void `StatisticObject::sortFunctions` (void)

Sort the functions so that list of all statistics will be in alphabetical order

Do this every time you add one or more functions. If you add several functions at the same time, you only need to do this once, after adding them.

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

- [src/statisticobject.h](#)
- [src/statisticobject.cpp](#)

6.49 StatisticsAllDialog Class Reference

```
#include <statisticsalldialog.h>
```

Public Member Functions

- [StatisticsAllDialog](#) (`quint32` **statisticsSingleClickTimeAll*, `quint32` **statisticsDoubleClickTimeAll*, `quint32` **statisticsSingleClickTime365*, `quint32` **statisticsDoubleClickTime365*, `quint32` **statisticsSingleClickTime30*, `quint32` **statisticsDoubleClickTime30*, `quint32` **statisticsSingleClickTime7*, `quint32` **statisticsDoubleClickTime7*, `quint32` **statisticsSingleClickTimeToday*, `quint32` **statisticsDoubleClickTimeToday*, `quint32` **statisticsSingleClickTimeSession*, `quint32` **statisticsDoubleClickTimeSession*, `QWidget` **parent*=0)
- [~StatisticsAllDialog](#) ()

Protected Member Functions

- void [changeEvent](#) (`QEvent` **e*)

6.49.1 Detailed Description

A class which models dialog for displaying click statistics for all modes

6.49.2 Constructor & Destructor Documentation

6.49.2.1 `StatisticsAllDialog::StatisticsAllDialog (quint32 * statisticsSingleClickTimeAll, quint32 * statisticsDoubleClickTimeAll, quint32 * statisticsSingleClickTime365, quint32 * statisticsDoubleClickTime365, quint32 * statisticsSingleClickTime30, quint32 * statisticsDoubleClickTime30, quint32 * statisticsSingleClickTime7, quint32 * statisticsDoubleClickTime7, quint32 * statisticsSingleClickTimeToday, quint32 * statisticsDoubleClickTimeToday, quint32 * statisticsSingleClickTimeSession, quint32 * statisticsDoubleClickTimeSession, QWidget * parent = 0)`

Constructor

Arrays for all single click statistics must have (NUM_BUTTONS + NUM_UNPROGRAMMABLE_BUTTONS) indices, and for all double-click statistics, NUM_DOUBLECLICK_BUTTONS indices.

Parameters

statisticsSingleClickTimeAll Array for single click statistics for clicks older than 365 days (required)
statisticsDoubleClickTimeAll Array for double-click statistics for clicks older than 365 days (required)

statisticsSingleClickTime365 Array for single click statistics for 365...31 days old clicks (required)
statisticsDoubleClickTime365 Array for double-click statistics for 365...31 days old clicks (required)

statisticsSingleClickTime30 Array for single click statistics for 30...8 days old clicks (required)

statisticsDoubleClickTime30 Array for double-click statistics for 30...8 days old clicks (required)

statisticsSingleClickTime7 Array for single click statistics for 7...1 days old clicks (required)

statisticsDoubleClickTime7 Array for double-click statistics for 7...1 days old clicks (required)

statisticsSingleClickTimeToday Array for single click statistics for today's clicks (required)

statisticsDoubleClickTimeToday Array for double-click statistics for today's clicks (required)

statisticsSingleClickTimeSession Array for single click statistics that happened after the program was launched (required)

statisticsDoubleClickTimeSession Array for double-click statistics that happened after the program was launched (required)

parent Parent of this QObject (optional)

6.49.2.2 `StatisticsAllDialog::~StatisticsAllDialog ()`

Destructor

6.49.3 Member Function Documentation

6.49.3.1 `void StatisticsAllDialog::changeEvent (QEvent * e) [protected]`

Reimplementation of QObject's changeEvent

Parameters

e Event given by Qt

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

- [src/statisticsalldialog.h](#)
- [src/statisticsalldialog.cpp](#)

6.50 StatisticsAllModel Class Reference

```
#include <statisticsallmodel.h>
```

Public Member Functions

- **StatisticsAllModel** (quint32 *statisticsSingleClick, quint32 *statisticsDoubleClick, QObject *parent=0)
- int **rowCount** (const QModelIndex &parent=QModelIndex()) const
Implementation of the virtual function rowCount(...) in QAbstractTableModel.
- int **columnCount** (const QModelIndex &parent=QModelIndex()) const
Implementation of the virtual function columnCount(...) in QAbstractTableModel.
- QVariant **data** (const QModelIndex &index, int role) const
Implementation of the virtual function data(...) in QAbstractTableModel.
- QVariant **headerData** (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const
Implementation of the virtual function headerData(...) in QAbstractTableModel.

6.50.1 Detailed Description

A class which models a table of click statistics for all modes

6.50.2 Constructor & Destructor Documentation

6.50.2.1 StatisticsAllModel::StatisticsAllModel (quint32 * statisticsSingleClick, quint32 * statisticsDoubleClick, QObject * parent = 0)

Constructor

Array for single click statistics must have (NUM_BUTTONS + NUM_UNPROGRAMMABLE_BUTTONS) indices, and for double-click statistics, NUM_DOUBLECLICK_BUTTONS indices.

Parameters

- statisticsSingleClick* Array for single click statistics (required)
- statisticsDoubleClick* Array for double-click statistics (required)
- parent* Parent of this QObject (optional)

6.50.3 Member Function Documentation

6.50.3.1 int StatisticsAllModel::columnCount (const QModelIndex & parent = QModelIndex ()) const

Implementation of the virtual function columnCount(...) in QAbstractTableModel.

Parameters

- parent* Parent, for which we return column count (optional)

Returns

2 always

6.50.3.2 QVariant StatisticsAllModel::data (const QModelIndex & *index*, int *role*) const

Implementation of the virtual function data(...) in QAbstractTableModel.

Returns the data of the desired statistic

Parameters

index Reference to the index, from which we get the statistics data

role See Qt documentation of views

Returns

The data of the statistic line

6.50.3.3 QVariant StatisticsAllModel::headerData (int *section*, Qt::Orientation *orientation*, int *role* = Qt::DisplayRole) const

Implementation of the virtual function headerData(...) in QAbstractTableModel.

Gives the headers for those views that use them

Parameters

section See Qt documentation of views

orientation See Qt documentation of views

role See Qt documentation of views

Returns

The data for the headers

6.50.3.4 int StatisticsAllModel::rowCount (const QModelIndex & *parent* = QModelIndex()) const

Implementation of the virtual function rowCount(...) in QAbstractTableModel.

Returns the number of statistics lines in the table

Parameters

parent Parent, for which we return row count (optional)

Returns

The number of statistics lines

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

- [src/statisticsallmodel.h](#)
- [src/statisticsallmodel.cpp](#)

6.51 StatisticsDialog Class Reference

```
#include <statisticsdialog.h>
```

Public Member Functions

- [StatisticsDialog](#) (const QList< struct [statisticFunctionData](#) > &statisticsAll, const QList< struct [statisticFunctionData](#) > &statistics365, const QList< struct [statisticFunctionData](#) > &statistics30, const QList< struct [statisticFunctionData](#) > &statistics7, const QList< struct [statisticFunctionData](#) > &statisticsToday, const QList< struct [statisticFunctionData](#) > &statisticsSession, QString applicationName, QWidget *parent=0)
- [~StatisticsDialog](#) ()

Protected Member Functions

- void [changeEvent](#) (QEvent *e)

6.51.1 Detailed Description

A class which models dialog for displaying click statistics for a specific mode

6.51.2 Constructor & Destructor Documentation

- 6.51.2.1 StatisticsDialog::StatisticsDialog (const QList< struct [statisticFunctionData](#) > & *statisticsAll*, const QList< struct [statisticFunctionData](#) > & *statistics365*, const QList< struct [statisticFunctionData](#) > & *statistics30*, const QList< struct [statisticFunctionData](#) > & *statistics7*, const QList< struct [statisticFunctionData](#) > & *statisticsToday*, const QList< struct [statisticFunctionData](#) > & *statisticsSession*, QString *applicationName*, QWidget * *parent* = 0)**

Constructor

Parameters

- statisticsAll* Statistics for clicks older than 365 days (required)
statistics365 Statistics for 365...31 days old clicks (required)
statistics30 Statistics for 30...8 days old clicks (required)
statistics7 Statistics for 7...1 days old clicks (required)
statisticsToday Statistics for today's clicks (required)
statisticsSession Statistics that happened after the program was launched (required)
applicationName Name of the application that these statistics are for (required)
parent Parent of this QObject (optional)

6.51.2.2 StatisticsDialog::~StatisticsDialog ()

Destructor

6.51.3 Member Function Documentation

6.51.3.1 void StatisticsDialog::changeEvent (QEvent * e) [protected]

Reimplementation of QObject's changeEvent

Parameters

e Event given by Qt

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

- [src/statisticsdialog.h](#)
- [src/statisticsdialog.cpp](#)

6.52 StatisticsModel Class Reference

```
#include <statisticsmodel.h>
```

Public Member Functions

- [StatisticsModel](#) (const QList< struct [statisticFunctionData](#) > &statisticsList, QObject *parent=0)
- int [rowCount](#) (const QModelIndex &parent=QModelIndex()) const
Implementation of the virtual function rowCount(...) in QAbstractTableModel.
- int [columnCount](#) (const QModelIndex &parent=QModelIndex()) const
Implementation of the virtual function columnCount(...) in QAbstractTableModel.
- QVariant [data](#) (const QModelIndex &index, int role) const
Implementation of the virtual function data(...) in QAbstractTableModel.
- QVariant [headerData](#) (int section, Qt::Orientation orientation, int role=Qt::DisplayRole) const
Implementation of the virtual function headerData(...) in QAbstractTableModel.

6.52.1 Detailed Description

A class which models a table of click statistics for a specific mode

6.52.2 Constructor & Destructor Documentation

6.52.2.1 StatisticsModel::StatisticsModel (const QList< struct statisticFunctionData > &statisticsList, QObject * parent = 0)

Constructor

Parameters

statisticsList Single- and double-click statistics to display (required)
parent Parent of this QObject (optional)

6.52.3 Member Function Documentation

6.52.3.1 `int StatisticsModel::columnCount (const QModelIndex & parent = QModelIndex ()) const`

Implementation of the virtual function `columnCount(...)` in `QAbstractTableModel`.

Parameters

parent Parent, for which we return column count (optional)

Returns

2 always

6.52.3.2 `QVariant StatisticsModel::data (const QModelIndex & index, int role) const`

Implementation of the virtual function `data(...)` in `QAbstractTableModel`.

Returns the data of the desired statistic

Parameters

index Reference to the index, from which we get the statistics data

role See Qt documentation of views

Returns

The data of the statistic line

6.52.3.3 `QVariant StatisticsModel::headerData (int section, Qt::Orientation orientation, int role = Qt::DisplayRole) const`

Implementation of the virtual function `headerData(...)` in `QAbstractTableModel`.

Gives the headers for those views that use them

Parameters

section See Qt documentation of views

orientation See Qt documentation of views

role See Qt documentation of views

Returns

The data for the headers

6.52.3.4 `int StatisticsModel::rowCount (const QModelIndex & parent = QModelIndex ()) const`

Implementation of the virtual function `rowCount(...)` in `QAbstractTableModel`.

Returns the number of statistics lines in the table

Parameters

parent Parent, for which we return row count (optional)

Returns

The number of statistics lines

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

- [src/statisticsmodel.h](#)
- [src/statisticsmodel.cpp](#)

6.53 stringToKeyInfo Struct Reference

Public Attributes

- const char * **internalString**
- const char * **GUIText**
- int **keyQt**
- quint8 **HID**
- int **forcedModifier**
- bool **numPad**

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

- [src/keypress_common.h](#)

6.54 stringToModifierInfo Struct Reference

Public Attributes

- const char * **internalString**
- const char * **GUIText**
- int **keyQt**
- bool **right**
- unsigned int **specialNativeScanCode**
- quint8 **HID**

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

- [src/keypress_common.h](#)

Chapter 7

File Documentation

7.1 src/addapplicationdialog.h File Reference

Defines the [AddApplicationDialog](#) class, which creates a dialog for adding an application.

```
#include <QtGui/QDialog>
#include "groupobject.h"
#include "mydialogs_common.h"
```

Classes

- class [AddApplicationDialog](#)

Namespaces

- namespace [Ui](#)

7.1.1 Detailed Description

Defines the [AddApplicationDialog](#) class, which creates a dialog for adding an application.

7.2 src/addfunctiondialog.h File Reference

Defines the [AddFunctionDialog](#) class, which creates a dialog for adding a function.

```
#include <QtGui/QDialog>
#include <QKeyEvent>
#include <QList>
#include <QToolButton>
#include "mydialogs_common.h"
#include "keypress_common.h"
```

```
#include "keyobject.h"
```

Classes

- class [AddFunctionDialog](#)

Namespaces

- namespace [Ui](#)

7.2.1 Detailed Description

Defines the [AddFunctionDialog](#) class, which creates a dialog for adding a function.

7.3 src/applicationlistmodel.h File Reference

Defines the [ApplicationListModel](#) class, subclassed from [QAbstractListModel](#).

```
#include <QAbstractListModel>
#include <QObject>
#include <QList>
#include <QAbstractItemModel>
#include "groupobject.h"
#include "applicationobject.h"
```

Classes

- class [ApplicationListModel](#)

7.3.1 Detailed Description

Defines the [ApplicationListModel](#) class, subclassed from [QAbstractListModel](#).

7.4 src/applicationobject.h File Reference

Defines the [ApplicationObject](#) class.

```
#include <QObject>
#include "setup_common.h"
#include "categoryobject.h"
```

Classes

- class [ApplicationObject](#)

7.4.1 Detailed Description

Defines the [ApplicationObject](#) class.

