



Delta3D Version 2.4.0

**dtQt::**

## **Reference Manual**



# Contents

---

<b>1</b>	<b>Main Page</b>	<b>1</b>
<b>2</b>	<b>Directory Documentation</b>	<b>3</b>
2.1	inc/dtQt/ Directory Reference . . . . .	3
2.2	src/dtQt/ Directory Reference . . . . .	5
2.3	inc/ Directory Reference . . . . .	6
2.4	src/ Directory Reference . . . . .	7
<b>3</b>	<b>Namespace Documentation</b>	<b>9</b>
3.1	dtDAL Namespace Reference . . . . .	9
3.2	dtQt Namespace Reference . . . . .	10
3.2.1	Detailed Description . . . . .	12
3.2.2	Variable Documentation . . . . .	12
3.2.2.1	NUM_DECIMAL_DIGITS_DOUBLE . . . . .	12
3.2.2.2	NUM_DECIMAL_DIGITS_FLOAT . . . . .	12
3.2.2.3	STATIC_KEY_MAP . . . . .	12
<b>4</b>	<b>Class Documentation</b>	<b>13</b>
4.1	BaseLibraryListEditor Class Reference . . . . .	13
4.1.1	Member Enumeration Documentation . . . . .	14
4.1.1.1	Errors . . . . .	14
4.1.2	Constructor & Destructor Documentation . . . . .	14
4.1.2.1	BaseLibraryListEditor . . . . .	14
4.1.2.2	~BaseLibraryListEditor . . . . .	14
4.1.3	Member Function Documentation . . . . .	14
4.1.3.1	EnableButtons . . . . .	14
4.1.3.2	GetLibraryListWidget . . . . .	14
4.1.3.3	GetLibraryNames . . . . .	14
4.1.3.4	HandleFailure . . . . .	14
4.1.3.5	RefreshLibraries . . . . .	14
4.1.3.6	SelectLibraryToOpen . . . . .	14
4.1.3.7	ShiftLibraryDown . . . . .	14
4.1.3.8	ShiftLibraryUp . . . . .	14
4.1.3.9	showEvent . . . . .	14
4.1.3.10	SpawnDeleteConfirmation . . . . .	14
4.1.3.11	SpawnFileBrowser . . . . .	15
4.2	BasePropertyEditor Class Reference . . . . .	16
4.2.1	Detailed Description . . . . .	16

4.2.2	Member Typedef Documentation . . . . .	17
4.2.2.1	PropertyContainerRefPtrVector . . . . .	17
4.2.3	Constructor & Destructor Documentation . . . . .	17
4.2.3.1	BasePropertyEditor . . . . .	17
4.2.3.2	~BasePropertyEditor . . . . .	17
4.2.4	Member Function Documentation . . . . .	17
4.2.4.1	ActorPropertyChanged . . . . .	17
4.2.4.2	buildDynamicControls . . . . .	17
4.2.4.3	CommitCurrentEdits . . . . .	17
4.2.4.4	GetDynamicControlFactory . . . . .	17
4.2.4.5	GetGroupBoxLabelText . . . . .	17
4.2.4.6	GetPropertyEditorModel . . . . .	17
4.2.4.7	GetRootControl . . . . .	17
4.2.4.8	GetSelectedPropertyContainers . . . . .	17
4.2.4.9	HandlePropertyContainersSelected . . . . .	17
4.2.4.10	PropertyAboutToChangeFromControl . . . . .	17
4.2.4.11	PropertyChangedFromControl . . . . .	17
4.2.4.12	ProxyNameChanged . . . . .	17
4.3	DeltaStepper Class Reference . . . . .	18
4.3.1	Constructor & Destructor Documentation . . . . .	18
4.3.1.1	DeltaStepper . . . . .	18
4.3.1.2	~DeltaStepper . . . . .	18
4.3.2	Member Function Documentation . . . . .	18
4.3.2.1	Start . . . . .	18
4.3.2.2	Stop . . . . .	18
4.3.2.3	Tick . . . . .	18
4.4	DialogListSelection Class Reference . . . . .	19
4.4.1	Detailed Description . . . . .	19
4.4.2	Constructor & Destructor Documentation . . . . .	19
4.4.2.1	DialogListSelection . . . . .	19
4.4.2.2	~DialogListSelection . . . . .	19
4.4.3	Member Function Documentation . . . . .	19
4.4.3.1	GetSelectedItem . . . . .	19
4.4.3.2	onCurrentRowChanged . . . . .	20
4.4.3.3	onItemClicked . . . . .	20
4.4.3.4	onItemDoubleClicked . . . . .	20
4.4.3.5	onSelectionChanged . . . . .	20
4.4.3.6	SetListItems . . . . .	20
4.5	DynamicAbstractControl Class Reference . . . . .	21
4.5.1	Detailed Description . . . . .	23
4.5.2	Constructor & Destructor Documentation . . . . .	23

4.5.2.1	DynamicAbstractControl . . . . .	23
4.5.2.2	~DynamicAbstractControl . . . . .	24
4.5.3	Member Function Documentation . . . . .	24
4.5.3.1	actorPropertyChanged . . . . .	24
4.5.3.2	createEditor . . . . .	24
4.5.3.3	getChild . . . . .	24
4.5.3.4	getChildCount . . . . .	24
4.5.3.5	getChildIndex . . . . .	24
4.5.3.6	getDescription . . . . .	24
4.5.3.7	getDisplayName . . . . .	25
4.5.3.8	GetDynamicControlFactory . . . . .	25
4.5.3.9	GetModel . . . . .	25
4.5.3.10	getParent . . . . .	25
4.5.3.11	getValueAsString . . . . .	25
4.5.3.12	handleSubEditDestroy . . . . .	25
4.5.3.13	InitializeData . . . . .	25
4.5.3.14	InstallEventFilterOnControl . . . . .	26
4.5.3.15	isControlDoesCustomPainting . . . . .	26
4.5.3.16	isEditable . . . . .	26
4.5.3.17	NeedsPersistentEditor . . . . .	26
4.5.3.18	NotifyParentOfPreUpdate . . . . .	26
4.5.3.19	OnChildPreUpdate . . . . .	26
4.5.3.20	paintColumn . . . . .	26
4.5.3.21	PropertyAboutToChange . . . . .	26
4.5.3.22	PropertyAboutToChangePassThrough . . . . .	26
4.5.3.23	PropertyChanged . . . . .	27
4.5.3.24	PropertyChangedPassThrough . . . . .	27
4.5.3.25	SetBackgroundColor . . . . .	27
4.5.3.26	SetDynamicControlFactory . . . . .	27
4.5.3.27	SetTreeView . . . . .	27
4.5.3.28	sizeHint . . . . .	27
4.5.3.29	updateData . . . . .	27
4.5.3.30	updateEditorFromModel . . . . .	27
4.5.3.31	updateModelFromEditor . . . . .	27
4.5.4	Member Data Documentation . . . . .	28
4.5.4.1	mInitialized . . . . .	28
4.5.4.2	mModel . . . . .	28
4.5.4.3	mParent . . . . .	28
4.5.4.4	mPropContainer . . . . .	28
4.5.4.5	mPropertyTree . . . . .	28
4.6	DynamicAbstractParentControl Class Reference . . . . .	29

4.6.1	Detailed Description . . . . .	30
4.6.2	Constructor & Destructor Documentation . . . . .	31
4.6.2.1	DynamicAbstractParentControl . . . . .	31
4.6.2.2	~DynamicAbstractParentControl . . . . .	31
4.6.3	Member Function Documentation . . . . .	31
4.6.3.1	getChild . . . . .	31
4.6.3.2	getChildCount . . . . .	31
4.6.3.3	getChildIndex . . . . .	31
4.6.3.4	removeAllChildren . . . . .	31
4.6.4	Member Data Documentation . . . . .	31
4.6.4.1	mChildren . . . . .	31
4.7	DynamicArrayControl Class Reference . . . . .	32
4.7.1	Detailed Description . . . . .	34
4.7.2	Constructor & Destructor Documentation . . . . .	34
4.7.2.1	DynamicArrayControl . . . . .	34
4.7.2.2	~DynamicArrayControl . . . . .	34
4.7.3	Member Function Documentation . . . . .	34
4.7.3.1	createEditor . . . . .	34
4.7.3.2	getDescription . . . . .	34
4.7.3.3	getDisplayName . . . . .	34
4.7.3.4	getValueAsString . . . . .	34
4.7.3.5	handleSubEditDestroy . . . . .	34
4.7.3.6	InitializeData . . . . .	34
4.7.3.7	isEditable . . . . .	35
4.7.3.8	onAddClicked . . . . .	35
4.7.3.9	onClearClicked . . . . .	35
4.7.3.10	resizeChildren . . . . .	35
4.7.3.11	updateData . . . . .	35
4.8	DynamicArrayElementControl Class Reference . . . . .	36
4.8.1	Detailed Description . . . . .	37
4.8.2	Constructor & Destructor Documentation . . . . .	37
4.8.2.1	DynamicArrayElementControl . . . . .	37
4.8.2.2	~DynamicArrayElementControl . . . . .	37
4.8.3	Member Function Documentation . . . . .	38
4.8.3.1	createEditor . . . . .	38
4.8.3.2	getDescription . . . . .	38
4.8.3.3	getDisplayName . . . . .	38
4.8.3.4	GetIndex . . . . .	38
4.8.3.5	getValueAsString . . . . .	38
4.8.3.6	handleSubEditDestroy . . . . .	38
4.8.3.7	InitializeData . . . . .	38

4.8.3.8	isEditable . . . . .	38
4.8.3.9	OnChildPreUpdate . . . . .	38
4.8.3.10	onCopyClicked . . . . .	38
4.8.3.11	onDeleteClicked . . . . .	38
4.8.3.12	onShiftDownClicked . . . . .	38
4.8.3.13	onShiftUpClicked . . . . .	38
4.8.3.14	SetActive . . . . .	38
4.8.3.15	SetIndex . . . . .	39
4.8.3.16	updateData . . . . .	39
4.9	DynamicBoolControl Class Reference . . . . .	40
4.9.1	Detailed Description . . . . .	41
4.9.2	Constructor & Destructor Documentation . . . . .	41
4.9.2.1	DynamicBoolControl . . . . .	41
4.9.2.2	~DynamicBoolControl . . . . .	41
4.9.3	Member Function Documentation . . . . .	41
4.9.3.1	actorPropertyChanged . . . . .	41
4.9.3.2	createEditor . . . . .	41
4.9.3.3	getDescription . . . . .	41
4.9.3.4	getDisplayname . . . . .	42
4.9.3.5	getValueAsString . . . . .	42
4.9.3.6	handleSubEditDestroy . . . . .	42
4.9.3.7	InitializeData . . . . .	42
4.9.3.8	isEditable . . . . .	42
4.9.3.9	itemSelected . . . . .	42
4.9.3.10	updateData . . . . .	42
4.9.3.11	updateEditorFromModel . . . . .	42
4.9.3.12	updateModelFromEditor . . . . .	42
4.9.4	Member Data Documentation . . . . .	42
4.9.4.1	FALSE_LABEL . . . . .	42
4.9.4.2	TRUE_LABEL . . . . .	42
4.10	DynamicColorElementControl Class Reference . . . . .	43
4.10.1	Detailed Description . . . . .	44
4.10.2	Constructor & Destructor Documentation . . . . .	44
4.10.2.1	DynamicColorElementControl . . . . .	44
4.10.2.2	~DynamicColorElementControl . . . . .	44
4.10.3	Member Function Documentation . . . . .	44
4.10.3.1	actorPropertyChanged . . . . .	44
4.10.3.2	convertColorFloatToInt . . . . .	45
4.10.3.3	convertColorIntToFloat . . . . .	45
4.10.3.4	createEditor . . . . .	45
4.10.3.5	getDescription . . . . .	45