7.5 src/AtUsbHid.h File Reference

Header file for using AtUsbHid DLL, from Atmel.

```
#include <stdio.h>
```

Defines

- #define **ERROR_USB_DEVICE_NOT_FOUND** 0xE0000001
- #define **ERROR_USB_DEVICE_NO_CAPABILITIES** 0xE0000002
- #define **AT_USB_HID_DLL** L"AtUsbHid"
- #define **ATUSBHID_API** extern "C" __declspec(dllimport)
- #define **chBEGINTHREADEX**(psa, cbStack, pfnStartAddr, pvParam, fdwCreate, pdwThreadId)
- #define **STDCALL** __stdcall
- #define **DECLARE_FUNCTION_POINTER**(FUNC) PF_##FUNC lp##FUNC=NULL;
- #define **LOAD_FUNCTION_POINTER**(DLL, FUNC) lp##FUNC = (PF_##FUNC)GetProcAddress(DLL, #FUNC);
- #define **ADDR_CHECK**(FUNC) if (lp##FUNC == NULL) {fprintf(stderr, "%s\n", "Error: Cannot get address of function."); return FALSE;}
- #define **DYNCALL**(FUNC) lp##FUNC

Typedefs

- typedef unsigned(__stdcall * **PTHREAD_START**)(void *)
- typedef BOOLEAN(STDCALL * **PF_findHidDevice**)(const UINT VendorID, const UINT ProductID)
- typedef void(STDCALL * **PF_closeDevice**)()
- typedef BOOLEAN(STDCALL * **PF_writeData**)(UCHAR *buffer)
- typedef BOOLEAN(STDCALL * **PF_readData**)(UCHAR *buffer)
- typedef int(STDCALL * **PF_hidRegisterDeviceNotification**)(HWND hWnd)
- typedef void(STDCALL * **PF_hidUnregisterDeviceNotification**)(HWND hWnd)
- typedef int(STDCALL * **PF_isMyDeviceNotification**)(DWORD dwData)
- typedef BOOLEAN(STDCALL * **PF_setFeature**)(UCHAR *buffer)
- typedef int(STDCALL * **PF_getFeatureReportLength**)()
- typedef int(STDCALL * **PF_getInputReportLength**)()
- typedef int(STDCALL * **PF_getOutputReportLength**)()

Functions

- ATUSBHID_API BOOLEAN STDCALL **findHidDevice** (const UINT VendorID, const UINT ProductID)
- ATUSBHID_API void STDCALL **closeDevice** (void)
- ATUSBHID_API BOOLEAN STDCALL **writeData** (UCHAR *buf)
- ATUSBHID_API BOOLEAN STDCALL **readData** (UCHAR *buffer)
- ATUSBHID_API int STDCALL **hidRegisterDeviceNotification** (HWND hWnd)

- ATUSBHID_API void STDCALL **hidUnregisterDeviceNotification** (HWND hWnd)
- ATUSBHID_API int STDCALL **isMyDeviceNotification** (DWORD dwData)
- ATUSBHID_API BOOLEAN STDCALL **setFeature** (UCHAR *buffer)
- ATUSBHID_API int STDCALL **getFeatureReportLength** (void)
- ATUSBHID_API int STDCALL **getOutputReportLength** (void)
- ATUSBHID_API int STDCALL **getInputReportLength** (void)
- **DECLARE_FUNCTION_POINTER** (findHidDevice)
- **DECLARE_FUNCTION_POINTER** (closeDevice)
- **DECLARE_FUNCTION_POINTER** (writeData)
- **DECLARE_FUNCTION_POINTER** (readData)
- **DECLARE_FUNCTION_POINTER** (hidRegisterDeviceNotification)
- **DECLARE_FUNCTION_POINTER** (hidUnregisterDeviceNotification)
- **DECLARE_FUNCTION_POINTER** (isMyDeviceNotification)
- **DECLARE_FUNCTION_POINTER** (setFeature)
- **DECLARE_FUNCTION_POINTER** (getFeatureReportLength)
- **DECLARE_FUNCTION_POINTER** (getOutputReportLength)
- **DECLARE_FUNCTION_POINTER** (getInputReportLength)

7.5.1 Detailed Description

Header file for using AtUsbHid DLL, from Atmel.

7.5.2 Define Documentation

7.5.2.1 `#define chBEGINTHREADEX(psa, cbStack, pfnStartAddr, pvParam, fdwCreate, pdwThreadId)`

Value:

```
(HANDLE)_beginthreadex(
    (void *)          (psa),
    (unsigned)        (cbStack),
    (PTHREAD_START) (pfnStartAddr),
    (void *)          (pvParam),
    (unsigned)        (fdwCreate),
    (unsigned *)      (pdwThreadId))
```

7.6 src/autoswitchdetailsdialog.h File Reference

Defines the [AutoswitchDetailsDialog](#) class, which creates a dialog for changing autoswitch details of an application.

```
#include <QtGui/QDialog>
#include "mydialogs_common.h"
```

Classes

- class [AutoswitchDetailsDialog](#)

Namespaces

- namespace [Ui](#)

7.6.1 Detailed Description

Defines the [AutoswitchDetailsDialog](#) class, which creates a dialog for changing autoswitch details of an application.

7.7 src/buttonbindingobject.h File Reference

Defines the [ButtonBindingObject](#) class.

```
#include "functionobject.h"
```

Classes

- class [ButtonBindingObject](#)

7.7.1 Detailed Description

Defines the [ButtonBindingObject](#) class.

7.8 src/categorylistmodel.h File Reference

Defines the [CategoryListModel](#) class, subclassed from [QAbstractListModel](#).

```
#include <QObject>
#include <QList>
#include <QAbstractItemModel>
#include "categoryobject.h"
```

Classes

- class [CategoryListModel](#)

7.8.1 Detailed Description

Defines the [CategoryListModel](#) class, subclassed from [QAbstractListModel](#).

7.9 src/categoryobject.h File Reference

Defines the [CategoryObject](#) class.

```
#include "setup_common.h"
```

```
#include "functionobject.h"
```

Classes

- class [CategoryObject](#)

7.9.1 Detailed Description

Defines the [CategoryObject](#) class.

7.10 src/copyapplicationdialog.h File Reference

Defines the [CopyApplicationDialog](#) class, which creates a dialog for copying an application.

```
#include <QtGui/QDialog>
#include "groupobject.h"
```

Classes

- class [CopyApplicationDialog](#)

Namespaces

- namespace [Ui](#)

7.10.1 Detailed Description

Defines the [CopyApplicationDialog](#) class, which creates a dialog for copying an application.

7.11 src/copycategorydialog.h File Reference

Defines the [CopyCategoryDialog](#) class, which creates a dialog for copying a category of functions.

```
#include <QtGui>
#include "groupobject.h"
```

Classes

- class [CopyCategoryDialog](#)

Namespaces

- namespace [Ui](#)

7.11.1 Detailed Description

Defines the [CopyCategoryDialog](#) class, which creates a dialog for copying a category of functions.

7.12 src/copyfunctiondialog.h File Reference

Defines the [CopyFunctionDialog](#) class, which creates a dialog for copying a functions.

```
#include <QtGui/QDialog>
#include "groupobject.h"
```

Classes

- class [CopyFunctionDialog](#)

Namespaces

- namespace [Ui](#)

7.12.1 Detailed Description

Defines the [CopyFunctionDialog](#) class, which creates a dialog for copying a functions.

7.13 src/editapplicationdialog.h File Reference

Defines the [EditApplicationDialog](#) class, which creates a dialog for editing an application.

```
#include <QtGui/QDialog>
#include "groupobject.h"
#include "mydialogs_common.h"
```

Classes

- class [EditApplicationDialog](#)

Namespaces

- namespace [Ui](#)

7.13.1 Detailed Description

Defines the [EditApplicationDialog](#) class, which creates a dialog for editing an application.

7.14 src/editcategorydialog.h File Reference

Defines the [EditCategoryDialog](#) class, which creates a dialog for editing a category of functions.

```
#include <QtGui/QDialog>
#include "mydialogs_common.h"
```

Classes

- class [EditCategoryDialog](#)

Namespaces

- namespace [Ui](#)

7.14.1 Detailed Description

Defines the [EditCategoryDialog](#) class, which creates a dialog for editing a category of functions.

7.15 src/editfunctiondialog.h File Reference

Defines the [EditFunctionDialog](#) class, which creates a dialog for editing a function.

```
#include <QtGui/QDialog>
#include <QKeyEvent>
#include <QList>
#include <QToolButton>
#include "mydialogs_common.h"
#include "keypress_common.h"
#include "keyobject.h"
```

Classes

- class [EditFunctionDialog](#)

Namespaces

- namespace [Ui](#)

7.15.1 Detailed Description

Defines the [EditFunctionDialog](#) class, which creates a dialog for editing a function.

7.16 src/functiondelegate.h File Reference

Defines the [FunctionDelegate](#) class, subclassed from [QItemDelegate](#).

```
#include <QItemDelegate>
```

Classes

- class [FunctionDelegate](#)

7.16.1 Detailed Description

Defines the [FunctionDelegate](#) class, subclassed from [QItemDelegate](#).

7.17 src/functionlistmodel.h File Reference

Defines the [FunctionListModel](#) class, subclassed from [QAbstractListModel](#).

```
#include <QObject>
```

```
#include <QList>
```

```
#include <QAbstractItemModel>
```

```
#include "categoryobject.h"
```

Classes

- class [FunctionListModel](#)

7.17.1 Detailed Description

Defines the [FunctionListModel](#) class, subclassed from [QAbstractListModel](#).

7.18 src/functionobject.h File Reference

Defines the [FunctionObject](#) class.

```
#include "setup_common.h"
```

Classes

- class [FunctionObject](#)

7.18.1 Detailed Description

Defines the [FunctionObject](#) class.

7.19 src/groupobject.h File Reference

Defines the [GroupObject](#) class.

```
#include "setup_common.h"
#include "applicationobject.h"
```

Classes

- class [GroupObject](#)

7.19.1 Detailed Description

Defines the [GroupObject](#) class.

7.20 src/grouporappwidget.h File Reference

Defines the [GroupOrAppWidget](#) class, which creates a header of buttons for choosing a group or application.

```
#include <QWidget>
#include <QMouseEvent>
```

Classes

- class [GroupOrAppWidget](#)

7.20.1 Detailed Description

Defines the [GroupOrAppWidget](#) class, which creates a header of buttons for choosing a group or application.

7.21 src/hardwarethread.h File Reference

Defines the [HardwareThread](#) class.

```
#include <QThread>
#include "groupobject.h"
#include "applicationobject.h"
```

Classes

- class [HardwareThread](#)

Enumerations

- enum `tasks` { `TASK_INIT`, `TASK_CHECK_DIFFERENCES` }

7.21.1 Detailed Description

Defines the [HardwareThread](#) class.

7.22 src/identifiedpushbutton.h File Reference

Defines the [IdentifiedPushButton](#) class, which creates a button widget for choosing a group or application.

```
#include <QPushButton>
```

Classes

- class [IdentifiedPushButton](#)

Variables

- const QString `BUTTON_COLOR_APPLICATION_INACTIVE` = "#162b49"
- const QString `BUTTON_COLOR_APPLICATION_ACTIVE` = "#3a4e6d"
- const QString `BUTTON_COLOR_GROUP_INACTIVE` = "#c9d7ef"
- const QString `BUTTON_COLOR_GROUP_ACTIVE` = "#ffffff"

7.22.1 Detailed Description

Defines the [IdentifiedPushButton](#) class, which creates a button widget for choosing a group or application.

7.23 src/keyobject.h File Reference

Defines the [KeyObject](#) class, for modeling a keypress, with its modifiers and delays.

```
#include <QObject>
#include "keypress_common.h"
```

Classes

- class [KeyObject](#)

7.23.1 Detailed Description

Defines the [KeyObject](#) class, for modeling a keypress, with its modifiers and delays.

7.24 src/keypress_common.h File Reference

Structures, constants and functions which are used by several classes, and which are related to keypresses.

```
#include <QtGui>
```

Classes

- struct [MyKey](#)
- struct [MyKeypressHID](#)
- struct [charToKeyInfo](#)
- struct [stringToKeyInfo](#)
- struct [stringToModifierInfo](#)

Functions

- bool [internalStringFromQtKey](#) (const int key, const bool numPad, QString &internalString)
- bool [internalStringFromQtModifier](#) (const struct [MyKey](#) modifier, QString &internalString)

Variables

- const quint32 **FLAG_DOUBLECLICK** = 0x01
- const quint32 **FLAG_S1** = 0x02
- const quint32 **FLAG_S2** = 0x04
- const quint32 **DEFAULT_DELAY_MS_1** = 0
- const quint32 **DEFAULT_DELAY_MS_2** = 0
- const char **SPECIAL_START** = '['
- const char **SPECIAL_END** = ']'
- const char **DELAY_START** = '{'
- const char **DELAY_END** = '}'
- const char **DELAY_SEPARATOR** = ','
- const char **ESCAPE_CHARACTER** = '\\'
- const char **SIMULTANEOUS_KEYPRESS** = '+'
- const int **HID_1** = 30
- const int **HID_EXCLAM** = 30
- const int **HID_2** = 31
- const int **HID_AT** = 31
- const int **HID_3** = 32
- const int **HID_NUMSIGN** = 32
- const int **HID_4** = 33
- const int **HID_DOLLAR** = 33
- const int **HID_5** = 34
- const int **HID_PERCENT** = 34
- const int **HID_6** = 35
- const int **HID_CIRCUM** = 35
- const int **HID_7** = 36
- const int **HID_AMP** = 36
- const int **HID_8** = 37
- const int **HID_ASTERISK** = 37

- const int **HID_9** = 38
- const int **HID_LPAREN** = 38
- const int **HID_0** = 39
- const int **HID_RPAREN** = 39
- const int **HID_NUM_1** = 89
- const int **HID_NUM_2** = 90
- const int **HID_NUM_3** = 91
- const int **HID_NUM_4** = 92
- const int **HID_NUM_5** = 93
- const int **HID_NUM_6** = 94
- const int **HID_NUM_7** = 95
- const int **HID_NUM_8** = 96
- const int **HID_NUM_9** = 97
- const int **HID_NUM_0** = 98
- const int **HID_RETURN** = 40
- const int **HID_ESCAPE** = 41
- const int **HID_BACKSPACE** = 42
- const int **HID_TAB** = 43
- const int **HID_SPACEBAR** = 44
- const int **HID_MINUS** = 45
- const int **HID_UNDERSCORE** = 45
- const int **HID_PLUS** = 46
- const int **HID_EQ** = 46
- const int **HID_LBRACKET** = 47
- const int **HID_LBRACE** = 47
- const int **HID_RBRACE** = 48
- const int **HID_RBRACKET** = 48
- const int **HID_BACKSLASH** = 49
- const int **HID_BAR** = 49
- const int **HID_COLON** = 51
- const int **HID_SEMICOLON** = 51
- const int **HID_APOSTROPHE** = 52
- const int **HID_DQUOTE** = 52
- const int **HID_GRAVE** = 53
- const int **HID_TILDE** = 53
- const int **HID_COMMA** = 54
- const int **HID_LESS** = 54
- const int **HID_DOT** = 55
- const int **HID_MORE** = 55
- const int **HID_SLASH** = 56
- const int **HID_QUESTION** = 56
- const int **HID_CAPS_LOCK** = 57
- const int **HID_PRINT** = 70
- const int **HID_SCROLL_LOCK** = 71
- const int **HID_PAUSE** = 72
- const int **HID_INSERT** = 73
- const int **HID_HOME** = 74
- const int **HID_PAGEUP** = 75
- const int **HID_DELETE** = 76
- const int **HID_END** = 77

- const int **HID_PAGEDOWN** = 78
- const int **HID_RIGHT** = 79
- const int **HID_LEFT** = 80
- const int **HID_DOWN** = 81
- const int **HID_UP** = 82
- const int **HID_NUMLOCK** = 83
- const int **HID_NUM_DIVIDE** = 84
- const int **HID_MULTIPLY** = 85
- const int **HID_NUM_MULTIPLY** = 85
- const int **HID_NUM_MINUS** = 86
- const int **HID_NUM_PLUS** = 87
- const int **HID_ENTER** = 88
- const int **HID_A** = 4
- const int **HID_B** = 5
- const int **HID_C** = 6
- const int **HID_D** = 7
- const int **HID_E** = 8
- const int **HID_F** = 9
- const int **HID_G** = 10
- const int **HID_H** = 11
- const int **HID_I** = 12
- const int **HID_J** = 13
- const int **HID_K** = 14
- const int **HID_L** = 15
- const int **HID_M** = 16
- const int **HID_N** = 17
- const int **HID_O** = 18
- const int **HID_P** = 19
- const int **HID_Q** = 20
- const int **HID_R** = 21
- const int **HID_S** = 22
- const int **HID_T** = 23
- const int **HID_U** = 24
- const int **HID_V** = 25
- const int **HID_W** = 26
- const int **HID_X** = 27
- const int **HID_Y** = 28
- const int **HID_Z** = 29
- const int **HID_F1** = 0x3A
- const int **HID_F2** = 0x3B
- const int **HID_F3** = 0x3C
- const int **HID_F4** = 0x3D
- const int **HID_F5** = 0x3E
- const int **HID_F6** = 0x3F
- const int **HID_F7** = 0x40
- const int **HID_F8** = 0x41
- const int **HID_F9** = 0x42
- const int **HID_F10** = 0x43
- const int **HID_F11** = 0x44
- const int **HID_F12** = 0x45

- const int **HID_SB** = HID_SPACEBAR
- const int **HID_MN** = HID_MINUS
- const int **HID_SL** = HID_SLASH
- const int **HID_BS** = HID_BACKSLASH
- const int **SPECIAL_HID_MOUSE_BACK** = 0xED
- const int **SPECIAL_HID_MOUSE_FORWARD** = 0xEE
- const int **SPECIAL_HID_MOUSE_L1** = 0xE8
- const int **SPECIAL_HID_MOUSE_MAP** = 0xEF
- const int **SPECIAL_HID_MOUSE_R1** = 0xE9
- const int **SPECIAL_HID_MOUSE_WH_DN** = 0xEC
- const int **SPECIAL_HID_MOUSE_WH_PR** = 0xEB
- const int **SPECIAL_HID_MOUSE_WH_UP** = 0xEA
- const int **SPECIAL_HID_ALTSTICK** = 0xF0
- const int **SPECIAL_HID_MODE_CHANGE** = 0xF1
- const int **SPECIAL_HID_CPI_LOW** = 0xF2
- const int **SPECIAL_HID_CPI_DEFAULT** = 0xF3
- const int **SPECIAL_HID_CPI_HIGH** = 0xF4
- const char **SPECIAL_KEY_STRING_LEFT_CONTROL** [] = "L_CTRL"
- const char **SPECIAL_KEY_STRING_RIGHT_CONTROL** [] = "R_CTRL"
- const char **SPECIAL_KEY_STRING_LEFT_SHIFT** [] = "L_SHIFT"
- const char **SPECIAL_KEY_STRING_RIGHT_SHIFT** [] = "R_SHIFT"
- const char **SPECIAL_KEY_STRING_LEFT_ALT** [] = "L_ALT"
- const char **SPECIAL_KEY_STRING_RIGHT_ALT** [] = "R_ALT"
- const char **SPECIAL_KEY_STRING_LEFT_GUI** [] = "L_GUI"
- const char **SPECIAL_KEY_STRING_RIGHT_GUI** [] = "R_GUI"
- const char **SPECIAL_KEY_STRING_AMPERSAND** [] = "AMP"
- const char **SPECIAL_KEY_STRING_BACKSPACE** [] = "BACKSPACE"
- const char **SPECIAL_KEY_STRING_RETURN** [] = "RETURN"
- const char **SPECIAL_KEY_STRING_DIVIDE** [] = "DIVIDE"
- const char **SPECIAL_KEY_STRING_DOWN** [] = "DOWN"
- const char **SPECIAL_KEY_STRING_ENTER** [] = "ENTER"
- const char **SPECIAL_KEY_STRING_END** [] = "END"
- const char **SPECIAL_KEY_STRING_HOME** [] = "HOME"
- const char **SPECIAL_KEY_STRING_INSERT** [] = "INS"
- const char **SPECIAL_KEY_STRING_CAPS_LOCK** [] = "CAPSLK"
- const char **SPECIAL_KEY_STRING_F1** [] = "F1"
- const char **SPECIAL_KEY_STRING_F2** [] = "F2"
- const char **SPECIAL_KEY_STRING_F3** [] = "F3"
- const char **SPECIAL_KEY_STRING_F4** [] = "F4"
- const char **SPECIAL_KEY_STRING_F5** [] = "F5"
- const char **SPECIAL_KEY_STRING_F6** [] = "F6"
- const char **SPECIAL_KEY_STRING_F7** [] = "F7"
- const char **SPECIAL_KEY_STRING_F8** [] = "F8"
- const char **SPECIAL_KEY_STRING_F9** [] = "F9"
- const char **SPECIAL_KEY_STRING_F10** [] = "F10"
- const char **SPECIAL_KEY_STRING_F11** [] = "F11"
- const char **SPECIAL_KEY_STRING_F12** [] = "F12"
- const char **SPECIAL_KEY_STRING_ESCAPE** [] = "ESC"
- const char **SPECIAL_KEY_STRING_DELETE** [] = "DEL"
- const char **SPECIAL_KEY_STRING_LEFT** [] = "LEFT"

- const char **SPECIAL_KEY_STRING_RIGHT** [] = "RIGHT"
- const char **SPECIAL_KEY_STRING_SPACE** [] = "SPACE"
- const char **SPECIAL_KEY_STRING_TAB** [] = "TAB"
- const char **SPECIAL_KEY_STRING_UP** [] = "UP"
- const char **SPECIAL_KEY_STRING_MINUS** [] = "MINUS"
- const char **SPECIAL_KEY_STRING_PLUS** [] = "PLUS"
- const char **SPECIAL_KEY_STRING_NUM_0** [] = "NUM_0"
- const char **SPECIAL_KEY_STRING_NUM_1** [] = "NUM_1"
- const char **SPECIAL_KEY_STRING_NUM_2** [] = "NUM_2"
- const char **SPECIAL_KEY_STRING_NUM_3** [] = "NUM_3"
- const char **SPECIAL_KEY_STRING_NUM_4** [] = "NUM_4"
- const char **SPECIAL_KEY_STRING_NUM_5** [] = "NUM_5"
- const char **SPECIAL_KEY_STRING_NUM_6** [] = "NUM_6"
- const char **SPECIAL_KEY_STRING_NUM_7** [] = "NUM_7"
- const char **SPECIAL_KEY_STRING_NUM_8** [] = "NUM_8"
- const char **SPECIAL_KEY_STRING_NUM_9** [] = "NUM_9"
- const char **SPECIAL_KEY_STRING_NUM_DIVIDE** [] = "NUM_DIVIDE"
- const char **SPECIAL_KEY_STRING_NUM_PLUS** [] = "NUM_PLUS"
- const char **SPECIAL_KEY_STRING_NUM_MINUS** [] = "NUM_MINUS"
- const char **SPECIAL_KEY_STRING_NUM_MULTIPLY** [] = "NUM_MULTIPLY"
- const char **SPECIAL_KEY_STRING_MULTIPLY** [] = "MULTIPLY"
- const char **SPECIAL_KEY_STRING_PAUSE** [] = "PAUSE"
- const char **SPECIAL_KEY_STRING_PAGEDOWN** [] = "PG_DN"
- const char **SPECIAL_KEY_STRING_PAGEUP** [] = "PG_UP"
- const char **SPECIAL_KEY_STRING_PRINT** [] = "PRINT"
- const char **SPECIAL_KEY_STRING_SCROLL_LOCK** [] = "SCRLK"
- const char **SPECIAL_KEY_STRING_SEMICOLON** [] = "SEMICOLON"
- const char **SPECIAL_KEY_STRING_BRACKET_LEFT** [] = "LBRACKET"
- const char **SPECIAL_KEY_STRING_BRACKET_RIGHT** [] = "RBRACKET"
- const char **SPECIAL_KEY_STRING_NUMLOCK** [] = "NUMLOCK"
- const char **SPECIAL_KEY_STRING_MOUSE_BACK** [] = "MOUSE_BACK"
- const char **SPECIAL_KEY_STRING_MOUSE_FORWARD** [] = "MOUSE_FORWARD"
- const char **SPECIAL_KEY_STRING_MOUSE_L1** [] = "MOUSE_L1"
- const char **SPECIAL_KEY_STRING_MOUSE_MAP** [] = "MOUSE_MAP"
- const char **SPECIAL_KEY_STRING_MOUSE_R1** [] = "MOUSE_R1"
- const char **SPECIAL_KEY_STRING_MOUSE_WH_DN** [] = "MOUSE_WH_DN"
- const char **SPECIAL_KEY_STRING_MOUSE_WH_PR** [] = "MOUSE_WH_PR"
- const char **SPECIAL_KEY_STRING_MOUSE_WH_UP** [] = "MOUSE_WH_UP"
- const char **SPECIAL_KEY_STRING_ALTSTICK** [] = "ALTSTICK"
- const char **SPECIAL_KEY_STRING_CHANGE_MODE** [] = "CH_MODE"
- const char **SPECIAL_KEY_STRING_CPI_LOW** [] = "CPI_LOW"
- const char **SPECIAL_KEY_STRING_CPI_DEFAULT** [] = "CPI_DEFAULT"
- const char **SPECIAL_KEY_STRING_CPI_HIGH** [] = "CPI_HIGH"
- struct [charToKeyInfo](#) **CHAR_TO_KEY_HID** []
- struct [stringToKeyInfo](#) **STRING_TO_KEY_HID** []
- struct [stringToModifierInfo](#) **STRING_TO_MODIFIER_HID** []

7.24.1 Detailed Description

Structures, constants and functions which are used by several classes, and which are related to keypresses.

7.24.2 Function Documentation

7.24.2.1 `bool internalStringFromQtKey (const int key, const bool numPad, QString & internalString)`

Gets the internal string representation of Qt's representation of key

Parameters

key Qt representation of key

numPad True if this is numpad key, false otherwise

internalString Reference to where the string representation is placed

Returns

true if converted ok, or false if failed to convert

7.24.2.2 `bool internalStringFromQtModifier (const struct MyKey modifier, QString & internalString)`

Gets the internal string representation of Qt's representation of modifier

Parameters

modifier `MyKey` structure that contains Qt representation of modifier

internalString Reference to where the string representation is placed

Returns

true if converted ok, or false if failed to convert

7.24.3 Variable Documentation

7.24.3.1 `struct stringToModifierInfo STRING_TO_MODIFIER_HID[]`

Initial value:

```
{
    {SPECIAL_KEY_STRING_LEFT_ALT, "Left Alt", Qt::Key_Alt, false, 0, 0x04},
    {SPECIAL_KEY_STRING_LEFT_CONTROL, "Left Ctrl", Qt::Key_Control, false, 0, 0x01},
    {SPECIAL_KEY_STRING_LEFT_GUI, "Left Windows", Qt::Key_Meta, false, 0, 0x08},
    {SPECIAL_KEY_STRING_LEFT_SHIFT, "Left Shift", Qt::Key_Shift, false, 0, 0x02},
    {SPECIAL_KEY_STRING_RIGHT_ALT, "Right Alt", Qt::Key_AltGr, true, 0, 0x40},
    {SPECIAL_KEY_STRING_RIGHT_CONTROL, "Right Ctrl", Qt::Key_Control, true, 285, 0x10},
    {SPECIAL_KEY_STRING_RIGHT_GUI, "Right Windows", Qt::Key_Meta, true, 348, 0x80},
    {SPECIAL_KEY_STRING_RIGHT_SHIFT, "Right Shift", Qt::Key_Shift, true, 54, 0x20}
}
```

7.25 src/keypressparser.h File Reference

Parser class for the keypress sequence format used by the xml-formatted setup file.

```
#include "keypress_common.h"  
#include "keyobject.h"
```

Classes

- class [KeypressParser](#)

7.25.1 Detailed Description

Parser class for the keypress sequence format used by the xml-formatted setup file. The parser is a singleton object.

7.26 src/macrofunctionsdialog.h File Reference

Defines the [MacroFunctionsDialog](#) class, which creates a dialog for displaying the supported special keys and functions in macros.

```
#include <QDialog>
```

Classes

- class [MacroFunctionsDialog](#)

Namespaces

- namespace [Ui](#)

7.26.1 Detailed Description

Defines the [MacroFunctionsDialog](#) class, which creates a dialog for displaying the supported special keys and functions in macros.

7.27 src/mainwindow.h File Reference

Defines the [MainWindow](#) class, which creates the advanced view.

```
#include <QtGui>  
#include <QList>  
#include <QListView>  
#include <QPainter>  
#include <QtSvg>
```

```
#include "setupparser.h"
#include "identifiedpushbutton.h"
#include "mouseobject.h"
#include "keypress_common.h"
#include "hardwarethread.h"
#include "mapwidget.h"
#include "cairo/cairo.h"
#include "cairo/cairo-pdf.h"
```

Classes

- class [MainWindow](#)

Namespaces

- namespace [Ui](#)

Enumerations

- enum `objectType` { `TYPE_GROUP`, `TYPE_APPLICATION`, `TYPE_CATEGORY`, `TYPE_FUNCTION` }

7.27.1 Detailed Description

Defines the [MainWindow](#) class, which creates the advanced view.

7.28 src/mapwidget.h File Reference

Defines the [MapWidget](#) class, which creates a widget for showing mode map.

```
#include <QWidget>
#include <QPixmap>
```

Classes

- class [MapWidget](#)

7.28.1 Detailed Description

Defines the [MapWidget](#) class, which creates a widget for showing mode map.

7.29 src/mousebuttonobject.h File Reference

Defines the [MouseButtonObject](#) class.

```
#include "applicationobject.h"  
#include "buttonbindingobject.h"
```

Classes

- class [MouseButtonObject](#)

Variables

- const int `MOUSE_BUTTON_NAME_MAX_LEN` = 8

7.29.1 Detailed Description

Defines the [MouseButtonObject](#) class.