4.10.3.6	getDisplayName	45
4.10.3.7	getValue	45
4.10.3.8	getValueAsString	45
4.10.3.9	handleSubEditDestroy	45
4.10.3.10	InitializeData	45
4.10.3.11	isEditable	45
4.10.3.12	setValue	45
4.10.3.13	updateData	45
4.10.3.14	updateEditorFromModel	45
4.10.3.15	updateModelFromEditor	46
4.11	DynamicColorRGBAControl Class Reference	47
4.11.1	Detailed Description	48
4.11.2	Constructor & Destructor Documentation	48
4.11.2.1	DynamicColorRGBAControl	48
4.11.2.2	~DynamicColorRGBAControl	48
4.11.3	Member Function Documentation	48
4.11.3.1	actorPropertyChanged	48
4.11.3.2	addSelfToParentWidget	48
4.11.3.3	colorPickerPressed	48
4.11.3.4	createEditor	49
4.11.3.5	CreateElementControl	49
4.11.3.6	getDescription	49
4.11.3.7	getDisplayName	49
4.11.3.8	getValueAsString	49
4.11.3.9	handleSubEditDestroy	49
4.11.3.10	InitializeData	49
4.11.3.11	installEventFilterOnControl	49
4.11.3.12	isEditable	49
4.11.3.13	NeedsPersistentEditor	49
4.11.3.14	updateData	49
4.11.3.15	updateEditorFromModel	49
4.11.3.16	updateModelFromEditor	50
4.12	DynamicContainerControl Class Reference	51
4.12.1	Detailed Description	52
4.12.2	Constructor & Destructor Documentation	52
4.12.2.1	DynamicContainerControl	52
4.12.2.2	~DynamicContainerControl	52
4.12.3	Member Function Documentation	52
4.12.3.1	getDescription	52
4.12.3.2	getDisplayName	52
4.12.3.3	getValueAsString	52

4.12.3.4	InitializeData . . . . .	52
4.12.3.5	isEditable . . . . .	52
4.12.3.6	updateData . . . . .	53
4.13	DynamicControlFactory Class Reference . . . . .	54
4.13.1	Constructor & Destructor Documentation . . . . .	54
4.13.1.1	DynamicControlFactory . . . . .	54
4.13.2	Member Function Documentation . . . . .	54
4.13.2.1	CreateDynamicControl . . . . .	54
4.13.2.2	RegisterControlForDataType . . . . .	54
4.14	DynamicDoubleControl Class Reference . . . . .	55
4.14.1	Detailed Description . . . . .	56
4.14.2	Constructor & Destructor Documentation . . . . .	56
4.14.2.1	DynamicDoubleControl . . . . .	56
4.14.2.2	~DynamicDoubleControl . . . . .	56
4.14.3	Member Function Documentation . . . . .	56
4.14.3.1	actorPropertyChanged . . . . .	56
4.14.3.2	createEditor . . . . .	56
4.14.3.3	getDescription . . . . .	56
4.14.3.4	getDisplayname . . . . .	56
4.14.3.5	getValueAsString . . . . .	56
4.14.3.6	handleSubEditDestroy . . . . .	57
4.14.3.7	InitializeData . . . . .	57
4.14.3.8	isEditable . . . . .	57
4.14.3.9	updateData . . . . .	57
4.14.3.10	updateEditorFromModel . . . . .	57
4.14.3.11	updateModelFromEditor . . . . .	57
4.15	DynamicEnumControl Class Reference . . . . .	58
4.15.1	Detailed Description . . . . .	59
4.15.2	Constructor & Destructor Documentation . . . . .	59
4.15.2.1	DynamicEnumControl . . . . .	59
4.15.2.2	~DynamicEnumControl . . . . .	59
4.15.3	Member Function Documentation . . . . .	59
4.15.3.1	actorPropertyChanged . . . . .	59
4.15.3.2	createEditor . . . . .	59
4.15.3.3	getDescription . . . . .	59
4.15.3.4	getDisplayname . . . . .	59
4.15.3.5	getValueAsString . . . . .	60
4.15.3.6	handleSubEditDestroy . . . . .	60
4.15.3.7	InitializeData . . . . .	60
4.15.3.8	isEditable . . . . .	60
4.15.3.9	itemSelected . . . . .	60

4.15.3.10	updateData . . . . .	60
4.15.3.11	updateEditorFromModel . . . . .	60
4.15.3.12	updateModelFromEditor . . . . .	60
4.16	DynamicFloatControl Class Reference . . . . .	61
4.16.1	Detailed Description . . . . .	62
4.16.2	Constructor & Destructor Documentation . . . . .	62
4.16.2.1	DynamicFloatControl . . . . .	62
4.16.2.2	~DynamicFloatControl . . . . .	62
4.16.3	Member Function Documentation . . . . .	62
4.16.3.1	actorPropertyChanged . . . . .	62
4.16.3.2	createEditor . . . . .	62
4.16.3.3	getDescription . . . . .	62
4.16.3.4	getDisplayName . . . . .	62
4.16.3.5	getValueAsString . . . . .	62
4.16.3.6	handleSubEditDestroy . . . . .	63
4.16.3.7	initializeData . . . . .	63
4.16.3.8	isEditable . . . . .	63
4.16.3.9	updateData . . . . .	63
4.16.3.10	updateEditorFromModel . . . . .	63
4.16.3.11	updateModelFromEditor . . . . .	63
4.17	DynamicGroupControl Class Reference . . . . .	64
4.17.1	Detailed Description . . . . .	65
4.17.2	Constructor & Destructor Documentation . . . . .	65
4.17.2.1	DynamicGroupControl . . . . .	65
4.17.2.2	~DynamicGroupControl . . . . .	65
4.17.3	Member Function Documentation . . . . .	65
4.17.3.1	addChildControl . . . . .	65
4.17.3.2	addSelfToParentWidget . . . . .	65
4.17.3.3	getChildGroupControl . . . . .	65
4.17.3.4	getDisplayName . . . . .	65
4.17.3.5	updateData . . . . .	65
4.18	DynamicIntControl Class Reference . . . . .	66
4.18.1	Detailed Description . . . . .	67
4.18.2	Constructor & Destructor Documentation . . . . .	67
4.18.2.1	DynamicIntControl . . . . .	67
4.18.2.2	~DynamicIntControl . . . . .	67
4.18.3	Member Function Documentation . . . . .	67
4.18.3.1	actorPropertyChanged . . . . .	67
4.18.3.2	createEditor . . . . .	67
4.18.3.3	getDescription . . . . .	67
4.18.3.4	getDisplayName . . . . .	67

4.18.3.5	getValueAsString . . . . .	67
4.18.3.6	handleSubEditDestroy . . . . .	68
4.18.3.7	InitializeData . . . . .	68
4.18.3.8	isEditable . . . . .	68
4.18.3.9	updateData . . . . .	68
4.18.3.10	updateEditorFromModel . . . . .	68
4.18.3.11	updateModelFromEditor . . . . .	68
4.19	DynamicLabelControl Class Reference . . . . .	69
4.19.1	Detailed Description . . . . .	70
4.19.2	Constructor & Destructor Documentation . . . . .	70
4.19.2.1	DynamicLabelControl . . . . .	70
4.19.2.2	~DynamicLabelControl . . . . .	70
4.19.3	Member Function Documentation . . . . .	70
4.19.3.1	getDescription . . . . .	70
4.19.3.2	getDisplayName . . . . .	70
4.19.3.3	getValueAsString . . . . .	70
4.19.3.4	InitializeData . . . . .	70
4.19.3.5	setDisplayValues . . . . .	70
4.19.3.6	updateData . . . . .	70
4.20	DynamicLongControl Class Reference . . . . .	72
4.20.1	Detailed Description . . . . .	73
4.20.2	Constructor & Destructor Documentation . . . . .	73
4.20.2.1	DynamicLongControl . . . . .	73
4.20.2.2	~DynamicLongControl . . . . .	73
4.20.3	Member Function Documentation . . . . .	73
4.20.3.1	actorPropertyChanged . . . . .	73
4.20.3.2	createEditor . . . . .	73
4.20.3.3	getDescription . . . . .	73
4.20.3.4	getDisplayName . . . . .	73
4.20.3.5	getValueAsString . . . . .	73
4.20.3.6	handleSubEditDestroy . . . . .	74
4.20.3.7	InitializeData . . . . .	74
4.20.3.8	isEditable . . . . .	74
4.20.3.9	updateData . . . . .	74
4.20.3.10	updateEditorFromModel . . . . .	74
4.20.3.11	updateModelFromEditor . . . . .	74
4.21	DynamicNoUpdateParentControl Class Reference . . . . .	75
4.21.1	Member Function Documentation . . . . .	76
4.21.1.1	updateData . . . . .	76
4.22	DynamicResourceControlBase Class Reference . . . . .	77
4.22.1	Detailed Description . . . . .	78

4.22.2	Constructor & Destructor Documentation . . . . .	78
4.22.2.1	DynamicResourceControlBase . . . . .	78
4.22.2.2	~DynamicResourceControlBase . . . . .	78
4.22.3	Member Function Documentation . . . . .	79
4.22.3.1	actorPropertyChanged . . . . .	79
4.22.3.2	clearPressed . . . . .	79
4.22.3.3	createEditor . . . . .	79
4.22.3.4	getCurrentResource . . . . .	79
4.22.3.5	getDescription . . . . .	79
4.22.3.6	getDisplayName . . . . .	79
4.22.3.7	GetProperty . . . . .	79
4.22.3.8	getValueAsString . . . . .	79
4.22.3.9	handleSubEditDestroy . . . . .	79
4.22.3.10	InitializeData . . . . .	79
4.22.3.11	installEventFilterOnControl . . . . .	79
4.22.3.12	isEditable . . . . .	79
4.22.3.13	updateData . . . . .	80
4.22.3.14	updateEditorFromModel . . . . .	80
4.22.3.15	updateModelFromEditor . . . . .	80
4.22.3.16	useCurrentPressed . . . . .	80
4.23	DynamicStringControl Class Reference . . . . .	81
4.23.1	Detailed Description . . . . .	82
4.23.2	Constructor & Destructor Documentation . . . . .	82
4.23.2.1	DynamicStringControl . . . . .	82
4.23.2.2	~DynamicStringControl . . . . .	82
4.23.3	Member Function Documentation . . . . .	82
4.23.3.1	actorPropertyChanged . . . . .	82
4.23.3.2	createEditor . . . . .	82
4.23.3.3	getDescription . . . . .	82
4.23.3.4	getDisplayName . . . . .	82
4.23.3.5	getValueAsString . . . . .	82
4.23.3.6	handleSubEditDestroy . . . . .	83
4.23.3.7	InitializeData . . . . .	83
4.23.3.8	isEditable . . . . .	83
4.23.3.9	updateData . . . . .	83
4.23.3.10	updateEditorFromModel . . . . .	83
4.23.3.11	updateModelFromEditor . . . . .	83
4.24	DynamicVecNControl< PropertyType > Class Template Reference . . . . .	84
4.24.1	Detailed Description . . . . .	85
4.24.2	Member Typedef Documentation . . . . .	85
4.24.2.1	PropertyGetValueType . . . . .	85

4.24.3	Constructor & Destructor Documentation	85
4.24.3.1	DynamicVecNControl	85
4.24.3.2	~DynamicVecNControl	85
4.24.4	Member Function Documentation	85
4.24.4.1	CreateElementControl	85
4.24.4.2	getDescription	85
4.24.4.3	getDisplayName	85
4.24.4.4	getValueAsString	85
4.24.4.5	initializeData	85
4.24.4.6	isEditable	85
4.25	DynamicVectorElementControl Class Reference	86
4.25.1	Detailed Description	87
4.25.2	Constructor & Destructor Documentation	88
4.25.2.1	DynamicVectorElementControl	88
4.25.2.2	DynamicVectorElementControl	88
4.25.2.3	DynamicVectorElementControl	88
4.25.2.4	DynamicVectorElementControl	88
4.25.2.5	DynamicVectorElementControl	88
4.25.2.6	DynamicVectorElementControl	88
4.25.2.7	DynamicVectorElementControl	88
4.25.2.8	DynamicVectorElementControl	88
4.25.2.9	DynamicVectorElementControl	88
4.25.2.10	~DynamicVectorElementControl	88
4.25.3	Member Function Documentation	88
4.25.3.1	actorPropertyChanged	88
4.25.3.2	createEditor	89
4.25.3.3	getDescription	89
4.25.3.4	getDisplayName	89
4.25.3.5	getValue	89
4.25.3.6	getValueAsString	89
4.25.3.7	handleSubEditDestroy	89
4.25.3.8	initializeData	89
4.25.3.9	isEditable	89
4.25.3.10	setValue	89
4.25.3.11	updateData	89
4.25.3.12	updateEditorFromModel	89
4.25.3.13	updateModelFromEditor	90
4.26	GLWidgetFactory Class Reference	91
4.26.1	Detailed Description	91
4.26.2	Constructor & Destructor Documentation	91
4.26.2.1	GLWidgetFactory	91

4.26.2.2	~GLWidgetFactory . . . . .	91
4.26.3	Member Function Documentation . . . . .	91
4.26.3.1	CreateWidget . . . . .	91
4.27	LibraryPathsEditor Class Reference . . . . .	92
4.27.1	Constructor & Destructor Documentation . . . . .	92
4.27.1.1	LibraryPathsEditor . . . . .	92
4.27.1.2	~LibraryPathsEditor . . . . .	93
4.27.2	Member Function Documentation . . . . .	93
4.27.2.1	AnyItemsSelected . . . . .	93
4.27.2.2	RefreshButtons . . . . .	93
4.27.2.3	ShiftPathDown . . . . .	93
4.27.2.4	ShiftPathUp . . . . .	93
4.27.2.5	SpawnDeleteConfirmation . . . . .	93
4.27.2.6	SpawnFileBrowser . . . . .	93
4.28	OSGAdapterWidget Class Reference . . . . .	94
4.28.1	Constructor & Destructor Documentation . . . . .	95
4.28.1.1	OSGAdapterWidget . . . . .	95
4.28.1.2	~OSGAdapterWidget . . . . .	95
4.28.2	Member Function Documentation . . . . .	95
4.28.2.1	GetGraphicsWindow . . . . .	95
4.28.2.2	GetGraphicsWindow . . . . .	95
4.28.2.3	initializeGL . . . . .	95
4.28.2.4	keyPressEvent . . . . .	95
4.28.2.5	keyReleaseEvent . . . . .	95
4.28.2.6	mouseMoveEvent . . . . .	95
4.28.2.7	mousePressEvent . . . . .	95
4.28.2.8	mouseReleaseEvent . . . . .	95
4.28.2.9	paintGL . . . . .	95
4.28.2.10	paintGLImpl . . . . .	95
4.28.2.11	resizeGL . . . . .	95
4.28.2.12	resizeGLImpl . . . . .	95
4.28.2.13	SetGraphicsWindow . . . . .	95
4.28.2.14	ThreadedInitializeGL . . . . .	95
4.28.2.15	ThreadedMakeCurrent . . . . .	95
4.28.2.16	ThreadedUpdateGL . . . . .	95
4.28.2.17	wheelEvent . . . . .	95
4.28.3	Member Data Documentation . . . . .	95
4.28.3.1	mDoResize . . . . .	95
4.28.3.2	mDrawOnSeparateThread . . . . .	95
4.28.3.3	mGraphicsWindow . . . . .	95
4.28.3.4	mThreadGLContext . . . . .	95

4.28.3.5	mTimer . . . . .	95
4.29	OSGGraphicsWindowQt Class Reference . . . . .	96
4.29.1	Member Typedef Documentation . . . . .	97
4.29.1.1	BaseClass . . . . .	97
4.29.2	Constructor & Destructor Documentation . . . . .	97
4.29.2.1	OSGGraphicsWindowQt . . . . .	97
4.29.2.2	~OSGGraphicsWindowQt . . . . .	97
4.29.3	Member Function Documentation . . . . .	97
4.29.3.1	checkEvents . . . . .	97
4.29.3.2	className . . . . .	97
4.29.3.3	closeImplementation . . . . .	97
4.29.3.4	GetQGLWidget . . . . .	97
4.29.3.5	GetQGLWidget . . . . .	97
4.29.3.6	getWindowRectangle . . . . .	97
4.29.3.7	grabFocus . . . . .	97
4.29.3.8	grabFocusIfPointerInWindow . . . . .	97
4.29.3.9	isRealizedImplementation . . . . .	97
4.29.3.10	isSameKindAs . . . . .	97
4.29.3.11	libraryName . . . . .	97
4.29.3.12	makeCurrentImplementation . . . . .	97
4.29.3.13	realizeImplementation . . . . .	97
4.29.3.14	releaseContextImplementation . . . . .	97
4.29.3.15	requestClose . . . . .	97
4.29.3.16	resizedImplementation . . . . .	97
4.29.3.17	setCursor . . . . .	97
4.29.3.18	SetQGLWidget . . . . .	97
4.29.3.19	setWindowDecorationImplementation . . . . .	97
4.29.3.20	setWindowName . . . . .	98
4.29.3.21	setWindowRectangleImplementation . . . . .	98
4.29.3.22	swapBuffersImplementation . . . . .	98
4.29.3.23	useCursor . . . . .	98
4.29.3.24	valid . . . . .	98
4.30	ProjectContextDialog Class Reference . . . . .	99
4.30.1	Constructor & Destructor Documentation . . . . .	99
4.30.1.1	ProjectContextDialog . . . . .	99
4.30.1.2	~ProjectContextDialog . . . . .	99
4.30.2	Member Function Documentation . . . . .	99
4.30.2.1	getProjectPath . . . . .	99
4.30.2.2	spawnFileBrowser . . . . .	99
4.31	PropertyEditorDelegate Class Reference . . . . .	100
4.31.1	Detailed Description . . . . .	100

4.31.2	Constructor & Destructor Documentation . . . . .	100
4.31.2.1	PropertyEditorDelegate . . . . .	100
4.31.2.2	~PropertyEditorDelegate . . . . .	100
4.31.3	Member Function Documentation . . . . .	101
4.31.3.1	createEditor . . . . .	101
4.31.3.2	drawDecoration . . . . .	101
4.31.3.3	eventFilter . . . . .	101
4.31.3.4	paint . . . . .	101
4.31.3.5	setEditorData . . . . .	101
4.31.3.6	setModelData . . . . .	101
4.31.3.7	sizeHint . . . . .	101
4.32	PropertyEditorModel Class Reference . . . . .	102
4.32.1	Detailed Description . . . . .	103
4.32.2	Constructor & Destructor Documentation . . . . .	103
4.32.2.1	PropertyEditorModel . . . . .	103
4.32.2.2	~PropertyEditorModel . . . . .	103
4.32.3	Member Function Documentation . . . . .	103
4.32.3.1	columnCount . . . . .	103
4.32.3.2	data . . . . .	103
4.32.3.3	flags . . . . .	103
4.32.3.4	GetAbstractControlFromIndex . . . . .	103
4.32.3.5	GetRootControl . . . . .	103
4.32.3.6	hasChildren . . . . .	104
4.32.3.7	headerData . . . . .	104
4.32.3.8	index . . . . .	104
4.32.3.9	IndexOf . . . . .	104
4.32.3.10	insertRows . . . . .	104
4.32.3.11	isEditable . . . . .	104
4.32.3.12	parent . . . . .	104
4.32.3.13	removeRows . . . . .	104
4.32.3.14	rowCount . . . . .	104
4.32.3.15	setData . . . . .	104
4.32.3.16	setDescription . . . . .	104
4.32.3.17	SetRootControl . . . . .	104
4.32.4	Friends And Related Function Documentation . . . . .	104
4.32.4.1	DynamicAbstractParentControl . . . . .	104
4.32.4.2	DynamicGroupControl . . . . .	104
4.32.4.3	PropertyEditor . . . . .	104
4.33	PropertyEditorTreeView Class Reference . . . . .	105
4.33.1	Detailed Description . . . . .	106
4.33.2	Constructor & Destructor Documentation . . . . .	106

4.33.2.1	PropertyEditorTreeView	106
4.33.2.2	~PropertyEditorTreeView	106
4.33.3	Member Function Documentation	106
4.33.3.1	closeEditor	106
4.33.3.2	currentChanged	106
4.33.3.3	isReadOnly	106
4.33.3.4	reset	106
4.33.3.5	selectionChanged	106
4.33.3.6	setRoot	106
4.33.4	Member Data Documentation	106
4.33.4.1	ROW_COLOR_EVEN	106
4.33.4.2	ROW_COLOR_ODD	106
4.34	QtGuiWindowSystemWrapper Class Reference	107
4.34.1	Constructor & Destructor Documentation	107
4.34.1.1	QtGuiWindowSystemWrapper	107
4.34.1.2	~QtGuiWindowSystemWrapper	107
4.34.2	Member Function Documentation	107
4.34.2.1	createGraphicsContext	107
4.34.2.2	EnableQtGUIWrapper	107
4.34.2.3	getNumScreens	107
4.34.2.4	getScreenResolution	107
4.34.2.5	SetGLWidgetFactory	107
4.34.2.6	setScreenRefreshRate	107
4.34.2.7	setScreenResolution	107
4.35	QtKeyboardMap Class Reference	108
4.35.1	Constructor & Destructor Documentation	108
4.35.1.1	QtKeyboardMap	108
4.35.1.2	~QtKeyboardMap	108
4.35.2	Member Function Documentation	108
4.35.2.1	remapKey	108
4.36	QtKeyboardMap Class Reference	109
4.36.1	Constructor & Destructor Documentation	109
4.36.1.1	QtKeyboardMap	109
4.36.1.2	~QtKeyboardMap	109
4.36.2	Member Function Documentation	109
4.36.2.1	remapKey	109
4.37	SubQComboBox Class Reference	110
4.37.1	Detailed Description	110
4.37.2	Constructor & Destructor Documentation	110
4.37.2.1	SubQComboBox	110
4.37.2.2	~SubQComboBox	110