7.30 src/mousehardware.h File Reference

Defines the [MouseHardware](#) class.

```
#include "setup_common.h"
```

Classes

- struct [PROFILE_INFO_BLOCK](#)
- class [MouseHardware](#)

Defines

- #define `BitMsk(bit)` (1 << (bit))
- #define `BitNtst(arg, bit)` bool((arg) & BitMsk(bit))
- #define `BitNset(arg, bit)` ((arg) |= BitMsk(bit))
- #define `VID` 0x03EB
- #define `PID` 0x2022
- #define `TIME_OUT` 100
- #define `WM_OK` 0
- #define `WM_BAD_PATH` 1
- #define `WM_DLL_ERROR` 2
- #define `WM_COULD_NOT_OPEN_DEVICE` 3
- #define `WM_NO_USB_DEVICE_CAPABILITIES` 4
- #define `WM_ERROR` 5
- #define `WM_FAIL` 6
- #define `WM_NO_DATA_AVAILABLE` 7
- #define `CMD_REV` 0x01

- #define **CMD_ERASE_FLASH** 0x02
- #define **CMD_ERASE_4KBLOCK** 0x03
- #define **CMD_PROFILE_INFO** 0x04
- #define **CMD_PROFILE_INFO_READ** 0x05
- #define **CMD_SET_PROFILE** 0x06
- #define **CMD_START_BOOTLOADER** 0x07
- #define **CMD_PROFILE_DATA** 0x08
- #define **CMD_DONE** 0x09
- #define **CMD_READ_4K** 0x0A
- #define **CMD_READ_STATISTICS** 0x0B
- #define **CMD_POLL_PROFILE_STATE** 0x0C
- #define **CMD_GET_BLOCK_CRC** 0x0D
- #define **CMD_GET_PROFILE_INFO_BLOCK** 0x0F
- #define **CMD_GET_JOY_XY** 0x10
- #define **CMD_GET_PROFILE** 0x11

Variables

- const int **CMD_RESPONSE_TIMEOUT_MS** = 2000
- const int **CMD_RESPONSE_CHECK_INTERVAL_MS** = 1
- const quint8 **THUMBPAD_MODE_DISABLED** = 0
- const quint8 **THUMBPAD_MODE_DIRECTIONAL_8KEY** = 1
- const quint8 **THUMBPAD_MODE_DIRECTIONAL_16KEY** = 2
- const quint8 **THUMBPAD_MODE_ANALOG_JOYSTICK** = 3
- const quint8 **THUMBPAD_MODE_ANALOG_JOYSTICK_NOBUTTONS** = 4
- const quint8 **THUMBPAD_MODE_INDEPENDENT_4KEY** = 5
- const quint8 **THUMBPAD_MODE_INDEPENDENT_8KEY** = 6
- const unsigned int **NUM_BUTTONS** = 27
- const unsigned int **NUM_DOUBLECLICK_BUTTONS** = 17
- const unsigned int **NUM_UNPROGRAMMABLE_BUTTONS** = 2

7.30.1 Detailed Description

Defines the [MouseHardware](#) class. The mouse hardware is a singleton object.

7.31 src/mouseobject.h File Reference

Defines the [MouseObject](#) class.

```
#include <QObject>
#include "mousebuttonobject.h"
#include "mousehardware.h"
#include "setup_common.h"
#include "keypress_common.h"
#include "groupobject.h"
```

Classes

- struct [mouseButtonRepresentation](#)
- struct [mouseButtonWithFlags](#)
- class [MouseObject](#)

Enumerations

- enum `cmdResult` { `CMD_OK`, `CMD_DISCONNECTED`, `CMD_ERROR` }
- enum `mouseState` { `MOUSE_OBJECT_NOT_INITIALIZED`, `MOUSE_HARDWARE_NOT_FOUND`, `MOUSE_HARDWARE_UNINITIALIZED`, `MOUSE_READY` }

Variables

- const QString `software_version` = "1.0.1"
- const int `CHOICE_EVENT` = -1
- const int `DISCONNECTED_EVENT` = -2
- const int `MAP_EVENT` = -3
- struct [mouseButtonRepresentation](#) `representation` []

7.31.1 Detailed Description

Defines the [MouseObject](#) class. The mouse is a singleton object.

7.31.2 Variable Documentation

7.31.2.1 struct `mouseButtonRepresentation` `representation`[]

Initial value:

```
{
    {"A1", "A1", A1}, {"A2", "A2", A2}, {"A3", "A3", A3}, {"A4", "A4", A4},
    {"A5", "A5", A5}, {"A6", "A6", A6},
    {"A7", "A7", A7}, {"B1", "B1", B1}, {"B2", "B2", B2}, {"B3", "B3", B3},
    {"B4", "B4", B4}, {"B5", "B5", B5},
    {"B6", "B6", B6}, {"B7", "B7", B7}, {"T1", "T1", T1}, {"T2", "T2", T2},
    {"W_PR", "S3", W_PR}, {"W_UP", "S1", W_UP}, {"W_DN", "S2", W_DN},
    {"J1", "J1", J1}, {"J2", "J2", J2},
    {"J3", "J3", J3}, {"J4", "J4", J4},
    {"J5", "J5", J5}, {"J6", "J6", J6},
    {"J7", "J7", J7}, {"J8", "J8", J8}, {"L1", "L1", L1}, {"R1", "R1", R1}
}
```

7.32 `src/mousewidget.h` File Reference

Defines the [MouseWidget](#) class, which creates a widget for showing the mouse graphic in advanced view.

```
#include <QWidget>
#include <QMouseEvent>
#include <QSvgRenderer>
```

```
#include "mouseobject.h"
```

Classes

- class [MouseWidget](#)

7.32.1 Detailed Description

Defines the [MouseWidget](#) class, which creates a widget for showing the mouse graphic in advanced view.

7.33 src/myallstatisticstable.h File Reference

Defines the [MyAllStatisticsTable](#) class, which creates a table view for displaying statistics for all modes.

```
#include <QTableView>
```

Classes

- class [MyAllStatisticsTable](#)

7.33.1 Detailed Description

Defines the [MyAllStatisticsTable](#) class, which creates a table view for displaying statistics for all modes.

7.34 src/mydialogs_common.h File Reference

Structures, constants and functions which are used by several dialog classes.

```
#include "groupobject.h"
```

Enumerations

- enum `action` { `MODIFY_ACTION`, `REMOVE_ACTION`, `CANCEL_ACTION`, `COPY_ACTION` }

Functions

- unsigned int [getFreeProfileNumber](#) (const QList< [GroupObject](#) > &groupList)
- bool [caseInsensitiveLessThan](#) (const QString &s1, const QString &s2)

7.34.1 Detailed Description

Structures, constants and functions which are used by several dialog classes.

7.34.2 Function Documentation

7.34.2.1 `bool caseInsensitiveLessThan (const QString & s1, const QString & s2)`

Case-insensitive comparison function for two strings

Used in `qSort()`

Parameters

- s1* First string to compare
- s2* Second string to compare

Returns

true if $s1 < s2$, or false otherwise

7.34.2.2 `unsigned int getFreeProfileNumber (const QList< GroupObject > & groupList)`

Get the lowest, free profile number or 0 if there is none

Parameters

groupList The MainWindow's `groupList`, based on which we look for free profile number

Returns

Lowest free profile number, or 0 if there is none

7.35 `src/mylistview.h` File Reference

Defines the [MyListView](#) class, which has some custom features that default list views don't support.

```
#include <QListView>
#include <QWidget>
```

Classes

- class [MyListView](#)

7.35.1 Detailed Description

Defines the [MyListView](#) class, which has some custom features that default list views don't support.

7.36 `src/mystaticstable.h` File Reference

Defines the [MyStatisticsTable](#) class, which creates a table view for displaying statistics for a specific mode.

```
#include <QTableView>
```

Classes

- class [MyStatisticsTable](#)

7.36.1 Detailed Description

Defines the [MyStatisticsTable](#) class, which creates a table view for displaying statistics for a specific mode.

7.37 src/mytoolbutton.h File Reference

Defines the [MyToolButton](#) class, which is a custom tool button for navigation tab.

```
#include <QToolButton>
```

Classes

- class [MyToolButton](#)

Enumerations

- enum [MyToolButtonType](#) {
 MY_TOOLBUTTON_TYPE_NONE, MY_TOOLBUTTON_TYPE_GROUP_LEFTEDGE,
 MY_TOOLBUTTON_TYPE_GROUP_LEFT, MY_TOOLBUTTON_TYPE_GROUP_-
 RIGHT,
 MY_TOOLBUTTON_TYPE_GROUP_RIGHTEDGE, MY_TOOLBUTTON_TYPE_APP_-
 LEFTEDGE, MY_TOOLBUTTON_TYPE_APP_LEFT, MY_TOOLBUTTON_TYPE_-
 APP_RIGHT,
 MY_TOOLBUTTON_TYPE_APP_RIGHTEDGE }

7.37.1 Detailed Description

Defines the [MyToolButton](#) class, which is a custom tool button for navigation tab.

7.38 src/setup_common.h File Reference

Structures, constants and functions which are used by several classes, and that are related to the setup file.

```
#include <QDir>  
#include <QIODevice>  
#include <QDomDocument>  
#include <QObject>  
#include <QMap>  
#include <QtGlobal>
```

Classes

- struct [mouseButtonAction](#)
- struct [statisticFunctionData](#)

Enumerations

- enum [functionType](#) {
NONE, KEYPRESS, MACRO, KEY,
MOUSEBUTTON }

Different types of events for handling the function.

- enum [mouseButtons](#) {
A1, A2, A3, A4,
A5, A6, A7, B1,
B2, B3, B4, B5,
B6, B7, T1, T2,
W_PR, W_UP, W_DN, J1,
J2, J3, J4, J5,
J6, J7, J8, L1,
R1 }

Functions

- bool [getChildByName](#) (const QDomNode parent, QDomNode &child, const QString tagName, const QString nameAttributeValue="")
- bool [getGrandChildByName](#) (const QDomNode parent, QDomNode &grandChild, const QString parentTagName, const QString childTagName, const QString nameAttributeValue="")
- bool [hasChildrenWithAttribute](#) (QDomNode parent, const QString tagName, const QString attribute, const QString value)
- void [deleteChildrenByAttribute](#) (QDomNode parent, const QString tagName, const QString attribute, const QString value)
- void [giveParseError](#) (bool isBindingsFile, const QString details="")
- bool [caseInsensitiveStatisticFunctionDataLessThan](#) (const struct [statisticFunctionData](#) &s1, const struct [statisticFunctionData](#) &s2)

Variables

- const quint8 **MOUSE_BUTTON_LEFT_CLICK = 1**
- const quint8 **MOUSE_BUTTON_RIGHT_CLICK = 2**
- const quint8 **MOUSE_BUTTON_SCROLL_WHEEL_UP = 3**
- const quint8 **MOUSE_BUTTON_SCROLL_WHEEL_PRESS = 4**
- const quint8 **MOUSE_BUTTON_SCROLL_WHEEL_DOWN = 5**
- const quint8 **MOUSE_BUTTON_BACK_CLICK = 6**
- const quint8 **MOUSE_BUTTON_FORWARD_CLICK = 7**
- const quint8 **SPECIAL_BUTTON_ALTSTICK = 8**
- const quint8 **SPECIAL_BUTTON_MAP = 9**

- const QString **CONFIG_FILENAME** = "setup.xml"
- const QString **CONFIG_BACKUP_FILENAME** = "setup_backup.xml"
- const QString **CONFIG_DEFAULT_FILENAME** = "setup_default.xml"
- const QString **BINDINGS_FILENAME** = "bindings.xml"
- const QString **BINDINGS_BACKUP_FILENAME** = "bindings_backup.xml"
- const QString **BINDINGS_DEFAULT_FILENAME** = "bindings_default.xml"
- const QString **STATISTICS_FILENAME** = "statistics.xml"
- const [mouseButtonAction](#) **mouseButtonActions** []
- const char **JOYSTICK_MODE_TEXT_NO_JOYSTICK** [] = "No joystick"
- const char **JOYSTICK_MODE_TEXT_8DIR** [] = "Joystick as 8 directional keys"
- const char **JOYSTICK_MODE_TEXT_16DIR** [] = "Joystick as 16 directional keys"
- const char **JOYSTICK_MODE_TEXT_4IND** [] = "Joystick as 4 assignable buttons"
- const char **JOYSTICK_MODE_TEXT_8IND** [] = "Joystick as 8 assignable buttons"
- const char **JOYSTICK_MODE_TEXT_ANALOG_7_BUTTONS** [] = "Analog joystick, 7 joystick buttons"
- const char **JOYSTICK_MODE_TEXT_ANALOG_NO_BUTTONS** [] = "Analog joystick, no joystick buttons"
- const quint8 **JOYSTICK_MODE_NUM_NO_JOYSTICK** = 0
- const quint8 **JOYSTICK_MODE_NUM_8DIR** = 1
- const quint8 **JOYSTICK_MODE_NUM_16DIR** = 2
- const quint8 **JOYSTICK_MODE_NUM_ANALOG_7_BUTTONS** = 3
- const quint8 **JOYSTICK_MODE_NUM_ANALOG_NO_BUTTONS** = 4
- const quint8 **JOYSTICK_MODE_NUM_4IND** = 5
- const quint8 **JOYSTICK_MODE_NUM_8IND** = 6
- const char *const **joystickModeTexts** []
- const unsigned int **mapVerticalSize** = 720
- const float **pdf_y_offset** = 0.15
- const float **pdf_rectangles** [][][4]
- const float **pdf_rectangles_landscape** [][][4]
- const unsigned int **NUM_PROFILES** = 64
- const unsigned int **NUM_BLOCKS** = 128
- const quint8 **DEFAULT_CPI** = (quint8)((2100 - 100) / 25)
- const quint8 **MAXIMUM_CPI** = (quint8)((5600 - 100) / 25)
- const quint8 **DEFAULT_CPI_LOW** = (quint8)((400 - 100) / 25)
- const quint8 **DEFAULT_CPI_HIGH** = (quint8)((3200 - 100) / 25)
- const quint8 **DEFAULT_JOYSTICK_MODE** = 4
- const quint8 **MAXIMUM_JOYSTICK_MODE** = 6
- const quint8 **MINIMUM_SCROLL_WHEEL_LINES** = 1
- const quint8 **DEFAULT_SCROLL_WHEEL_LINES** = 3
- const quint8 **MAXIMUM_SCROLL_WHEEL_LINES** = 100
- const quint8 **MINIMUM_SENSITIVITY** = 1
- const quint8 **DEFAULT_SENSITIVITY** = 10
- const quint8 **MAXIMUM_SENSITIVITY** = 20
- const quint8 **MINIMUM_DOUBLE_CLICK_SPEED** = 1
- const quint8 **DEFAULT_DOUBLE_CLICK_SPEED** = 5
- const quint8 **MAXIMUM_DOUBLE_CLICK_SPEED** = 255
- const QString **SETUP_ROOT_TAG** = "warmouse"
Setup file's root tag.
- const QString **SETUP_NODE_FUNCTION_LIST_TAG** = "functionList"

Setup file's function list tag.

- const QString `SETUP_NODE_FUNCTION_TAG` = "function"
Setup file's function tag.
- const QString `SETUP_NODE_STRUCTURE_TAG` = "structure"
Setup file's structure tag.
- const QString `SETUP_NODE_GROUP_TAG` = "group"
Setup file's group tag.
- const QString `SETUP_NODE_APPLICATION_TAG` = "app"
Setup file's application tag.
- const QString `SETUP_NODE_CATEGORY_TAG` = "category"
Setup file's category tag.
- const QString `SETUP_NODE_FUNCTION_REFERENCE_TAG` = "functionRef"
Setup file's function reference tag.
- const QString `SETUP_NODE_SETTINGS_TAG` = "settings"
Setup file's settings tag.
- const QString `SETUP_NODE_SETTING_TAG` = "setting"
Setup file's setting tag.
- const QString `SETUP_NODE_FUNCTION_TYPE_ATTRIBUTE` = "type"
Setup file's function tag's type-attribute.
- const QString `SETUP_NODE_FUNCTION_NAME_ATTRIBUTE` = "name"
Setup file's function tag's name-attribute.
- const QString `SETUP_NODE_FUNCTION_TYPE_NONE` = "none"
Setup file's function tag's type-attribute's value NONE.
- const QString `SETUP_NODE_FUNCTION_TYPE_KEYPRESS` = "keypress"
Setup file's function tag's type-attribute's value KEYPRESS.
- const QString `SETUP_NODE_FUNCTION_TYPE_MACRO` = "macro"
Setup file's function tag's type-attribute's value MACRO.
- const QString `SETUP_NODE_FUNCTION_TYPE_KEY` = "key"
Setup file's function tag's type-attribute's value KEY.
- const QString `SETUP_NODE_FUNCTION_TYPE_MOUSEBUTTON` = "mbutton"
Setup file's function tag's type-attribute's value MOUSEBUTTON.
- const QString `SETUP_NODE_GROUP_NAME_ATTRIBUTE` = "name"
Setup file's group tag's name-attribute.

- const QString `SETUP_NODE_APPLICATION_NAME_ATTRIBUTE` = "name"
Setup file's application tag's name-attribute.
- const QString `SETUP_NODE_APPLICATION_PROFILE_NUMBER_ATTRIBUTE` = "pnum"
Setup file's application tag's profile number attribute.
- const QString `SETUP_NODE_CATEGORY_NAME_ATTRIBUTE` = "name"
Setup file's category tag's name-attribute.
- const QString `SETUP_NODE_FUNCTION_REFERENCE_NAME_ATTRIBUTE` = "name"
Setup file's function reference tag's name-attribute.
- const QString `SETUP_NODE_APPLICATION_FILE_NAME_ATTRIBUTE` = "fname"
Setup file's application tag's file name attribute.
- const QString `SETUP_NODE_APPLICATION_TITLE_PART_ATTRIBUTE` = "tpart"
Setup file's application tag's window title part attribute.
- const QString `SETUP_NODE_APPLICATION_AUTOSWITCH_DISABLED_ATTRIBUTE` = "noauto"
Setup file's application tag's autoswitch disabled attribute.
- const QString `SETUP_NODE_APPLICATION_CPI_ATTRIBUTE` = "cpi"
Setup file's application tag's CPI attribute.
- const QString `SETUP_NODE_APPLICATION_CPI_LOW_ATTRIBUTE` = "cpi_l"
Setup file's application tag's CPI-low attribute.
- const QString `SETUP_NODE_APPLICATION_CPI_HIGH_ATTRIBUTE` = "cpi_h"
Setup file's application tag's CPI-high attribute.
- const QString `SETUP_NODE_APPLICATION_JOYSTICK_MODE_ATTRIBUTE` = "joystick"
Setup file's application tag's joystick mode attribute.
- const QString `SETUP_NODE_APPLICATION_SENSITIVITY_ATTRIBUTE` = "sens"
Setup file's application tag's sensitivity attribute.
- const QString `SETUP_NODE_APPLICATION_SCROLL_WHEEL_LINES_ATTRIBUTE` = "swlines"
Setup file's application tag's scroll wheel lines attribute.
- const QString `SETUP_NODE_APPLICATION_DOUBLE_CLICK_SPEED_ATTRIBUTE` = "dc-speed"
Setup file's application tag's double click speed attribute.
- const QString `SETUP_NODE_SETTING_NAME_ATTRIBUTE` = "name"
Setup file's setting tag's name-attribute.
- const QString `BINDINGS_ROOT_TAG` = "warmouse"
Bindings file's root tag.

- const QString `BINDINGS_NODE_BINDINGS_TAG` = "bindings"
Bindings file's bindings tag.
- const QString `BINDINGS_NODE_APPLICATION_TAG` = "app"
Bindings file's application tag.
- const QString `BINDINGS_NODE_BINDING_TAG` = "binding"
Bindings file's binding tag.
- const QString `BINDINGS_NODE_APPLICATION_NAME_ATTRIBUTE` = "name"
Bindings file's application tag's name-attribute.
- const QString `BINDINGS_NODE_FUNCTION_ATTRIBUTE` = "function"
Bindings file's binding tag's function-attribute.
- const QString `BINDINGS_NODE_BUTTON_ATTRIBUTE` = "button"
Bindings file's binding tag's button-attribute.
- const QString `BINDINGS_NODE_S1_ATTRIBUTE` = "S1"
Bindings file's binding tag's S1-toggle attribute.
- const QString `BINDINGS_NODE_S2_ATTRIBUTE` = "S2"
Bindings file's binding tag's S2-toggle attribute.
- const QString `BINDINGS_NODE_DOUBLECLICK_ATTRIBUTE` = "dc"
Bindings file's binding tag's double-click-toggle attribute.
- const QString `BINDINGS_TOGGLE_INACTIVE_VALUE` = "0"
Bindings file's toggles have this value, if the toggle is not on:
- const QString `BINDINGS_TOGGLE_ACTIVE_VALUE` = "1"
Bindings file's toggles have this value, if the toggle is on:
- const QString `EXPORT_ROOT_TAG` = "warmouse"
Bindings file's root tag.
- const QString `EXPORT_NODE_EXPORT_TAG` = "export"
Export file's export tag.
- const QString `EXPORT_NODE_APPLICATION_TAG` = "app"
Export file's application tag.
- const QString `EXPORT_NODE_BINDINGS_TAG` = "bindings"
Export file's bindings tag.
- const QString `EXPORT_NODE_APPLICATION_NAME_ATTRIBUTE` = "name"
Export file's application tag's name-attribute.
- const QString `EXPORT_NODE_APPLICATION_GROUP_ATTRIBUTE` = "group"

Export file's application tag's group-attribute.

- const QString `EXPORT_NODE_APPLICATION_FILE_NAME_ATTRIBUTE` = "fname"
Export file's application tag's file name attribute.
- const QString `EXPORT_NODE_APPLICATION_TITLE_PART_ATTRIBUTE` = "tpart"
Export file's application tag's window title part attribute.
- const QString `EXPORT_NODE_APPLICATION_AUTOSWITCH_DISABLED_ATTRIBUTE` = "noauto"
Export file's application tag's autoswitch disabled attribute.
- const QString `EXPORT_NODE_APPLICATION_CPI_ATTRIBUTE` = "cpi"
Export file's application tag's CPI attribute.
- const QString `EXPORT_NODE_APPLICATION_CPI_LOW_ATTRIBUTE` = "cpi_l"
Export file's application tag's CPI-low attribute.
- const QString `EXPORT_NODE_APPLICATION_CPI_HIGH_ATTRIBUTE` = "cpi_h"
Export file's application tag's CPI-high attribute.
- const QString `EXPORT_NODE_APPLICATION_JOYSTICK_MODE_ATTRIBUTE` = "joystick"
Export file's application tag's joystick mode attribute.
- const QString `EXPORT_NODE_APPLICATION_SENSITIVITY_ATTRIBUTE` = "sens"
Export file's application tag's sensitivity attribute.
- const QString `EXPORT_NODE_APPLICATION_SCROLL_WHEEL_LINES_ATTRIBUTE` = "swlines"
Export file's application tag's scroll wheel lines attribute.
- const QString `EXPORT_NODE_APPLICATION_DOUBLE_CLICK_SPEED_ATTRIBUTE` = "dc-speed"
Export file's application tag's double click speed attribute.
- const QString `STATISTICS_ROOT_TAG` = "warmouse"
Statistics file's root tag.
- const QString `STATISTICS_NODE_STATISTICS_TAG` = "statistics"
Statistics file's bindings tag.
- const QString `STATISTICS_NODE_APPLICATION_TAG` = "app"
Statistics file's application tag.
- const QString `STATISTICS_NODE_FUNCTION_TAG` = "function"
Statistics file's function tag.
- const QString `STATISTICS_NODE_APPLICATION_NAME_ATTRIBUTE` = "name"
Statistics file's application tag's name-attribute.

- const QString `STATISTICS_NODE_FUNCTION_NAME_ATTRIBUTE` = "name"
Statistics file's function tag's name-attribute.
- const QString `STATISTICS_NODE_FUNCTION_CLICKS_ATTRIBUTE` = "cl"
Statistics file's function tag's clicks-attribute.
- const QString `STATISTICS_NODE_FUNCTION_BUTTON_ATTRIBUTE` = "btn"
Statistics file's function tag's button-attribute.
- const QString `STATISTICS_NODE_FUNCTION_DOUBLECLICK_ATTRIBUTE` = "dc"
Statistics file's function tag's button-attribute.
- const QString `STATISTICS_NODE_DAY_TAG` = "day"
Statistics file's day tag, inside function.
- const QString `STATISTICS_NODE_DATE_ATTRIBUTE` = "date"
Statistics file's day or 3 month tag's date-attribute.
- const QString `STATISTICS_NODE_WEEK_TAG` = "wk"
Statistics file's week tag, inside function.
- const QString `STATISTICS_NODE_WEEK_NUMBER_ATTRIBUTE` = "num"
Statistics file's week tag's week number attribute.
- const QString `STATISTICS_NODE_YEAR_ATTRIBUTE` = "y"
Statistics file's week tag's year attribute.
- const QString `STATISTICS_NODE_2_MONTHS_TAG` = "two_m"
Statistics file's 3 month tag.
- const QString `SETUP_NODE_SETTING_NAME_LANGUAGE` = "language"
Setup file's setting tag's name-attribute value for language setting.

7.38.1 Detailed Description

Structures, constants and functions which are used by several classes, and that are related to the setup file.

7.38.2 Enumeration Type Documentation

7.38.2.1 enum functionType

Different types of events for handling the function.

Some functions require a simple keypress, possibly with control keys. Others require a sequence of keypresses. Use NONE, if the function is not currently implemented at all, but will be in the future, with some new way.

7.38.3 Function Documentation

7.38.3.1 `bool caseInsensitiveStatisticFunctionDataLessThan (const struct statisticFunctionData & s1, const struct statisticFunctionData & s2)`

Case-insensitive comparison function for [statisticFunctionData](#) structs.

Used in `qSort()`

Parameters

- s1* First struct to compare
- s2* Second struct to compare

Returns

true if $s1 < s2$, or false otherwise

7.38.3.2 `void deleteChildrenByAttribute (QDomNode parent, const QString tagName, const QString attribute, const QString value)`

Deletes all children that have a specific attribute with specific value.

Wrapper function for the DOM-parser which ships with Qt

Parameters

- parent* Parent node, whose children we check
- tagName* Tag name of the child that we seek
- attribute* Attribute name that we check for deletion
- value* Attribute value that we check for deletion

Returns

true if child was found, or false if child was not found

7.38.3.3 `bool getChildByName (const QDomNode parent, QDomNode & child, const QString tagName, const QString nameAttributeValue = "")`

Finds the first child of a node with the given tag name.

Wrapper function for the DOM-parser which ships with Qt

Parameters

- parent* Parent node, whose children we check
- child* Reference to a child node, which the function sets if match is found
- tagName* Tag name of the child that we seek
- nameAttributeValue* Optionally, the value of the sought child's name attribute

Returns

true if child was found, or false if child was not found

7.38.3.4 `bool getGrandChildByName (const QDomNode parent, QDomNode & grandChild, const QString parentTagName, const QString childTagName, const QString nameAttributeValue = "")`

Finds the first grandchild of a node with the given tag name, and given parent tag name.

Wrapper function for the DOM-parser which ships with Qt

Parameters

parent Parent node, whose children we check

grandChild Reference to a grandchild node, which the function sets if match is found

parentTagName Tag name of the child's parent

childTagName Tag name of the child that we seek

nameAttributeValue Optionally, the value of the sought grandchild' name attribute

Returns

true if child was found, or false if child was not found

7.38.3.5 `void giveParseError (bool isBindingsFile, const QString details = "")`

Gives a parse error with a message box, with the given details.

Parameters

isBindingsFile True if this parse error is for bindings file, false if for setup file

details Detailed explanation of this parse error

7.38.3.6 `bool hasChildrenWithAttribute (QDomNode parent, const QString tagName, const QString attribute, const QString value)`

Checks if any children have a specific attribute with specific value.

Wrapper function for the DOM-parser which ships with Qt

Parameters

parent Parent node, whose children we check

tagName Tag name of the child that we seek

attribute Attribute name that we check

value Attribute value that we check

Returns

true if child was found, or false if child was not found

7.38.4 Variable Documentation

7.38.4.1 const char* const joystickModeTexts[]

Initial value:

```
{JOYSTICK_MODE_TEXT_NO_JOYSTICK, JOYSTICK_MODE_TEXT_8DIR, JOYSTICK_MODE_TEXT_16DIR,
    IR,
    JOYSTICK_MODE_TEXT_ANALOG_7_BUTTONS, JOY
    STICK_MODE_TEXT_ANALOG_NO_BUTTONS,
    JOYSTICK_MODE_TEXT_4IND, JOYSTICK_MODE_T
    EXT_8IND}
```

7.38.4.2 const mouseButtonAction mouseButtonActions[]

Initial value:

```
{{"[MOUSE_L1]", QT_TRANSLATE_NOOP("QApplication", "Left mouse button"), MOUSE_BU
    TTON_LEFT_CLICK},
    {"[MOUSE_R1]", QT_TRANSLATE_NOOP(
    "QApplication", "Right mouse button"), MOUSE_BUTTON_RIGHT_CLICK},
    {"[MOUSE_WH_UP]", QT_TRANSLATE_NO
    OP("QApplication", "Scroll wheel up"), MOUSE_BUTTON_SCROLL_WHEEL_UP},
    {"[MOUSE_WH_DN]", QT_TRANSLATE_NO
    OP("QApplication", "Scroll wheel down"), MOUSE_BUTTON_SCROLL_WHEEL_PRESS},
    {"[MOUSE_WH_PR]", QT_TRANSLATE_NO
    OP("QApplication", "Scroll wheel click"), MOUSE_BUTTON_SCROLL_WHEEL_DOWN},
    {"[MOUSE_BACK]", QT_TRANSLATE_NOO
    P("QApplication", "Back button click"), MOUSE_BUTTON_BACK_CLICK},
    {"[MOUSE_FORWARD]", QT_TRANSLATE_
    NOOP("QApplication", "Forward button click"), MOUSE_BUTTON_FORWARD_CLICK},
    {"[ALTSTICK]", QT_TRANSLATE_NOOP(
    "QApplication", "Altstick"), SPECIAL_BUTTON_ALTSTICK},
    {"[MOUSE_MAP]", QT_TRANSLATE_NOOP
    ("QApplication", "Display mode map"), SPECIAL_BUTTON_MAP},
    {"[CH_MODE]", QT_TRANSLATE_NOOP("
    QApplication", "Change Mode"), 0},
    {"[CPI_LOW]", QT_TRANSLATE_NOOP("
    QApplication", "CPI low"), 0},
    {"[CPI_DEFAULT]", QT_TRANSLATE_NO
    OP("QApplication", "CPI default"), 0},
    {"[CPI_HIGH]", QT_TRANSLATE_NOOP(
    "QApplication", "CPI high"), 0}
}
```