4.38	SubQLabel Class Reference	112
4.38.1	Detailed Description	112
4.38.2	Constructor & Destructor Documentation	112
4.38.2.1	SubQLabel	112
4.38.2.2	~SubQLabel	113
4.39	SubQLineEdit Class Reference	114
4.39.1	Detailed Description	114
4.39.2	Constructor & Destructor Documentation	115
4.39.2.1	SubQLineEdit	115
4.39.2.2	~SubQLineEdit	115
4.40	SubQPushButton Class Reference	116
4.40.1	Detailed Description	116
4.40.2	Constructor & Destructor Documentation	116
4.40.2.1	SubQPushButton	116
4.40.2.2	~SubQPushButton	116
4.41	SubQSpinBox Class Reference	118
4.41.1	Detailed Description	118
4.41.2	Constructor & Destructor Documentation	118
4.41.2.1	SubQSpinBox	118
4.41.2.2	~SubQSpinBox	119
4.42	SubQWidget Class Reference	120
4.42.1	Detailed Description	120
4.42.2	Constructor & Destructor Documentation	120
4.42.2.1	SubQWidget	120
4.42.2.2	~SubQWidget	121
4.42.3	Member Function Documentation	121
4.42.3.1	addManagedChild	121
4.43	ViewWindow Class Reference	122
4.43.1	Detailed Description	122
4.43.2	Constructor & Destructor Documentation	123
4.43.2.1	ViewWindow	123
4.43.2.2	~ViewWindow	123
4.43.3	Member Function Documentation	123
4.43.3.1	GetGraphicsWindow	123
4.43.3.2	GetGraphicsWindow	123
4.43.3.3	glDraw	123
4.43.3.4	initializeGL	123
4.43.3.5	keyPressEvent	123
4.43.3.6	keyReleaseEvent	123
4.43.3.7	mouseMoveEvent	123
4.43.3.8	mousePressEvent	123

4.43.3.9	mouseReleaseEvent . . . . .	123
4.43.3.10	OnMessage . . . . .	123
4.43.3.11	paintGL . . . . .	123
4.43.3.12	paintGLImpl . . . . .	123
4.43.3.13	resizeGL . . . . .	123
4.43.3.14	resizeGLImpl . . . . .	123
4.43.3.15	SetGraphicsWindow . . . . .	123
4.43.3.16	ThreadedInitializeGL . . . . .	123
4.43.3.17	ThreadedMakeCurrent . . . . .	123
4.43.3.18	ThreadedUpdateGL . . . . .	123
4.43.3.19	wheelEvent . . . . .	123
4.43.4	Member Data Documentation . . . . .	123
4.43.4.1	mDoResize . . . . .	123
4.43.4.2	mDrawOnSeparateThread . . . . .	123
4.43.4.3	mGraphicsWindow . . . . .	123
4.43.4.4	mThreadGLContext . . . . .	123
4.43.4.5	mTimer . . . . .	123
<b>5</b>	<b>File Documentation</b>	<b>125</b>
5.1	baselibrarylisteditor.cpp File Reference . . . . .	125
5.2	baselibrarylisteditor.h File Reference . . . . .	126
5.3	basepropertyeditor.cpp File Reference . . . . .	127
5.4	basepropertyeditor.h File Reference . . . . .	128
5.5	deltastepper.cpp File Reference . . . . .	129
5.6	deltastepper.h File Reference . . . . .	130
5.7	dialoglistselection.cpp File Reference . . . . .	131
5.8	dialoglistselection.h File Reference . . . . .	132
5.9	dynamicabstractcontrol.cpp File Reference . . . . .	133
5.10	dynamicabstractcontrol.h File Reference . . . . .	134
5.11	dynamicabstractparentcontrol.cpp File Reference . . . . .	135
5.12	dynamicabstractparentcontrol.h File Reference . . . . .	136
5.13	dynamicarraycontrol.cpp File Reference . . . . .	137
5.14	dynamicarraycontrol.h File Reference . . . . .	138
5.15	dynamicarrayelementcontrol.cpp File Reference . . . . .	139
5.16	dynamicarrayelementcontrol.h File Reference . . . . .	140
5.17	dynamicboolcontrol.cpp File Reference . . . . .	141
5.18	dynamicboolcontrol.h File Reference . . . . .	142
5.19	dynamiccolorelementcontrol.cpp File Reference . . . . .	143
5.20	dynamiccolorelementcontrol.h File Reference . . . . .	144
5.21	dynamiccolorrrgbacontrol.cpp File Reference . . . . .	145
5.22	dynamiccolorrrgbacontrol.h File Reference . . . . .	146

5.23	dynamiccontainercontrol.cpp File Reference . . . . .	147
5.24	dynamiccontainercontrol.h File Reference . . . . .	148
5.25	dynamicdoublecontrol.cpp File Reference . . . . .	149
5.26	dynamicdoublecontrol.h File Reference . . . . .	150
5.27	dynamicenumcontrol.cpp File Reference . . . . .	151
5.28	dynamicenumcontrol.h File Reference . . . . .	152
5.29	dynamicfloatcontrol.cpp File Reference . . . . .	153
5.30	dynamicfloatcontrol.h File Reference . . . . .	154
5.31	dynamicgroupcontrol.cpp File Reference . . . . .	155
5.32	dynamicgroupcontrol.h File Reference . . . . .	156
5.33	dynamicintcontrol.cpp File Reference . . . . .	157
5.34	dynamicintcontrol.h File Reference . . . . .	158
5.35	dynamiclabelcontrol.cpp File Reference . . . . .	159
5.36	dynamiclabelcontrol.h File Reference . . . . .	160
5.37	dynamiclongcontrol.cpp File Reference . . . . .	161
5.38	dynamiclongcontrol.h File Reference . . . . .	162
5.39	dynamicresourcecontrolbase.cpp File Reference . . . . .	163
5.40	dynamicresourcecontrolbase.h File Reference . . . . .	164
5.41	dynamicstringcontrol.cpp File Reference . . . . .	165
5.42	dynamicstringcontrol.h File Reference . . . . .	166
5.43	dynamicsubwidgets.h File Reference . . . . .	167
5.44	dynamicvecncontrol.h File Reference . . . . .	168
5.45	dynamicvectorelementcontrol.cpp File Reference . . . . .	169
5.46	dynamicvectorelementcontrol.h File Reference . . . . .	170
5.47	export.h File Reference . . . . .	171
	5.47.1 Define Documentation . . . . .	171
	5.47.1.1 DT_QT_EXPORT . . . . .	171
5.48	glwidgetfactory.h File Reference . . . . .	172
5.49	librarypathseditor.cpp File Reference . . . . .	173
	5.49.1 Enumeration Type Documentation . . . . .	173
	5.49.1.1 "@0 . . . . .	173
5.50	librarypathseditor.h File Reference . . . . .	174
5.51	mainpage.h File Reference . . . . .	175
	5.51.1 Detailed Description . . . . .	175
5.52	osgadapterwidget.cpp File Reference . . . . .	176
5.53	osgadapterwidget.h File Reference . . . . .	177
5.54	osggraphicswindowqt.cpp File Reference . . . . .	178
5.55	osggraphicswindowqt.h File Reference . . . . .	179
5.56	projectcontextdialog.cpp File Reference . . . . .	180
5.57	projectcontextdialog.h File Reference . . . . .	181
5.58	propertyeditordelegate.cpp File Reference . . . . .	182

5.59	propertyeditordelegate.h File Reference	183
5.60	propertyeditormodel.cpp File Reference	184
5.61	propertyeditormodel.h File Reference	185
5.62	propertyeditortreeview.cpp File Reference	186
5.63	propertyeditortreeview.h File Reference	187
5.64	qtguiwindowssystemwrapper.cpp File Reference	188
5.65	qtguiwindowssystemwrapper.h File Reference	189
5.66	typedefs.h File Reference	190
5.66.1	Typedef Documentation	190
5.66.1.1	ActorPropertyRefPtr	190
5.66.1.2	ActorProxyRefPtr	190
5.66.1.3	ActorProxyRefPtrVector	190
5.66.2	Function Documentation	190
5.66.2.1	Q_DECLARE_METATYPE	190
5.66.2.2	Q_DECLARE_METATYPE	190
5.67	viewwindow.cpp File Reference	191
5.67.1	Variable Documentation	191
5.67.1.1	STATIC_KEY_MAP	191
5.68	viewwindow.h File Reference	192



## Main Page

---

Delta3D is an Open Source engine which can be used for games, simulations, or other graphical applications.

The Delta3D framework exists as a number of modules, each sitting in its own library, enclosed within its own namespace. At the very core lies the dtCore library. This contains basic, low-level functionality which is mostly required for all 3D applications written in C++.

Around and alongside this sit other supporting libraries, such as dtUtil (containing reusable features which are useful for most applications), dtTerrain (for rendering terrain databases), dtGame, dtNet, etc.

Extensive online documentation is available from the Delta3D Docs section to help in using Delta3D.

The project's original reference guides generated by Doxygen from the source code may be viewed at the Delta3D API Documentation section.

To download source code, binaries, dependencies and sample datasets visit the Delta3D Downloads page.

For more about dependencies see the Delta3D Dependencies page.

The documentation you are looking at can be downloaded from [www.3draum.ch](http://www.3draum.ch).

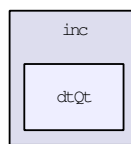
Enjoy!