7.38.4.3 const float pdf_rectangles[][4]

Initial value:

```
{{0.20, 0.05 + pdf_y_offset, 0.13, 0.125}, {0.35, 0.05 + pdf_y_offset, 0.13, 0.1
    25},
    {0.20, 0.20 + pdf_y_offset, 0.13, 0.125}, {0.3
    5, 0.2 + pdf_y_offset, 0.13, 0.125},
    {0.05, 0.35 + pdf_y_offset, 0.13, 0.125}, {0.2
    0, 0.35 + pdf_y_offset, 0.13, 0.125},
    {0.35, 0.35 + pdf_y_offset, 0.13, 0.125},
    {0.55, 0.05 + pdf_y_offset, 0.13, 0.125}, {0.7
    0, 0.05 + pdf_y_offset, 0.13, 0.125},
    {0.55, 0.20 + pdf_y_offset, 0.13, 0.125}, {0.7
    0, 0.20 + pdf_y_offset, 0.13, 0.125},
```

```

                                {0.55, 0.35 + pdf_y_offset, 0.13, 0.125}, {0.7
0, 0.35 + pdf_y_offset, 0.13, 0.125},
                                {0.85, 0.35 + pdf_y_offset, 0.13, 0.125},
                                {0.05, 0.65 + pdf_y_offset, 0.13, 0.125},{0.20
+ 0.015, 0.65 + pdf_y_offset, 0.13, 0.125},
                                {0.39, 0.6225 + pdf_y_offset, 0.13, 0.125}, {0
.39, 0.55 + pdf_y_offset, 0.13, 0.06},
                                {0.39, 0.76 + pdf_y_offset, 0.13, 0.06},
                                {0.55, 0.65 + pdf_y_offset, 0.13, 0.07},
                                {0.55, 0.55 + pdf_y_offset, 0.13, 0.07},
                                {0.70, 0.55 + pdf_y_offset, 0.13, 0.07},
                                {0.85, 0.55 + pdf_y_offset, 0.13, 0.07},
                                {0.85, 0.65 + pdf_y_offset, 0.13, 0.07},
                                {0.85, 0.75 + pdf_y_offset, 0.13, 0.07},
                                {0.70, 0.75 + pdf_y_offset, 0.13, 0.07},
                                {0.55, 0.75 + pdf_y_offset, 0.13, 0.07}}

```

7.38.4.4 const float pdf_rectangles_landscape[][4]

Initial value:

```

{{(0.20 * 1.33, 0.05 + pdf_y_offset, 0.175, 0.125), {0.35 * 1.33, 0.05 + pdf_y_of
fset, 0.175, 0.125},
                                {0.20 * 1.33, 0.20 + pdf_y_offset, 0
.175, 0.125}, {0.35 * 1.33, 0.20 + pdf_y_offset, 0.175, 0.125},
                                {0.05 * 1.33, 0.35 + pdf_y_offset, 0
.175, 0.125}, {0.20 * 1.33, 0.35 + pdf_y_offset, 0.175, 0.125},
                                {0.35 * 1.33, 0.35 + pdf_y_offset, 0
.175, 0.125},
                                {0.55 * 1.33, 0.05 + pdf_y_offset, 0
.175, 0.125}, {0.70 * 1.33, 0.05 + pdf_y_offset, 0.175, 0.125},
                                {0.55 * 1.33, 0.20 + pdf_y_offset, 0
.175, 0.125}, {0.70 * 1.33, 0.2 + pdf_y_offset, 0.175, 0.125},
                                {0.55 * 1.33, 0.35 + pdf_y_offset, 0
.175, 0.125},{0.70*1.33, 0.35 + pdf_y_offset, 0.175, 0.125},
                                {0.85 * 1.33, 0.35 + pdf_y_offset, 0
.175, 0.125},
                                {0.05 * 1.33, 0.65 + pdf_y_offset, 0
.175, 0.125},{0.20 * 1.33, 0.65 + pdf_y_offset, 0.175, 0.125},
                                {0.39 * 1.33, 0.6225 + pdf_y_offset,
0.175, 0.125}, {0.39 * 1.33, 0.55 + pdf_y_offset, 0.175, 0.06},
                                {0.39 * 1.33, 0.76 + pdf_y_offset, 0
.175, 0.06},
                                {0.55 * 1.33, 0.65 + pdf_y_offset, 0
.175, 0.07},
                                {0.55 * 1.33, 0.55 + pdf_y_offset, 0
.175, 0.07},
                                {0.70 * 1.33, 0.55 + pdf_y_offset, 0
.175, 0.07},
                                {0.85 * 1.33, 0.55 + pdf_y_offset, 0
.175, 0.07},
                                {0.85 * 1.33, 0.65 + pdf_y_offset, 0
.175, 0.07},
                                {0.85 * 1.33, 0.75 + pdf_y_offset, 0
.175, 0.07},
                                {0.70 * 1.33, 0.75 + pdf_y_offset, 0
.175, 0.07},
                                {0.70 * 1.33, 0.75 + pdf_y_offset, 0
.175, 0.07},
                                {0.70 * 1.33, 0.75 + pdf_y_offset, 0
.175, 0.07}}

```

```
.175, 0.07}}
                                {0.55 * 1.33, 0.75 + pdf_y_offset, 0
```

7.39 src/setup_objects_common.h File Reference

Functions which are used by several classes, and that are related to the setup file and that require including the group/application object headers.

```
#include "groupobject.h"
#include "applicationobject.h"
```

Functions

- bool [findApplicationByProfileNumber](#) (unsigned int profileNumber, [ApplicationObject](#) &application, QList< [GroupObject](#) > &groupList)

7.39.1 Detailed Description

Functions which are used by several classes, and that are related to the setup file and that require including the group/application object headers. These are in a separate file to avoid cyclical inclusions

7.39.2 Function Documentation

7.39.2.1 bool [findApplicationByProfileNumber](#) (unsigned int *profileNumber*, [ApplicationObject](#) & *application*, QList< [GroupObject](#) > & *groupList*)

Finds an application in given group list by its profile number

Parameters

- profileNumber* The profile number to seek
- application* Reference to where the application is placed if it is found
- groupList* The group list from which we search

Returns

- true if application was found, or false if it was not found

7.40 src/setupparser.h File Reference

Parser class for the xml-formatted setup file.

```
#include "setup_common.h"
#include "groupobject.h"
#include "mouseobject.h"
#include "statisticobject.h"
```

Classes

- struct [statistic](#)
- class [SetupParser](#)

Enumerations

- enum `modifications` { `SETUP_ADD`, `SETUP_DELETE`, `SETUP_MODIFY` }

Variables

- const char `SPECIAL_FUNCTION_L1` [] = "__L1"
- const char `SPECIAL_FUNCTION_R1` [] = "__R1"
- const char `SPECIAL_FUNCTION_W_PR` [] = "__W_PR"
- const char `SPECIAL_FUNCTION_W_UP` [] = "__W_UP"
- const char `SPECIAL_FUNCTION_W_DN` [] = "__W_DN"
- const char `SPECIAL_FUNCTION_FORWARD` [] = "__FORWARD"
- const char `SPECIAL_FUNCTION_BACK` [] = "__BACK"

7.40.1 Detailed Description

Parser class for the xml-formatted setup file. The parser is a singleton object.

7.41 src/simplescreen.h File Reference

Defines the [SimpleScreen](#) class, which creates the basic view, called simple screen in the code.

```
#include <QtGui/QDialog>
#include "setupparser.h"
```

Classes

- class [SimpleScreen](#)

Namespaces

- namespace [Ui](#)

7.41.1 Detailed Description

Defines the [SimpleScreen](#) class, which creates the basic view, called simple screen in the code.

7.42 src/statisticobject.h File Reference

Defines the [StatisticObject](#) class.

```
#include "setup_common.h"
```

Classes

- class [StatisticObject](#)

7.42.1 Detailed Description

Defines the [StatisticObject](#) class.

7.43 src/statisticsalldialog.h File Reference

Defines the [StatisticsAllDialog](#) class, which creates a dialog for displaying click statistics for all modes.

```
#include <QtGui/QDialog>
```

Classes

- class [StatisticsAllDialog](#)

Namespaces

- namespace [Ui](#)

7.43.1 Detailed Description

Defines the [StatisticsAllDialog](#) class, which creates a dialog for displaying click statistics for all modes.

7.44 src/statisticsallmodel.h File Reference

Defines the [StatisticsAllModel](#) class, subclassed from [QAbstractTableModel](#).

```
#include <QObject>
#include <QList>
#include <QAbstractTableModel>
#include "setup_common.h"
#include "setupparser.h"
```

Classes

- class [StatisticsAllModel](#)

7.44.1 Detailed Description

Defines the [StatisticsAllModel](#) class, subclassed from [QAbstractTableModel](#).

7.45 src/statisticsdialog.h File Reference

Defines the [StatisticsDialog](#) class, which creates a dialog for displaying click statistics for a specific mode.

```
#include <QtGui/QDialog>
#include "mystatisticstable.h"
#include "setupparser.h"
```

Classes

- class [StatisticsDialog](#)

Namespaces

- namespace [Ui](#)

7.45.1 Detailed Description

Defines the [StatisticsDialog](#) class, which creates a dialog for displaying click statistics for a specific mode.

7.46 src/statisticsmodel.h File Reference

Defines the [StatisticsModel](#) class, subclassed from [QAbstractTableModel](#).

```
#include <QObject>
#include <QList>
#include <QAbstractTableModel>
#include "setup_common.h"
#include "setupparser.h"
```

Classes

- class [StatisticsModel](#)

7.46.1 Detailed Description

Defines the [StatisticsModel](#) class, subclassed from [QAbstractTableModel](#).

Index

- ~AddApplicationDialog
 - AddApplicationDialog, 12
- ~AddFunctionDialog
 - AddFunctionDialog, 15
- ~ApplicationListModel
 - ApplicationListModel, 17
- ~ApplicationObject
 - ApplicationObject, 20
- ~AutoswitchDetailsDialog
 - AutoswitchDetailsDialog, 27
- ~CategoryObject
 - CategoryObject, 33
- ~CopyApplicationDialog
 - CopyApplicationDialog, 36
- ~CopyCategoryDialog
 - CopyCategoryDialog, 38
- ~CopyFunctionDialog
 - CopyFunctionDialog, 39
- ~EditApplicationDialog
 - EditApplicationDialog, 41
- ~EditCategoryDialog
 - EditCategoryDialog, 45
- ~EditFunctionDialog
 - EditFunctionDialog, 47
- ~GroupObject
 - GroupObject, 57
- ~KeyObject
 - KeyObject, 63
- ~MacroFunctionsDialog
 - MacroFunctionsDialog, 69
- ~MainWindow
 - MainWindow, 71
- ~SimpleScreen
 - SimpleScreen, 120
- ~StatisticsAllDialog
 - StatisticsAllDialog, 126
- ~StatisticsDialog
 - StatisticsDialog, 129
- activateProfile
 - MainWindow, 71
- activateProfileOtherView
 - SimpleScreen, 120
- addApplication
 - SetupParser, 108
- AddApplicationDialog, 11
 - ~AddApplicationDialog, 12
 - AddApplicationDialog, 11
 - changeEvent, 12
 - getCPI, 12
 - getCPI_high, 12
 - getCPI_low, 12
 - getFileName, 12
 - getJoystickMode, 13
 - getName, 13
 - getNewProfileNumber, 13
 - getProfileForReplaced, 13
 - getReplacedApplication, 13
 - getWindowTitlePart, 14
 - isOk, 14
- addBinding
 - SetupParser, 108
- addBindingNoReplace
 - SetupParser, 108
- addCategory
 - SetupParser, 109
- addFunction
 - SetupParser, 109
- AddFunctionDialog, 14
 - ~AddFunctionDialog, 15
 - AddFunctionDialog, 15
 - changeEvent, 15
 - eventFilter, 15
 - getFunctionName, 15
 - getKeypresses, 16
 - getType, 16
 - isOk, 16
 - keyPressEvent, 16
 - keyReleaseEvent, 16
- addGroup
 - SetupParser, 109
- addStatistics
 - SetupParser, 109
- ApplicationListModel, 17
 - ~ApplicationListModel, 17
 - ApplicationListModel, 17
 - data, 17
 - headerData, 18
 - rowCount, 18
- ApplicationObject, 18

- ~ApplicationObject, 20
- ApplicationObject, 20
- getAutoswitchDisabled, 20
- getCategoryList, 20
- getCPI, 20
- getCPI_high, 20
- getCPI_low, 21
- getDoubleClickSpeed, 21
- getFileName, 21
- getGroupName, 21
- getJoystickMode, 21
- getName, 21
- getProfileNumber, 22
- getScrollWheelLines, 22
- getSensitivity, 22
- getWindowTitlePart, 22
- operator<, 22
- operator=, 22
- operator==, 23
- setAutoswitchDisabled, 23
- setCategoryListByNode, 23
- setCPI, 23
- setCPI_high, 24
- setCPI_low, 24
- setDoubleClickSpeed, 24
- setFileName, 24
- setFunctionBound, 25
- setFunctionListByNode, 25
- setGroupName, 25
- setJoystickMode, 25
- setName, 26
- setProfileNumber, 26
- setScrollWheelLines, 26
- setSensitivity, 26
- setWindowTitlePart, 26
- AtUsbHid.h
 - chBEGINTHREADEX, 136
- AutoswitchDetailsDialog, 27
 - ~AutoswitchDetailsDialog, 27
 - AutoswitchDetailsDialog, 27
 - changeEvent, 28
 - getAction, 28
 - getFileName, 28
 - getWindowTitlePart, 28
- bindButton
 - MouseButtonObject, 77
- buttonActivated
 - MouseObject, 89
- ButtonBindingObject, 29
 - ButtonBindingObject, 29
 - getFlags, 29
 - getFunction, 29
 - operator=, 30
 - operator==, 30
 - setFlags, 30
 - setFunction, 30
- calculateChecksum
 - MouseHardware, 82
- caseInsensitiveLessThan
 - mydialogs_common.h, 156
- caseInsensitiveStatisticFunctionDataLessThan
 - setup_common.h, 165
- CategoryListModel, 31
 - CategoryListModel, 31
 - data, 31
 - headerData, 32
 - rowCount, 32
- CategoryObject, 32
 - ~CategoryObject, 33
 - CategoryObject, 33
 - getFunctionList, 33
 - getName, 33
 - operator<, 34
 - operator=, 34
 - operator==, 34
 - setFunctionBound, 34
 - setFunctionListByNode, 34
 - setName, 35
- changeEvent
 - AddApplicationDialog, 12
 - AddFunctionDialog, 15
 - AutoswitchDetailsDialog, 28
 - CopyApplicationDialog, 36
 - CopyCategoryDialog, 38
 - CopyFunctionDialog, 39
 - EditApplicationDialog, 42
 - EditCategoryDialog, 45
 - EditFunctionDialog, 47
 - MacroFunctionsDialog, 69
 - SimpleScreen, 120
 - StatisticsAllDialog, 126
 - StatisticsDialog, 130
- changeView
 - MainWindow, 71
 - SimpleScreen, 120
- charToKeyInfo, 35
- chBEGINTHREADEX
 - AtUsbHid.h, 136
- checkLibrary
 - MouseHardware, 82
- checkMouseState
 - MouseHardware, 82
- chooseTask
 - HardwareThread, 61
- closeLibrary
 - MouseHardware, 82