## Directory Documentation

---

### 2.1 inc/dtQt/ Directory Reference

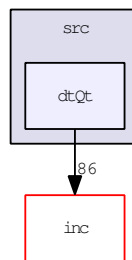


#### Files

- file **baselibrarylisteditor.h**
- file **basepropertyeditor.h**
- file **deltastepper.h**
- file **dialoglistselection.h**
- file **dynamicabstractcontrol.h**
- file **dynamicabstractparentcontrol.h**
- file **dynamicarraycontrol.h**
- file **dynamicarrayelementcontrol.h**
- file **dynamicboolcontrol.h**
- file **dynamiccolorelementcontrol.h**
- file **dynamiccolorrrgbacontrol.h**
- file **dynamiccontainercontrol.h**
- file **dynamicdoublecontrol.h**
- file **dynamicenumcontrol.h**
- file **dynamicfloatcontrol.h**
- file **dynamicgroupcontrol.h**
- file **dynamicintcontrol.h**
- file **dynamiclabelcontrol.h**
- file **dynamiclongcontrol.h**
- file **dynamicresourcecontrolbase.h**
- file **dynamicstringcontrol.h**
- file **dynamicsubwidgets.h**
- file **dynamicvecncontrol.h**
- file **dynamicvectorelementcontrol.h**
- file **export.h**
- file **glwidgetfactory.h**
- file **librarypathseditor.h**
- file **mainpage.h**
- file **osgadapterwidget.h**
- file **osggraphicswindowqt.h**
- file **projectcontextdialog.h**

- file **propertyeditordelegate.h**
- file **propertyeditormodel.h**
- file **propertyeditortreeview.h**
- file **qtguiwindowssystemwrapper.h**
- file **typedefs.h**
- file **viewwindow.h**

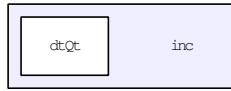
## 2.2 src/dtQt/ Directory Reference



### Files

- file **baselibrarylisteditor.cpp**
- file **basepropertyeditor.cpp**
- file **deltastepper.cpp**
- file **dialoglistselection.cpp**
- file **dynamicabstractcontrol.cpp**
- file **dynamicabstractparentcontrol.cpp**
- file **dynamicarraycontrol.cpp**
- file **dynamicarrayelementcontrol.cpp**
- file **dynamicboolcontrol.cpp**
- file **dynamiccolorelementcontrol.cpp**
- file **dynamiccolorrrgbacontrol.cpp**
- file **dynamiccontainercontrol.cpp**
- file **dynamicdoublecontrol.cpp**
- file **dynamicenumcontrol.cpp**
- file **dynamicfloatcontrol.cpp**
- file **dynamicgroupcontrol.cpp**
- file **dynamicintcontrol.cpp**
- file **dynamiclabelcontrol.cpp**
- file **dynamiclongcontrol.cpp**
- file **dynamicresourcecontrolbase.cpp**
- file **dynamicstringcontrol.cpp**
- file **dynamicvectorelementcontrol.cpp**
- file **librarypathseditor.cpp**
- file **osgadapterwidget.cpp**
- file **osggraphicswindowqt.cpp**
- file **projectcontextdialog.cpp**
- file **propertyeditordelegate.cpp**
- file **propertyeditormodel.cpp**
- file **propertyeditortreeview.cpp**
- file **qtguiwindowssystemwrapper.cpp**
- file **viewwindow.cpp**

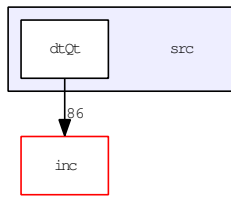
## 2.3 inc/ Directory Reference



### Directories

- directory **dtQt**

## 2.4 src/ Directory Reference



### Directories

- directory **dtQt**



# Namespace Documentation

---

## 3.1 dtDAL Namespace Reference

## 3.2 dtQt Namespace Reference

Contains classes that help integrate Delta3D with Qt.

### Classes

- class **BaseLibraryListEditor**
- class **BasePropertyEditor**  
*This class is the property editor for displaying and editing properties of selected objects.*
- class **DeltaStepper**
- class **DialogListSelection**  
*This is a generic dialog box for presenting a list of items for the user to choose from.*
- class **DynamicAbstractControl**  
*This is the base class for all the dynamic controls that are used in the property editor.*
- class **DynamicAbstractParentControl**  
*This is a base class for any dynamic control that has children.*
- class **DynamicArrayControl**  
*This is the dynamic control for the float data type - used in the property editor.*
- class **DynamicArrayElementControl**  
*This is the dynamic control for the float data type - used in the property editor.*
- class **DynamicBoolControl**  
*This is the dynamic control for the bool data type - used in the property editor.*
- class **DynamicColorElementControl**  
*This is the a sub control used by the various color property classes.*
- class **DynamicColorRGBAControl**  
*This is the dynamic control for the an RGBA Color picker - used in the property editor It adds a group of child elements to the tree, since you can't edit 3 things in one control easily.*
- class **DynamicContainerControl**  
*This is the dynamic control for the float data type - used in the property editor.*
- class **DynamicControlFactory**
- class **DynamicDoubleControl**  
*This is the dynamic control for the double data type - used in the property editor.*
- class **DynamicEnumControl**  
*This is the dynamic control for the enum data type - used in the property editor.*
- class **DynamicFloatControl**  
*This is the dynamic control for the float data type - used in the property editor.*
- class **DynamicGroupControl**  
*This is the dynamic control for the Group data type - used in the property editor The primary purpose of the group control is to provide a visual grouping of property types so that they aren't all laid out together.*
- class **DynamicIntControl**  
*This is the dynamic control for the int data type - used in the property editor.*
- class **DynamicLabelControl**

*This is the dynamic control that is a bit odd.*

- class **DynamicLongControl**

*This is the dynamic control for the long data type - used in the property editor.*

- class **DynamicNoUpdateParentControl**

- class **DynamicResourceControlBase**

*This is the resource actor property.*

- class **DynamicStringControl**

*This is the dynamic control for the String data type - used in the property editor.*

- class **DynamicVecNControl**

*This is the dynamic control for the N dimensional vector data type - used in the property editor It adds a group of child elements to the tree, since you can't edit multiple things in one control easily.*

- class **DynamicVectorElementControl**

*This is the a sub control used by the various vector property classes.*

- class **GLWidgetFactory**

*Abstract factory interface used to create specialized **OSGAdapterWidget** (p. 94) for custom application requirements.*

- class **LibraryPathsEditor**

- class **OSGAdapterWidget**

- class **OSGGraphicsWindowQt**

- class **ProjectContextDialog**

- class **PropertyEditorDelegate**

*This class is a delegate for editing commands between the model and the dynamic control.*

- class **PropertyEditorModel**

*This class is the model used to represent the properties for a selected object.*

- class **PropertyEditorTreeView**

*This class is the tree control for the properties of a proxy.*

- class **QtGuiWindowSystemWrapper**

- class **QtKeyboardMap**

- class **SubQComboBox**

*Subclass of QComboBox.*

- class **SubQLabel**

*Subclass of QLabel.*

- class **SubQLineEdit**

*This is collection of subclasses of QWidgets that updates a dynamic abstract control on destruction.*

- class **SubQPushButton**

*Subclass of QPushButton.*

- class **SubQSpinBox**

*Subclass of QSpinBox.*

- class **SubQWidget**

*Subclass of QWidget.*

- class **ViewWindow**

*Little class used to hold the Delta3D rendering surface.*

## Variables

- const unsigned int **NUM\_DECIMAL\_DIGITS\_DOUBLE** = 15
- const unsigned int **NUM\_DECIMAL\_DIGITS\_FLOAT** = 8
- static **QtKeyboardMap** **STATIC\_KEY\_MAP**

### 3.2.1 Detailed Description

Contains classes that help integrate Delta3D with Qt.

### 3.2.2 Variable Documentation

**3.2.2.1** const unsigned int **NUM\_DECIMAL\_DIGITS\_DOUBLE** = 15

**3.2.2.2** const unsigned int **NUM\_DECIMAL\_DIGITS\_FLOAT** = 8

**3.2.2.3** **QtKeyboardMap** **STATIC\_KEY\_MAP** [static]

## Class Documentation

---

### 4.1 BaseLibraryListEditor Class Reference

```
#include <inc/dtQt/baselibrarylisteditor.h>
```

#### Public Types

- enum **Errors** { **ERROR\_LIB\_NOT\_LOADED** = 0, **ERROR\_OBJECTS\_IN\_LIB\_EXIST**, **ERROR\_INVALID\_LIB**, **ERROR\_UNKNOWN** }

#### Public Slots

- void **EnableButtons** (int row)  
*Received when a library is currently selected.*
- void **HandleFailure** (const int code, const std::string &errorMsg="")  
*Handle a deletion failure.*
- virtual void **ShiftLibraryDown** ()  
*Shift the current library down 1 position.*
- virtual void **ShiftLibraryUp** ()  
*Shift the current library up 1 position.*
- virtual void **SpawnDeleteConfirmation** ()=0  
*Confirm deletion of libraries.*
- virtual void **SpawnFileBrowser** ()=0  
*Pop up the file browser for libraries and adds it to the list.*

#### Public Member Functions

- **BaseLibraryListEditor** (QWidget \*parent=NULL)  
*Constructor.*
- virtual ~**BaseLibraryListEditor** ()  
*Destructor.*
- void **showEvent** (QShowEvent \*)

## Protected Member Functions

- QListWidget & **GetLibraryListWidget** ()
- virtual void **GetLibraryNames** (std::vector< QListWidgetItem \* > &items) const =0
- void **RefreshLibraries** ()
- std::pair< std::string, std::string > **SelectLibraryToOpen** (const std::string &startingDir)

*Call this from SpawnFileBrowser to get a library to use.*

### 4.1.1 Member Enumeration Documentation

#### 4.1.1.1 enum Errors

Enumerator:

***ERROR\_LIB\_NOT\_LOADED***  
***ERROR\_OBJECTS\_IN\_LIB\_EXIST***  
***ERROR\_INVALID\_LIB***  
***ERROR\_UNKNOWN***

### 4.1.2 Constructor & Destructor Documentation

#### 4.1.2.1 BaseLibraryListEditor (QWidget \* *parent* = NULL)

Constructor.

#### 4.1.2.2 ~BaseLibraryListEditor () [virtual]

Destructor.

### 4.1.3 Member Function Documentation

#### 4.1.3.1 void EnableButtons (int *row*) [slot]

Received when a library is currently selected.

#### 4.1.3.2 QListWidget& GetLibraryListWidget () [inline, protected]

#### 4.1.3.3 virtual void GetLibraryNames (std::vector< QListWidgetItem \* > & *items*) const [protected, pure virtual]

#### 4.1.3.4 void HandleFailure (const int *code*, const std::string & *errorMsg* = "") [slot]

Handle a deletion failure.

#### 4.1.3.5 void RefreshLibraries () [protected]

#### 4.1.3.6 std::pair< std::string, std::string > SelectLibraryToOpen (const std::string & *startingDir*) [protected]

Call this from SpawnFileBrowser to get a library to use. It returns the system independent name of the library first then the full path to the library.

#### 4.1.3.7 void ShiftLibraryDown () [virtual, slot]

Shift the current library down 1 position.

#### 4.1.3.8 void ShiftLibraryUp () [virtual, slot]

Shift the current library up 1 position.

#### 4.1.3.9 void showEvent (QShowEvent \*)

#### 4.1.3.10 virtual void SpawnDeleteConfirmation () [pure virtual, slot]

Confirm deletion of libraries.