- closeOtherView
 - MainWindow, 71
 - SimpleScreen, 120
- closing
 - MapWidget, 76
- columnCount
 - StatisticsAllModel, 127
 - StatisticsModel, 131
- compareProfileInfoBlocks
 - MouseEvent, 89
- configureButtons
 - MouseEvent, 89
- contextMenuEvent
 - MyAllStatisticsTable, 101
 - MyStatisticsTable, 103
- CopyApplicationDialog, 35
 - ~CopyApplicationDialog, 36
 - changeEvent, 36
 - CopyApplicationDialog, 36
 - getApplicationName, 36
 - getSelectedGroup, 36
 - isOk, 37
- CopyCategoryDialog, 37
 - ~CopyCategoryDialog, 38
 - changeEvent, 38
 - CopyCategoryDialog, 37
 - getApplicationName, 38
 - getCategoryName, 38
 - getGroupName, 38
- CopyFunctionDialog, 39
 - ~CopyFunctionDialog, 39
 - changeEvent, 39
 - CopyFunctionDialog, 39
 - getApplicationName, 40
 - getCategoryName, 40
 - getFunctionName, 40
 - getGroupName, 40
- createInfoBlockContents
 - MouseEvent, 90
- createMappingBlockContents
 - MouseEvent, 90
- data
 - ApplicationListModel, 17
 - CategoryListModel, 31
 - FunctionListModel, 52
 - StatisticsAllModel, 128
 - StatisticsModel, 131
- deleteApplication
 - SetupParser, 110
- deleteCategory
 - SetupParser, 110
- deleteChildrenByAttribute
 - setup_common.h, 165
- deleteFunction
 - SetupParser, 110
- deleteGroup
 - SetupParser, 111
- EditApplicationDialog, 40
 - ~EditApplicationDialog, 41
 - changeEvent, 42
 - EditApplicationDialog, 41
 - getAction, 42
 - getCPI, 42
 - getCPI_high, 42
 - getCPI_low, 42
 - getFileName, 42
 - getJoystickMode, 43
 - getName, 43
 - getNewProfileNumber, 43
 - getProfileForReplaced, 43
 - getReplacedApplication, 43
 - getWindowTitlePart, 44
- EditCategoryDialog, 44
 - ~EditCategoryDialog, 45
 - changeEvent, 45
 - EditCategoryDialog, 45
 - getAction, 45
 - getCategoryName, 45
 - getCopiedApplicationName, 45
 - getCopiedCategoryName, 46
 - getCopiedGroupName, 46
- EditFunctionDialog, 46
 - ~EditFunctionDialog, 47
 - changeEvent, 47
 - EditFunctionDialog, 47
 - eventFilter, 47
 - getAction, 48
 - getCopiedApplicationName, 48
 - getCopiedCategoryName, 48
 - getCopiedFunctionName, 48
 - getCopiedGroupName, 48
 - getFunctionName, 49
 - getKeypresses, 49
 - getType, 49
 - keyPressEvent, 49
 - keyReleaseEvent, 49
 - setKeypresses, 49
- enableFirmwareUpdate
 - MouseEvent, 83
 - MouseEvent, 90
- eraseAll
 - MouseEvent, 90
- eraseBlock
 - MouseEvent, 83
- eraseFlash
 - MouseEvent, 83

- eventFilter
 - AddFunctionDialog, 15
 - EditFunctionDialog, 47
- findApplicationByProfileNumber
 - setup_objects_common.h, 169
- flagsChanged
 - MainWindow, 71
 - MouseEvent, 91
- FunctionDelegate, 50
 - FunctionDelegate, 50
 - paint, 50
- FunctionListModel, 51
 - data, 52
 - FunctionListModel, 51
 - headerData, 52
 - rowCount, 52
- FunctionObject, 53
 - FunctionObject, 53
 - getApplicationName, 54
 - getData, 54
 - getName, 54
 - getType, 54
 - isBound, 54
 - operator<, 54
 - operator>, 55
 - operator=, 54
 - operator==, 55
 - setApplicationName, 55
 - setBound, 55
 - setData, 55
 - setName, 56
 - setType, 56
- functionType
 - setup_common.h, 164
- getAction
 - AutoswitchDetailsDialog, 28
 - EditApplicationDialog, 42
 - EditCategoryDialog, 45
 - EditFunctionDialog, 48
- getActiveApplicationIndex
 - MainWindow, 71
- getActiveApplicationName
 - MainWindow, 72
 - MouseEvent, 91
- getActiveButton
 - MouseEvent, 91
- getActiveGroupIndex
 - MainWindow, 72
- getActiveProfile
 - MouseHardware, 83
- getActiveProfileNumber
 - MouseEvent, 91
- getAllClicks
 - StatisticObject, 123
- getAllStatistics
 - SetupParser, 111
- getAndResetStatistics
 - MouseEvent, 91
- getApplicationList
 - GroupObject, 57
- getApplicationName
 - CopyApplicationDialog, 36
 - CopyCategoryDialog, 38
 - CopyFunctionDialog, 40
 - FunctionObject, 54
 - StatisticObject, 123
- getApplicationStatistics
 - SetupParser, 112
- getAutoswitchDisabled
 - ApplicationObject, 20
- getBindingsDomDocument
 - SetupParser, 112
- getBlockChecksum
 - MouseHardware, 83
- getButtonBindings
 - MouseButtonObject, 77
- getButtonEnum
 - MouseEvent, 92
- getButtonString
 - MouseEvent, 92
- getCategoryList
 - ApplicationObject, 20
- getCategoryName
 - CopyCategoryDialog, 38
 - CopyFunctionDialog, 40
 - EditCategoryDialog, 45
- getChildByName
 - setup_common.h, 165
- getCopiedApplicationName
 - EditCategoryDialog, 45
 - EditFunctionDialog, 48
- getCopiedCategoryName
 - EditCategoryDialog, 46
 - EditFunctionDialog, 48
- getCopiedFunctionName
 - EditFunctionDialog, 48
- getCopiedGroupName
 - EditCategoryDialog, 46
 - EditFunctionDialog, 48
- getCPI
 - AddApplicationDialog, 12
 - ApplicationObject, 20
 - EditApplicationDialog, 42
- getCPI_high
 - AddApplicationDialog, 12
 - ApplicationObject, 20

- EditApplicationDialog, 42
- getCPI_low
 - AddApplicationDialog, 12
 - ApplicationObject, 21
 - EditApplicationDialog, 42
- getData
 - FunctionObject, 54
- getDelayMs1
 - KeyObject, 64
- getDelayMs2
 - KeyObject, 64
- getDoubleClickSpeed
 - ApplicationObject, 21
- getFileName
 - AddApplicationDialog, 12
 - ApplicationObject, 21
 - AutoswitchDetailsDialog, 28
 - EditApplicationDialog, 42
- getFirmwareRevision
 - MouseObject, 92
- getFlags
 - ButtonBindingObject, 29
 - MainWindow, 72
 - MouseObject, 93
- getFreeBlocks
 - MouseHardware, 84
- getFreeButtonDisplay
 - MouseObject, 93
 - MouseWidget, 99
- getFreeButtons
 - MouseObject, 93
- getFreeProfileNumber
 - mydialogs_common.h, 156
- getFunction
 - ButtonBindingObject, 29
- getFunctionBindings
 - MouseObject, 93
- getFunctionClicks
 - StatisticObject, 123
- getFunctionList
 - CategoryObject, 33
- getFunctionName
 - AddFunctionDialog, 15
 - CopyFunctionDialog, 40
 - EditFunctionDialog, 49
- getFunctionWithFlags
 - MouseButtonObject, 78
- getGrandChildByName
 - setup_common.h, 165
- getGroupList
 - MainWindow, 72
 - SetupParser, 112
- getGroupName
 - ApplicationObject, 21
 - CopyCategoryDialog, 38
 - CopyFunctionDialog, 40
- getHeaderSvg
 - MainWindow, 72
- getJoyCoords
 - MouseHardware, 84
 - MouseObject, 93
- getJoystickMode
 - AddApplicationDialog, 13
 - ApplicationObject, 21
 - EditApplicationDialog, 43
- getKey
 - KeyObject, 64
- getKeypresses
 - AddFunctionDialog, 16
 - EditFunctionDialog, 49
- getLanguage
 - SetupParser, 112
- getLastDifferenceResult
 - HardwareThread, 61
- getMacroGUIString
 - KeypressParser, 66
- getMacroInternalString
 - KeypressParser, 66
- getMappingBlockDifferences
 - MouseObject, 94
- getModifiers
 - KeyObject, 64
- getMouseButtonPointer
 - MouseObject, 94
- getMouseState
 - MouseObject, 94
- getName
 - AddApplicationDialog, 13
 - ApplicationObject, 21
 - CategoryObject, 33
 - EditApplicationDialog, 43
 - FunctionObject, 54
 - GroupObject, 57
 - MouseButtonObject, 78
- getNewProfileNumber
 - AddApplicationDialog, 13
 - EditApplicationDialog, 43
- getPartiallyFreeButtons
 - MouseObject, 94
- getProfileCopy
 - MouseHardware, 84
- getProfileForReplaced
 - AddApplicationDialog, 13
 - EditApplicationDialog, 43
- getProfileNumber
 - ApplicationObject, 22
- getReplacedApplication
 - AddApplicationDialog, 13

- EditApplicationDialog, 43
- getRevision
 - MouseHardware, 84
- getScrollWheelLines
 - ApplicationObject, 22
- getSelectedGroup
 - CopyApplicationDialog, 36
- getSensitivity
 - ApplicationObject, 22
- getType
 - AddFunctionDialog, 16
 - EditFunctionDialog, 49
 - FunctionObject, 54
- getWindowTitlePart
 - AddApplicationDialog, 14
 - ApplicationObject, 22
 - AutoswitchDetailsDialog, 28
 - EditApplicationDialog, 44
- giveParseError
 - setup_common.h, 166
- GroupObject, 56
 - ~GroupObject, 57
 - getApplicationList, 57
 - getName, 57
 - GroupObject, 57
 - operator<, 58
 - operator=, 58
 - operator==, 58
 - setApplicationListByNode, 58
 - setFunctionBound, 59
 - setName, 59
- GroupOrAppWidget, 59
 - GroupOrAppWidget, 60
 - setHaveNavigationTab, 60
 - setNumButtons, 60
- HardwareThread, 61
 - chooseTask, 61
 - getLastDifferenceResult, 61
 - HardwareThread, 61
 - run, 61
- hasBinding
 - SetupParser, 113
- hasChildrenWithAttribute
 - setup_common.h, 166
- headerData
 - ApplicationListModel, 18
 - CategoryListModel, 32
 - FunctionListModel, 52
 - StatisticsAllModel, 128
 - StatisticsModel, 131
- IdentifiedPushButton, 62
 - IdentifiedPushButton, 62
 - setColor, 62
- importApplication
 - SetupParser, 113
- initHardware
 - MouseObject, 95
- initialized
 - MainWindow, 72
- instance
 - KeypressParser, 67
 - MouseHardware, 85
 - MouseObject, 95
 - SetupParser, 113
- internalStringFromQtKey
 - keypress_common.h, 149
- internalStringFromQtModifier
 - keypress_common.h, 149
- isBound
 - FunctionObject, 54
- isBoundInApplication
 - MouseButtonObject, 78
- isBoundToFunction
 - MouseButtonObject, 78
- isBoundToFunctionWithFlags
 - MouseButtonObject, 79
- isBoundToThis
 - MouseButtonObject, 79
- isFunctionBound
 - SetupParser, 113
- isInitializedOK
 - MainWindow, 72
 - MouseHardware, 85
- isMouseConnected
 - MouseObject, 95
- isOk
 - AddApplicationDialog, 14
 - AddFunctionDialog, 16
 - CopyApplicationDialog, 37
- isOnlyPartiallyFreeInApplication
 - MouseButtonObject, 79
- isProfileSynced
 - MouseObject, 95
- joystickModeTexts
 - setup_common.h, 167
- KeyObject, 63
 - ~KeyObject, 63
 - getDelayMs1, 64
 - getDelayMs2, 64
 - getKey, 64
 - getModifiers, 64
 - KeyObject, 63, 64
 - operator=, 64
 - setDelayMs1, 65

- setDelayMs2, 65
 - setKey, 65
 - setModifiers, 65
- keypress_common.h
 - internalStringFromQtKey, 149
 - internalStringFromQtModifier, 149
 - STRING_TO_MODIFIER_HID, 149
- KeyPressEvent
 - AddFunctionDialog, 16
 - EditFunctionDialog, 49
- KeyPressParser, 66
 - getMacroGUIString, 66
 - getMacroInternalString, 66
 - instance, 67
 - keypressStringToHID, 67
 - keypressStringToQt, 67
 - macroStringToHID, 67
 - macroStringToQt, 68
- keypressStringToHID
 - KeyPressParser, 67
- keypressStringToQt
 - KeyPressParser, 67
- keyReleaseEvent
 - AddFunctionDialog, 16
 - EditFunctionDialog, 49
- MacroFunctionsDialog, 68
 - ~MacroFunctionsDialog, 69
 - changeEvent, 69
 - MacroFunctionsDialog, 69
- macroStringToHID
 - KeyPressParser, 67
- macroStringToQt
 - KeyPressParser, 68
- MainWindow, 69
 - ~MainWindow, 71
 - activateProfile, 71
 - changeView, 71
 - closeOtherView, 71
 - flagsChanged, 71
 - getActiveApplicationIndex, 71
 - getActiveApplicationName, 72
 - getActiveGroupIndex, 72
 - getFlags, 72
 - getGroupList, 72
 - getHeaderSvg, 72
 - initialized, 72
 - isInitializedOK, 72
 - MainWindow, 71
 - mapClosed, 73
 - paintEvent, 73
 - processExternalMessage, 73
 - profileEvent, 73
 - quitModeware, 73
 - refreshSimpleView, 73
 - setCategoryIndex, 73
 - setCategoryListFocus, 74
 - setCheckBoxes, 74
 - setFunctionIndex, 74
 - setFunctionListFocus, 74
 - setInactive, 74
 - showThisView, 74
 - stopAssigning, 75
 - updateHeader, 75
 - updateStatistics, 75
- mapClosed
 - MainWindow, 73
- MapWidget, 75
 - closing, 76
 - MapWidget, 76
- modifyApplication
 - SetupParser, 114
- modifyCategory
 - SetupParser, 114
- modifyFunction
 - SetupParser, 114
- modifyGroup
 - SetupParser, 115
- mouseButtonAction, 76
- mouseButtonActions
 - setup_common.h, 167
- MouseButtonObject, 76
 - bindButton, 77
 - getButtonBindings, 77
 - getFunctionWithFlags, 78
 - getName, 78
 - isBoundInApplication, 78
 - isBoundToFunction, 78
 - isBoundToFunctionWithFlags, 79
 - isBoundToThis, 79
 - isOnlyPartiallyFreeInApplication, 79
 - MouseButtonObject, 77
 - operator=, 79
 - unbindAll, 80
 - unbindButton, 80
- mouseButtonRepresentation, 80
- mouseButtonWithFlags, 81
- MouseHardware, 81
 - calculateChecksum, 82
 - checkLibrary, 82
 - checkMouseState, 82
 - closeLibrary, 82
 - enableFirmwareUpdate, 83
 - eraseBlock, 83
 - eraseFlash, 83
 - getActiveProfile, 83
 - getBlockChecksum, 83
 - getFreeBlocks, 84