#### 4.1.3.11 virtual void SpawnFileBrowser () [pure virtual, slot]

Pop up the file browser for libraries and adds it to the list. You can call SelectLibraryToOpen to pick the file

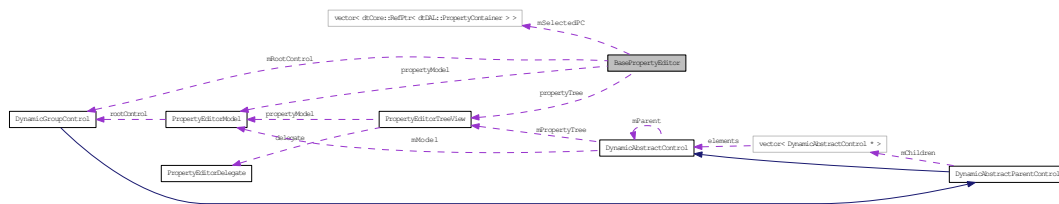
The documentation for this class was generated from the following files:

- **baselibrarylisteditor.h**
- **baselibrarylisteditor.cpp**

## 4.2 BasePropertyEditor Class Reference

This class is the property editor for displaying and editing properties of selected objects.

#include <inc/dtQt/basepropertyeditor.h> Collaboration diagram for BasePropertyEditor:



### Public Types

- typedef std::vector< dtCore::RefPtr< dtDAL::PropertyContainer > > **PropertyContainerRefPtrVector**

### Public Slots

- void **ActorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)
- void **HandlePropertyContainersSelected** (**PropertyContainerRefPtrVector** &actors)  
*Handles the actor selection changed event message from EditorEvents.*
- virtual void **PropertyAboutToChangeFromControl** (dtDAL::PropertyContainer &, dtDAL::ActorProperty &, const std::string &oldValue, const std::string &newValue)=0
- virtual void **PropertyChangedFromControl** (dtDAL::PropertyContainer &, dtDAL::ActorProperty &)=0
- void **ProxyNameChanged** (dtDAL::ActorProxy &pc, std::string oldName)

### Public Member Functions

- **BasePropertyEditor** (QMainWindow \*parent)  
*Constructor.*
- virtual ~**BasePropertyEditor** ()  
*Destructor.*
- void **CommitCurrentEdits** ()  
*Convenience method for making the property editor commit its changes.*
- **DynamicControlFactory** & **GetDynamicControlFactory** ()
- void **GetSelectedPropertyContainers** (std::vector< dtDAL::PropertyContainer \* > &toFill)  
*Fills a vector with pointers to all the currently selected actor proxies.*

### Protected Member Functions

- virtual void **buildDynamicControls** (dtDAL::PropertyContainer &propertyContainer, **DynamicGroupControl** \*parentControl=NULL)  
*Add all the dynamic controls for this proxy object.*
- virtual QString **GetGroupBoxLabelText** (const QString &baseGroupBoxName)
- **PropertyEditorModel** & **GetPropertyEditorModel** ()
- **DynamicGroupControl** \* **GetRootControl** ()

#### 4.2.1 Detailed Description

This class is the property editor for displaying and editing properties of selected objects. It shows the information about the selected actor(s) including type, name, location, rotation, etc... It is a dockable window.

## 4.2.2 Member Typedef Documentation

4.2.2.1 `typedef std::vector<dtCore::RefPtr<dtDAL::PropertyContainer> > PropertyContainerRefPtrVector`

## 4.2.3 Constructor & Destructor Documentation

4.2.3.1 `BasePropertyEditor (QMainWindow * parent)`

Constructor.

4.2.3.2 `~BasePropertyEditor () [virtual]`

Destructor.

## 4.2.4 Member Function Documentation

4.2.4.1 `void ActorPropertyChanged (dtDAL::PropertyContainer & propCon, dtDAL::ActorProperty & property) [slot]`

4.2.4.2 `void buildDynamicControls (dtDAL::PropertyContainer & propertyContainer, DynamicGroupControl * parentControl = NULL) [protected, virtual]`

Add all the dynamic controls for this proxy object.

4.2.4.3 `void CommitCurrentEdits () [inline]`

Convenience method for making the property editor commit its changes. There are some cases where an event can fire that needs to current state of the actor properties, but an edit is in process. Calling this will clear the focus of the property editor to make it commit the changes.

4.2.4.4 `DynamicControlFactory & GetDynamicControlFactory ()`

Returns the dynamic control factory. This is handy for registering new types.

4.2.4.5 `QString GetGroupBoxLabelText (const QString & baseGroupBoxName) [protected, virtual]`

4.2.4.6 `PropertyEditorModel & GetPropertyEditorModel () [protected]`

4.2.4.7 `DynamicGroupControl * GetRootControl () [protected]`

4.2.4.8 `void GetSelectedPropertyContainers (std::vector< dtDAL::PropertyContainer * > & toFill) [inline]`

Fills a vector with pointers to all the currently selected actor proxies.

4.2.4.9 `void HandlePropertyContainersSelected (PropertyContainerRefPtrVector & actors) [slot]`

Handles the actor selection changed event message from EditorEvents.

4.2.4.10 `virtual void PropertyAboutToChangeFromControl (dtDAL::PropertyContainer &, dtDAL::ActorProperty &, const std::string & oldValue, const std::string & newValue) [pure virtual, slot]`

4.2.4.11 `virtual void PropertyChangedFromControl (dtDAL::PropertyContainer &, dtDAL::ActorProperty &) [pure virtual, slot]`

4.2.4.12 `void ProxyNameChanged (dtDAL::ActorProxy & pc, std::string oldName) [slot]`

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

- `basepropertyeditor.h`
- `basepropertyeditor.cpp`

## 4.3 DeltaStepper Class Reference

```
#include <inc/dtQt/deltastepper.h>
```

### Public Slots

- void **Tick** ()

### Public Member Functions

- **DeltaStepper** ()
- virtual **~DeltaStepper** ()
- void **Start** ()
- void **Stop** ()

#### 4.3.1 Constructor & Destructor Documentation

4.3.1.1 **DeltaStepper** ()

4.3.1.2 **~DeltaStepper** () [virtual]

#### 4.3.2 Member Function Documentation

4.3.2.1 void **Start** ()

4.3.2.2 void **Stop** ()

4.3.2.3 void **Tick** () [slot]

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

- **deltastepper.h**
- **deltastepper.cpp**

## 4.4 DialogListSelection Class Reference

This is a generic dialog box for presenting a list of items for the user to choose from.

```
#include <inc/dtQt/dialoglistselection.h>
```

### Public Member Functions

- **DialogListSelection** (QWidget \*parent, const QString &windowTitle, const QString &groupName="")  
*Constructs a new selection dialog box.*
- virtual **~DialogListSelection** ()  
*Empty destructor.*
- const QString & **GetSelectedItem** () const  
*Gets the item currently selected in the dialog.*
- void **SetListItems** (const QStringList &list)  
*Sets the list of strings to be displayed in the list box.*

### Protected Slots

- void **onCurrentRowChanged** (int newRow)
- void **onItemClicked** (QListWidgetItem \*i)
- void **onItemDoubleClicked** (QListWidgetItem \*i)
- void **onSelectionChanged** ()

#### 4.4.1 Detailed Description

This is a generic dialog box for presenting a list of items for the user to choose from. The list is presented in a list box and contains ok and cancel buttons.

#### 4.4.2 Constructor & Destructor Documentation

##### 4.4.2.1 DialogListSelection (QWidget \* parent, const QString & windowTitle, const QString & groupName = "")

Constructs a new selection dialog box. Parameters

**parent** Parent widget to center this dialog over.

**windowTitle** Text to display in the dialog's title bar.

**groupName** An optional string. If specified, the list box will have a groupbox around it with this string as its title.

##### 4.4.2.2 ~DialogListSelection () [virtual]

Empty destructor. Returns

#### 4.4.3 Member Function Documentation

##### 4.4.3.1 const QString & GetSelectedItem () const

Gets the item currently selected in the dialog. Call this if the user presses the ok button to get the selected item. Returns The item selected from the list box.

4.4.3.2 void onCurrentRowChanged (int *newRow*) [protected, slot]

4.4.3.3 void onItemClicked (QListWidgetItem \* *i*) [protected, slot]

4.4.3.4 void onItemDoubleClicked (QListWidgetItem \* *i*) [protected, slot]

4.4.3.5 void onSelectionChanged () [protected, slot]

4.4.3.6 void SetListItems (const QStringList & *list*)

Sets the list of strings to be displayed in the list box. Parameters

***list*** The selection list.

Note The dialog will correctly handle an empty list, therefore, an empty list is valid, however, the dialog will not allow the user to do anything other than cancel or close.

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

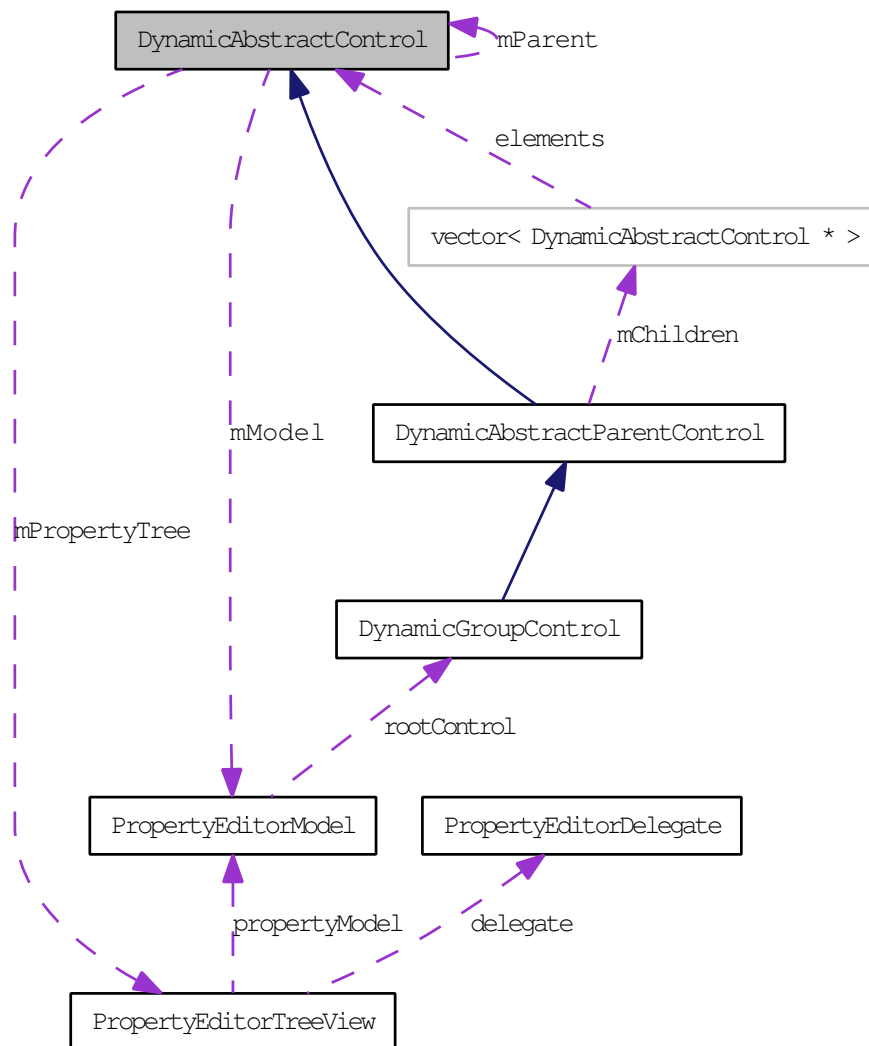
- **dialoglistselection.h**
- **dialoglistselection.cpp**