- getJoyCoords, 84
- getProfileCopy, 84
- getRevision, 84
- instance, 85
- isInitializedOK, 85
- pollProfileState, 85
- quit, 85
- readMouse, 86
- readProfileInfoBlock, 86
- setProfile, 86
- setProfileCopy, 86
- updateProfileCopy, 87
- writeMouse, 87
- writeProfileInfoBlock, 87
- writeProfileKeyMappingBlock, 87
- MouseObject, 88
 - buttonActivated, 89
 - compareProfileInfoBlocks, 89
 - configureButtons, 89
 - createInfoBlockContents, 90
 - createMappingBlockContents, 90
 - enableFirmwareUpdate, 90
 - eraseAll, 90
 - flagsChanged, 91
 - getActiveApplicationName, 91
 - getActiveButton, 91
 - getActiveProfileNumber, 91
 - getAndResetStatistics, 91
 - getButtonEnum, 92
 - getButtonString, 92
 - getFirmwareRevision, 92
 - getFlags, 93
 - getFreeButtonDisplay, 93
 - getFreeButtons, 93
 - getFunctionBindings, 93
 - getJoyCoords, 93
 - getMappingBlockDifferences, 94
 - getMouseButtonPointer, 94
 - getMouseState, 94
 - getPartiallyFreeButtons, 94
 - initHardware, 95
 - instance, 95
 - isMouseConnected, 95
 - isProfileSynced, 95
 - profileEvent, 95
 - quit, 96
 - refreshWidget, 96
 - refreshWidgetSignal, 96
 - setActiveButton, 96
 - setActiveProfile, 96
 - setButtonToFunction, 96
 - setFlags, 97
 - setFreeButtonDisplay, 97
 - setMainWindowPtr, 97
 - setMouseState, 97
 - setMouseWidget, 97
 - setProfileSynced, 98
 - statisticsEvent, 98
 - unbindAll, 98
- mouseobject.h
 - representation, 154
- mousePressEvent
 - MouseWidget, 99
- MouseWidget, 98
 - getFreeButtonDisplay, 99
 - mousePressEvent, 99
 - MouseWidget, 99
 - paintEvent, 99
 - refresh, 99
 - setFreeButtonDisplay, 100
- MyAllStatisticsTable, 100
 - contextMenuEvent, 101
 - MyAllStatisticsTable, 100
 - paintEvent, 101
 - recalculateColumnSizes, 101
 - setShortcuts, 101
- mydialogs_common.h
 - caseInsensitiveLessThan, 156
 - getFreeProfileNumber, 156
- MyKey, 101
- myKey, 102
- MyKeyPressHID, 102
- MyListView, 102
 - MyListView, 102
- MyStatisticsTable, 103
 - contextMenuEvent, 103
 - MyStatisticsTable, 103
 - paintEvent, 103
 - recalculateColumnSizes, 104
 - setShortcuts, 104
- MyToolButton, 104
 - MyToolButton, 104
 - paintEvent, 105
 - setType, 105
- operator<
 - ApplicationObject, 22
 - CategoryObject, 34
 - FunctionObject, 54
 - GroupObject, 58
 - StatisticObject, 123
- operator>
 - FunctionObject, 55
 - StatisticObject, 124
- operator=
 - ApplicationObject, 22
 - ButtonBindingObject, 30
 - CategoryObject, 34

- FunctionObject, 54
- GroupObject, 58
- KeyObject, 64
- MouseButtonObject, 79
- StatisticObject, 124
- operator==
 - ApplicationObject, 23
 - ButtonBindingObject, 30
 - CategoryObject, 34
 - FunctionObject, 55
 - GroupObject, 58
 - StatisticObject, 124
- paint
 - FunctionDelegate, 50
- paintEvent
 - MainWindow, 73
 - MouseWidget, 99
 - MyAllStatisticsTable, 101
 - MyStatisticsTable, 103
 - MyToolButton, 105
- parseBindings
 - SetupParser, 115
- parseConfig
 - SimpleScreen, 120
- parseSetup
 - SetupParser, 115
- parseStatistics
 - SetupParser, 116
- pdf_rectangles
 - setup_common.h, 167
- pdf_rectangles_landscape
 - setup_common.h, 168
- png_stream_to_byte_array_closure_t, 105
- pollProfileState
 - MouseHardware, 85
- processExternalMessage
 - MainWindow, 73
- PROFILE_INFO_BLOCK, 105
- profileEvent
 - MainWindow, 73
 - MouseObject, 95
- quit
 - MouseHardware, 85
 - MouseObject, 96
- quitModeware
 - MainWindow, 73
 - SimpleScreen, 121
- readBindingsFile
 - SetupParser, 116
- readMouse
 - MouseHardware, 86
- readProfileInfoBlock
 - MouseHardware, 86
- readSetupFile
 - SetupParser, 116
- readStatisticsFile
 - SetupParser, 117
- recalculateColumnSizes
 - MyAllStatisticsTable, 101
 - MyStatisticsTable, 104
- refresh
 - MouseWidget, 99
- refreshSimpleView
 - MainWindow, 73
- refreshView
 - SimpleScreen, 121
- refreshWidget
 - MouseObject, 96
- refreshWidgetSignal
 - MouseObject, 96
- removeBinding
 - SetupParser, 117
- representation
 - mouseobject.h, 154
- rowCount
 - ApplicationListModel, 18
 - CategoryListModel, 32
 - FunctionListModel, 52
 - StatisticsAllModel, 128
 - StatisticsModel, 131
- run
 - HardwareThread, 61
- saveApplication
 - SetupParser, 117
- setActiveButton
 - MouseObject, 96
- setActiveProfile
 - MouseObject, 96
- setApplicationListByNode
 - GroupObject, 58
- setApplicationName
 - FunctionObject, 55
 - StatisticObject, 124
- setAutoswitchDisabled
 - ApplicationObject, 23
- setBound
 - FunctionObject, 55
- setButtonToFunction
 - MouseObject, 96
- setCategoryIndex
 - MainWindow, 73
- setCategoryListByNode
 - ApplicationObject, 23
- setCategoryListFocus

- MainWindow, 74
- setCheckBoxes
 - MainWindow, 74
- setColor
 - IdentifiedPushButton, 62
- setCPI
 - ApplicationObject, 23
- setCPI_high
 - ApplicationObject, 24
- setCPI_low
 - ApplicationObject, 24
- setData
 - FunctionObject, 55
- setDelayMs1
 - KeyObject, 65
- setDelayMs2
 - KeyObject, 65
- setDoubleClickSpeed
 - ApplicationObject, 24
- setFileName
 - ApplicationObject, 24
- setFlags
 - ButtonBindingObject, 30
 - MouseObject, 97
- setFreeButtonDisplay
 - MouseObject, 97
 - MouseWidget, 100
- setFunction
 - ButtonBindingObject, 30
- setFunctionBound
 - ApplicationObject, 25
 - CategoryObject, 34
 - GroupObject, 59
 - SetupParser, 117
- setFunctionClicks
 - StatisticObject, 124
- setFunctionIndex
 - MainWindow, 74
- setFunctionListByNode
 - ApplicationObject, 25
 - CategoryObject, 34
- setFunctionListFocus
 - MainWindow, 74
- setGroupList
 - SetupParser, 118
- setGroupName
 - ApplicationObject, 25
- setHaveNavigationTab
 - GroupOrAppWidget, 60
- setInactive
 - MainWindow, 74
- setJoystickMode
 - ApplicationObject, 25
- setKey
 - KeyObject, 65
- setKeypresses
 - EditFunctionDialog, 49
- setLanguage
 - SetupParser, 118
- setMainWindowPtr
 - MouseObject, 97
- setModifiers
 - KeyObject, 65
- setMouseState
 - MouseObject, 97
- setMouseWidget
 - MouseObject, 97
- setName
 - ApplicationObject, 26
 - CategoryObject, 35
 - FunctionObject, 56
 - GroupObject, 59
- setNumButtons
 - GroupOrAppWidget, 60
- setProfile
 - MouseHardware, 86
- setProfileCopy
 - MouseHardware, 86
- setProfileNumber
 - ApplicationObject, 26
- setProfileSynced
 - MouseObject, 98
- setScrollWheelLines
 - ApplicationObject, 26
- setSensitivity
 - ApplicationObject, 26
- setShortcuts
 - MyAllStatisticsTable, 101
 - MyStatisticsTable, 104
- setType
 - FunctionObject, 56
 - MyToolButton, 105
- setup_common.h
 - caseInsensitiveStatisticFunctionData-LessThan, 165
 - deleteChildrenByAttribute, 165
 - functionType, 164
 - getChildByName, 165
 - getGrandChildByName, 165
 - giveParseError, 166
 - hasChildrenWithAttribute, 166
 - joystickModeTexts, 167
 - mouseButtonActions, 167
 - pdf_rectangles, 167
 - pdf_rectangles_landscape, 168
- setup_objects_common.h
 - findApplicationByProfileNumber, 169
- SetupParser, 106

- addApplication, 108
- addBinding, 108
- addBindingNoReplace, 108
- addCategory, 109
- addFunction, 109
- addGroup, 109
- addStatistics, 109
- deleteApplication, 110
- deleteCategory, 110
- deleteFunction, 110
- deleteGroup, 111
- getAllStatistics, 111
- getApplicationStatistics, 112
- getBindingsDomDocument, 112
- getGroupList, 112
- getLanguage, 112
- hasBinding, 113
- importApplication, 113
- instance, 113
- isFunctionBound, 113
- modifyApplication, 114
- modifyCategory, 114
- modifyFunction, 114
- modifyGroup, 115
- parseBindings, 115
- parseSetup, 115
- parseStatistics, 116
- readBindingsFile, 116
- readSetupFile, 116
- readStatisticsFile, 117
- removeBinding, 117
- saveApplication, 117
- setFunctionBound, 117
- setGroupList, 118
- setLanguage, 118
- thisFunctionExists, 118
- updateStatistics, 118
- setWindowTitlePart
 - ApplicationObject, 26
- showThisView
 - MainWindow, 74
 - SimpleScreen, 121
- SimpleScreen, 119
 - ~SimpleScreen, 120
 - activateProfileOtherView, 120
 - changeEvent, 120
 - changeView, 120
 - closeOtherView, 120
 - parseConfig, 120
 - quitModeware, 121
 - refreshView, 121
 - showThisView, 121
 - SimpleScreen, 120
 - updateActiveMode, 121
- sortFunctions
 - StatisticObject, 125
- src/addapplicationdialog.h, 133
- src/addfunctiondialog.h, 133
- src/applicationlistmodel.h, 134
- src/applicationobject.h, 134
- src/AtUsbHid.h, 135
- src/autoswitchdetailsdialog.h, 136
- src/buttonbindingobject.h, 137
- src/categorylistmodel.h, 137
- src/categoryobject.h, 137
- src/copyapplicationdialog.h, 138
- src/copycategorydialog.h, 138
- src/copyfunctiondialog.h, 139
- src/editapplicationdialog.h, 139
- src/editcategorydialog.h, 140
- src/editfunctiondialog.h, 140
- src/functiondelegate.h, 141
- src/functionlistmodel.h, 141
- src/functionobject.h, 141
- src/groupobject.h, 142
- src/grouporappwidget.h, 142
- src/hardwarethread.h, 142
- src/identifiedpushbutton.h, 143
- src/keyobject.h, 143
- src/keypress_common.h, 144
- src/keypressparser.h, 150
- src/macrofunctionsdialog.h, 150
- src/mainwindow.h, 150
- src/mapwidget.h, 151
- src/mousebuttonobject.h, 152
- src/mousehardware.h, 152
- src/mouseobject.h, 153
- src/mousewidget.h, 154
- src/myallstatisticstable.h, 155
- src/mydialogs_common.h, 155
- src/mylistview.h, 156
- src/mystatisticstable.h, 156
- src/mytoolbarbutton.h, 157
- src/setup_common.h, 157
- src/setup_objects_common.h, 169
- src/setupparser.h, 169
- src/simplescreen.h, 170
- src/statisticobject.h, 170
- src/statisticsalldialog.h, 171
- src/statisticsallmodel.h, 171
- src/statisticsdialog.h, 172
- src/statisticsmodel.h, 172
- statistic, 121
- statisticFunctionData, 122
- StatisticObject, 122
 - getAllClicks, 123
 - getApplicationName, 123
 - getFunctionClicks, 123

- operator<, 123
- operator>, 124
- operator=, 124
- operator==, 124
- setApplicationName, 124
- setFunctionClicks, 124
- sortFunctions, 125
- StatisticObject, 123
- StatisticsAllDialog, 125
 - ~StatisticsAllDialog, 126
 - changeEvent, 126
 - StatisticsAllDialog, 126
- StatisticsAllModel, 127
 - columnCount, 127
 - data, 128
 - headerData, 128
 - rowCount, 128
 - StatisticsAllModel, 127
- StatisticsDialog, 129
 - ~StatisticsDialog, 129
 - changeEvent, 130
 - StatisticsDialog, 129
- statisticsEvent
 - MouseObject, 98
- StatisticsModel, 130
 - columnCount, 131
 - data, 131
 - headerData, 131
 - rowCount, 131
 - StatisticsModel, 130
- stopAssigning
 - MainWindow, 75
- STRING_TO_MODIFIER_HID
 - keypress_common.h, 149
- stringToKeyInfo, 132
- stringToModifierInfo, 132

- thisFunctionExists
 - SetupParser, 118

- Ui, 9
- unbindAll
 - MouseButtonObject, 80
 - MouseObject, 98
- unbindButton
 - MouseButtonObject, 80
- updateActiveMode
 - SimpleScreen, 121
- updateHeader
 - MainWindow, 75
- updateProfileCopy
 - MouseHardware, 87
- updateStatistics
 - MainWindow, 75
 - SetupParser, 118
- writeMouse
 - MouseHardware, 87
- writeProfileInfoBlock
 - MouseHardware, 87
- writeProfileKeyMappingBlock
 - MouseHardware, 87