## 4.5 DynamicAbstractControl Class Reference

This is the base class for all the dynamic controls that are used in the property editor.

```
#include <inc/dtQt/dynamicabstractcontrol.h>
```

Inherited by **DynamicAbstractParentControl**, **DynamicBoolControl**, **DynamicColorElementControl**, **DynamicDoubleControl**, **DynamicEnumControl**, **DynamicFloatControl**, **DynamicIntControl**, **DynamicLabelControl**, **DynamicLongControl**, **DynamicStringControl**, and **DynamicVectorElementControl**. Collaboration diagram for DynamicAbstractControl:



### Public Slots

- virtual void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- virtual void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)  
*This method is called by one of the Sub Widgets from **DynamicSubWidgets.h** (p. 167) from the destructor of the passed in widget.*
- void **PropertyAboutToChangePassThrough** (dtDAL::PropertyContainer &, dtDAL::ActorProperty &prop, std::string oldValue, std::string newValue)  
*This is so child controls can, if desired can be connected to this which will emit the related signal.*

- void **PropertyChangedPassThrough** (dtDAL::PropertyContainer &, dtDAL::ActorProperty &prop)  
*This is so child controls can, if desired can be connected to this which will emit the related signal.*
- virtual bool **updateData** (QWidget \*widget)=0  
*Called when we should take the data out of the controls and put it into the actor.*

## Signals

- void **PropertyAboutToChange** (dtDAL::PropertyContainer &, dtDAL::ActorProperty &prop, std::string oldValue, std::string newValue)
- void **PropertyChanged** (dtDAL::PropertyContainer &, dtDAL::ActorProperty &prop)

## Public Member Functions

- **DynamicAbstractControl** ()  
*Constructor.*
- virtual ~**DynamicAbstractControl** ()  
*Destructor.*
- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)  
*Create the editor widget to be used when editing this control.*
- virtual **DynamicAbstractControl** \* **getChild** (int index)  
*This is the reverse of getChildIndex.*
- virtual int **getChildCount** ()  
*This tells you how many children you have.*
- virtual int **getChildIndex** (**DynamicAbstractControl** \*child)  
*This returns the index of the child so that the model can construct a QModelIndex.*
- virtual const QString **getDescription** ()  
*Returns a helpful description.*
- virtual const QString **getDisplayName** ()  
*Essentially returns a displayable string.*
- **DynamicControlFactory** \* **GetDynamicControlFactory** ()  
*If a control needs to create sub-controls, here's where it can get the factory.*
- **PropertyEditorModel** \* **GetModel** ()  
*Gets the editor model from the control.*
- **DynamicAbstractControl** \* **getParent** ()  
*Returns the parent of this control.*
- virtual const QString **getValueAsString** ()  
*Returns a string representation of the value.*
- virtual void **InitializeData** (**DynamicAbstractControl** \*newParent, **PropertyEditorModel** \*model, dtDAL::PropertyContainer \*pc, dtDAL::ActorProperty \*property)  
*Initialize the object.*

- virtual void **InstallEventFilterOnControl** (QObject \*filterObj)  
*Allows the abstract control to work with the delegate event filter.*
- virtual bool **isControlDoesCustomPainting** (int column)  
*Returns true if this control knows how to do custom painting for a specific column.*
- virtual bool **isEditable** ()  
*Returns true if this control is typically editable.*
- virtual bool **NeedsPersistentEditor** ()  
*Called when the abstract control is constructed to determine whether or not we need a persistent editor.*
- virtual void **NotifyParentOfPreUpdate** ()  
*This will notify the parent that the child is about to be updated.*
- virtual void **OnChildPreUpdate** (DynamicAbstractControl \*child)  
*This will be called by the child to notify their parent when they are being edited.*
- virtual void **paintColumn** (int column, QPainter \*painter, const QStyleOptionViewItem &opt)  
*This method allows you to do a custom paint on a specific column for a control.*
- void **SetBackgroundColor** (QWidget \*widget, QColor color)  
*A simple utility method to set the background color for all modes for a widget.*
- void **SetDynamicControlFactory** (DynamicControlFactory \*factory)  
*If a control needs to create sub-controls, here's where it can get the factory.*
- void **SetTreeView** (PropertyEditorTreeView \*newPropertyTree)  
*This was a temp hack used to set a parent widget for creating sub widgets.*
- virtual QSize **sizeHint** (int column, const QStyleOptionViewItem &opt)  
*This method works with **isControlDoesCustomPainting()** (p. 26) and **paintcolumn**.*
- virtual void **updateEditorFromModel** (QWidget \*widget)  
*This method is called by the delegate between the model and the view when data should be pushed to the editor from the model.*
- virtual bool **updateModelFromEditor** (QWidget \*widget)  
*This method is called by the delegate between the model and the view when data should be pushed into the model from the editor.*

### Protected Attributes

- bool **mInitialized**
- **PropertyEditorModel \* mModel**
- **DynamicAbstractControl \* mParent**
- dtCore::RefPtr< dtDAL::PropertyContainer > **mPropContainer**
- **PropertyEditorTreeView \* mPropertyTree**

#### 4.5.1 Detailed Description

This is the base class for all the dynamic controls that are used in the property editor.

#### 4.5.2 Constructor & Destructor Documentation

##### 4.5.2.1 DynamicAbstractControl ()

Constructor.

#### 4.5.2.2 ~DynamicAbstractControl () [virtual]

Destructor.

### 4.5.3 Member Function Documentation

#### 4.5.3.1 virtual void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [inline, virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented in [DynamicBoolControl](#) (p. 41), [DynamicColorElementControl](#) (p. 44), [DynamicColorRGBAControl](#) (p. 48), [DynamicDoubleControl](#) (p. 56), [DynamicEnumControl](#) (p. 59), [DynamicFloatControl](#) (p. 62), [DynamicIntControl](#) (p. 67), [DynamicLongControl](#) (p. 73), [DynamicResourceControlBase](#) (p. 79), [DynamicStringControl](#) (p. 82), and [DynamicVectorElementControl](#) (p. 88).

#### 4.5.3.2 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

Create the editor widget to be used when editing this control. It can be complex or simple. The default method does nothing.

Reimplemented in [DynamicArrayControl](#) (p. 34), [DynamicArrayElementControl](#) (p. 38), [DynamicBoolControl](#) (p. 41), [DynamicColorElementControl](#) (p. 45), [DynamicColorRGBAControl](#) (p. 49), [DynamicDoubleControl](#) (p. 56), [DynamicEnumControl](#) (p. 59), [DynamicFloatControl](#) (p. 62), [DynamicIntControl](#) (p. 67), [DynamicLongControl](#) (p. 73), [DynamicResourceControlBase](#) (p. 79), [DynamicStringControl](#) (p. 82), and [DynamicVectorElementControl](#) (p. 89).

#### 4.5.3.3 DynamicAbstractControl \* getChild (int *index*) [virtual]

This is the reverse of getChildIndex. It gets the child from the index. Note again that not all controls can have children. If you don't have children, then return NULL. Default is NULL

Reimplemented in [DynamicAbstractParentControl](#) (p. 31).

#### 4.5.3.4 int getChildCount () [virtual]

This tells you how many children you have. Note again that not all controls have children. Default is 0.

Reimplemented in [DynamicAbstractParentControl](#) (p. 31).

#### 4.5.3.5 int getChildIndex (DynamicAbstractControl \* *child*) [virtual]

This returns the index of the child so that the model can construct a QModelIndex. Note that not all dynamic controls can have children. However, there are several that can (such as Groups and complex controls like the Vec3). Therefore, we put the behavior on the base. If a control doesn't support children, just returns 0. Default is 0.

Reimplemented in [DynamicAbstractParentControl](#) (p. 31).

#### 4.5.3.6 const QString getDescription () [virtual]

Returns a helpful description. Usually a tooltip string. Each control needs to handle this, since some controls have properties and some don't. I.e., a group control doesn't have a real property under it, nor does the X and Y values of Vec3. Default is "".

Reimplemented in [DynamicArrayControl](#) (p. 34), [DynamicArrayElementControl](#) (p. 38), [DynamicBoolControl](#) (p. 41), [DynamicColorElementControl](#) (p. 45), [DynamicColorRGBAControl](#) (p. 49), [DynamicContainerControl](#) (p. 52), [DynamicDoubleControl](#) (p. 56), [DynamicEnumControl](#) (p. 59), [DynamicFloatControl](#) (p. 62), [DynamicIntControl](#) (p. 67), [DynamicLabelControl](#) (p. 70), [DynamicLongControl](#) (p. 73), [DynamicResourceControlBase](#) (p. 79), [DynamicStringControl](#) (p. 82), [DynamicVecNControl< PropertyType >](#) (p. 85), and [DynamicVectorElementControl](#) (p. 89).

**4.5.3.7 const QString getDisplayName () [virtual]**

Essentially returns a displayable string. Each control needs to handle this specially, since some controls have properties and some don't. I.e, a group control doesn't have a real property under it, nor does the X and Y values of Vec3. Default is "".

Reimplemented in **DynamicArrayControl** (p. 34), **DynamicArrayElementControl** (p. 38), **DynamicBoolControl** (p. 42), **DynamicColorElementControl** (p. 45), **DynamicColorRGBAControl** (p. 49), **DynamicContainerControl** (p. 52), **DynamicDoubleControl** (p. 56), **DynamicEnumControl** (p. 59), **DynamicFloatControl** (p. 62), **DynamicGroupControl** (p. 65), **DynamicIntControl** (p. 67), **DynamicLabelControl** (p. 70), **DynamicLongControl** (p. 73), **DynamicResourceControlBase** (p. 79), **DynamicStringControl** (p. 82), **DynamicVecNControl< PropertyType >** (p. 85), and **DynamicVectorElementControl** (p. 89).

**4.5.3.8 DynamicControlFactory \* GetDynamicControlFactory ()**

If a control needs to create sub-controls, here's where it can get the factory.

**4.5.3.9 PropertyEditorModel\* GetModel () [inline]**

Gets the editor model from the control. It should be rare that you ever need this.

**4.5.3.10 DynamicAbstractControl \* getParent ()**

Returns the parent of this control. NULL if there is no parent. All controls have a parent except for those at the root.

**4.5.3.11 const QString getValueAsString () [virtual]**

Returns a string representation of the value. This is typically only useful for controls that are not editable. Default is "".

Reimplemented in **DynamicArrayControl** (p. 34), **DynamicArrayElementControl** (p. 38), **DynamicBoolControl** (p. 42), **DynamicColorElementControl** (p. 45), **DynamicColorRGBAControl** (p. 49), **DynamicContainerControl** (p. 52), **DynamicDoubleControl** (p. 56), **DynamicEnumControl** (p. 60), **DynamicFloatControl** (p. 62), **DynamicIntControl** (p. 67), **DynamicLabelControl** (p. 70), **DynamicLongControl** (p. 73), **DynamicResourceControlBase** (p. 79), **DynamicStringControl** (p. 82), **DynamicVecNControl< PropertyType >** (p. 85), and **DynamicVectorElementControl** (p. 89).

**4.5.3.12 void handleSubEditDestroy (QWidget \* widget, QAbstractItemDelegate::EndEditHint hint = QAbstractItemDelegate::NoHint) [virtual, slot]**

This method is called by one of the Sub Widgets from **DynamicSubWidgets.h** (p. 167) from the destructor of the passed in widget. The intent is to work around the way editors work in QT trees. Since the editors are created only when the user should be editing the field and then destroyed as soon as they lose focus, it is important that any Dynamic controls not be holding on to bogus pointers. This method gives you a chance to set a pointer back to NULL or whatever.

Reimplemented in **DynamicArrayControl** (p. 34), **DynamicArrayElementControl** (p. 38), **DynamicBoolControl** (p. 42), **DynamicColorElementControl** (p. 45), **DynamicColorRGBAControl** (p. 49), **DynamicDoubleControl** (p. 57), **DynamicEnumControl** (p. 60), **DynamicFloatControl** (p. 63), **DynamicIntControl** (p. 68), **DynamicLongControl** (p. 74), **DynamicResourceControlBase** (p. 79), **DynamicStringControl** (p. 83), and **DynamicVectorElementControl** (p. 89).

**4.5.3.13 void InitializeData (DynamicAbstractControl \* newParent, PropertyEditorModel \* model, dtDAL::PropertyContainer \* pc, dtDAL::ActorProperty \* property) [virtual]**

Initialize the object. We don't do this in the constructor because it's created via the ObjectFactory which primarily works with parameterless constructors at the moment. Note - The base class does not store the property. Each subclass is responsible for doing that.

Reimplemented in **DynamicArrayControl** (p. 34), **DynamicArrayElementControl** (p. 38), **DynamicBoolControl** (p. 42), **DynamicColorElementControl** (p. 45), **DynamicColorRGBAControl** (p. 49), **DynamicContainerControl** (p. 52), **DynamicDoubleControl** (p. 57), **DynamicEnumControl** (p. 60), **DynamicFloatControl** (p. 63), **DynamicIntControl** (p. 68), **DynamicLabelControl** (p. 70), **DynamicLongControl** (p. 74), **DynamicResourceControlBase** (p. 79), **DynamicStringControl** (p. 83), **DynamicVecNControl< PropertyType >** (p. 85), and **DynamicVectorElementControl** (p. 89).

**4.5.3.14 void InstallEventFilterOnControl (QObject \* *filterObj*) [virtual]**

Allows the abstract control to work with the delegate event filter. If you have any nested controls that are created as part of the editor, you need to use one of the sub controls and make sure you handle NULL'ing your pointer and then override this method to install an event filter on your non-null editor objects. This is called by the **PropertyEditorDelegate** (p. 100) after the editor is created. The default calls `QObject::installEventFilter()`

**4.5.3.15 bool isControlDoesCustomPainting (int *column*) [virtual]**

Returns true if this control knows how to do custom painting for a specific column. This is necessary for some controls that have non string elements like a button or a color swatch. If this method returns true, you **MUST** override the paint and sizeHint methods or you won't see anything for your row. This method is based on the column, so you will only get paint and sizehint calls for a column that returns true. Default is false. Note Be careful using this on editable columns as you may get yourself in trouble with nested and competing paints. Returns True if your control does custom paint and **sizeHint()** (p. 27) for this column.

**4.5.3.16 bool isEditable () [virtual]**

Returns true if this control is typically editable. For instance, a float control is editable, but a group isn't. Note that the subelements of a vector are editor whereas the vector control itself isn't editable. If you return true, then you need to be able to delegate a control to edit the data. Default is false.

Reimplemented in **DynamicArrayControl** (p. 35), **DynamicArrayElementControl** (p. 38), **DynamicBoolControl** (p. 42), **DynamicColorElementControl** (p. 45), **DynamicColorRGBAControl** (p. 49), **DynamicContainerControl** (p. 52), **DynamicDoubleControl** (p. 57), **DynamicEnumControl** (p. 60), **DynamicFloatControl** (p. 63), **DynamicIntControl** (p. 68), **DynamicLongControl** (p. 74), **DynamicResourceControlBase** (p. 79), **DynamicStringControl** (p. 83), **DynamicVecNControl < PropertyType >** (p. 85), and **DynamicVectorElementControl** (p. 89).

**4.5.3.17 bool NeedsPersistentEditor () [virtual]**

Called when the abstract control is constructed to determine whether or not we need a persistent editor. Persistent editors are created after the control is created and closed when the control is removed from the tree. This is used for some controls that need to ALWAYS show a button or edit box in order to prevent user confusion. Default is false;

Note This doesn't work. Well, the method works, but there is currently no way to make an editor persistent as I'd like. By persistent, I mean, actually make the control show up permanently, so that the user doesn't have to click anything to make it show up.

Reimplemented in **DynamicColorRGBAControl** (p. 49).

**4.5.3.18 virtual void NotifyParentOfPreUpdate () [inline, virtual]**

This will notify the parent that the child is about to be updated.

**4.5.3.19 void OnChildPreUpdate (DynamicAbstractControl \* *child*) [virtual]**

This will be called by the child to notify their parent when they are being edited. This function should be overwritten if the parent needs to know when any of its children are about to be edited.

Reimplemented in **DynamicArrayElementControl** (p. 38).

**4.5.3.20 void paintColumn (int *column*, QPainter \* *painter*, const QStyleOptionViewItem & *opt*) [virtual]**

This method allows you to do a custom paint on a specific column for a control. It works with `isControlDoesCustomPainting` and **sizeHint()** (p. 27). For instance, the color control might put up a specific picker button or maybe a little color swatch of the selected color. The only way to do that would be in here.

**4.5.3.21 void PropertyAboutToChange (dtDAL::PropertyContainer &, dtDAL::ActorProperty & *prop*, std::string *oldValue*, std::string *newValue*) [signal]****4.5.3.22 void PropertyAboutToChangePassThrough (dtDAL::PropertyContainer & *proxy*, dtDAL::ActorProperty & *prop*, std::string *oldValue*, std::string *newValue*) [slot]**

This is so child controls can, if desired can be connected to this which will emit the related signal.

**4.5.3.23 void PropertyChanged (dtDAL::PropertyContainer &, dtDAL::ActorProperty & *prop*) [signal]**

**4.5.3.24 void PropertyChangedPassThrough (dtDAL::PropertyContainer & *proxy*, dtDAL::ActorProperty & *prop*) [slot]**

This is so child controls can, if desired can be connected to this which will emit the related signal.

**4.5.3.25 void SetBackgroundColor (QWidget \* *widget*, QColor *color*)**

A simple utility method to set the background color for all modes for a widget. This is typically used when we have wrapper widgets with children in the edit portion of a dynamic control to make the control blend in better...

**4.5.3.26 void SetDynamicControlFactory (DynamicControlFactory \* *factory*)**

If a control needs to create sub-controls, here's where it can get the factory.

**4.5.3.27 void SetTreeView (PropertyEditorTreeView \* *newPropertyTree*)**

This was a temp hack used to set a parent widget for creating sub widgets. This shouldn't be needed and can be deleted once the final tree is settled on

**4.5.3.28 QSize sizeHint (int *column*, const QStyleOptionViewItem & *opt*) [virtual]**

This method works with `isControlDoesCustomPainting()` (p. 26) and `paintcolumn`. You should return the size hint for your column. Default does nothing. Returns the QSize sizehint for this column in the tree.

**4.5.3.29 virtual bool updateData (QWidget \* *widget*) [pure virtual, slot]**

Called when we should take the data out of the controls and put it into the actor. This is typically trapped to a lost focus or return pressed or similar user event behavior. This can also be called by the parent control at the moment we change selection. Returns Returns true if any data was actually changed and successfully set on the control - This is purely virtual

Implemented in `DynamicArrayControl` (p. 35), `DynamicArrayElementControl` (p. 39), `DynamicBoolControl` (p. 42), `DynamicColorElementControl` (p. 45), `DynamicColorRGBAControl` (p. 49), `DynamicContainerControl` (p. 53), `DynamicDoubleControl` (p. 57), `DynamicEnumControl` (p. 60), `DynamicFloatControl` (p. 63), `DynamicGroupControl` (p. 65), `DynamicIntControl` (p. 68), `DynamicLabelControl` (p. 70), `DynamicLongControl` (p. 74), `DynamicResourceControlBase` (p. 80), `DynamicStringControl` (p. 83), `DynamicNoUpdateParentControl` (p. 76), and `DynamicVectorElementControl` (p. 89).

**4.5.3.30 void updateEditorFromModel (QWidget \* *widget*) [virtual]**

This method is called by the delegate between the model and the view when data should be pushed to the editor from the model. I'm not exactly sure when that happens, but I assume that's at construction time of the editor. The default does nothing.

- In all likelihood, this method won't need the widget that's passed in, since it's tracked by the individual subclass.

Reimplemented in `DynamicBoolControl` (p. 42), `DynamicColorElementControl` (p. 45), `DynamicColorRGBAControl` (p. 49), `DynamicDoubleControl` (p. 57), `DynamicEnumControl` (p. 60), `DynamicFloatControl` (p. 63), `DynamicIntControl` (p. 68), `DynamicLongControl` (p. 74), `DynamicResourceControlBase` (p. 80), `DynamicStringControl` (p. 83), and `DynamicVectorElementControl` (p. 89).

**4.5.3.31 bool updateModelFromEditor (QWidget \* *widget*) [virtual]**

This method is called by the delegate between the model and the view when data should be pushed into the model from the editor. This is called when the control thinks it's done editing. Perhaps the user pressed a key, or tabbed away, or something like that. The default does nothing and returns false.

Returns Returns true if the model was successfully changed. ie, the data was different and there were no errors. It would be nice if the caller of this method would generate the event message. However, since it can be called from a variety of places, we have to generate the property changed event from inside here. So, don't forget :)

Reimplemented in `DynamicBoolControl` (p. 42), `DynamicColorElementControl` (p. 46), `DynamicColorRGBAControl` (p. 50), `DynamicDoubleControl` (p. 57), `DynamicEnumControl` (p. 60), `DynamicFloatControl` (p. 63), `DynamicIntControl` (p. 68), `DynamicLongControl` (p. 74), `DynamicResourceControlBase` (p. 80), `DynamicStringControl` (p. 83), and `DynamicVectorElementControl` (p. 90).

#### 4.5.4 Member Data Documentation

4.5.4.1 `bool mInitialized` [protected]

4.5.4.2 `PropertyEditorModel* mModel` [protected]

4.5.4.3 `DynamicAbstractControl* mParent` [protected]

4.5.4.4 `dtCore::RefPtr<dtDAL::PropertyContainer> mPropContainer` [protected]

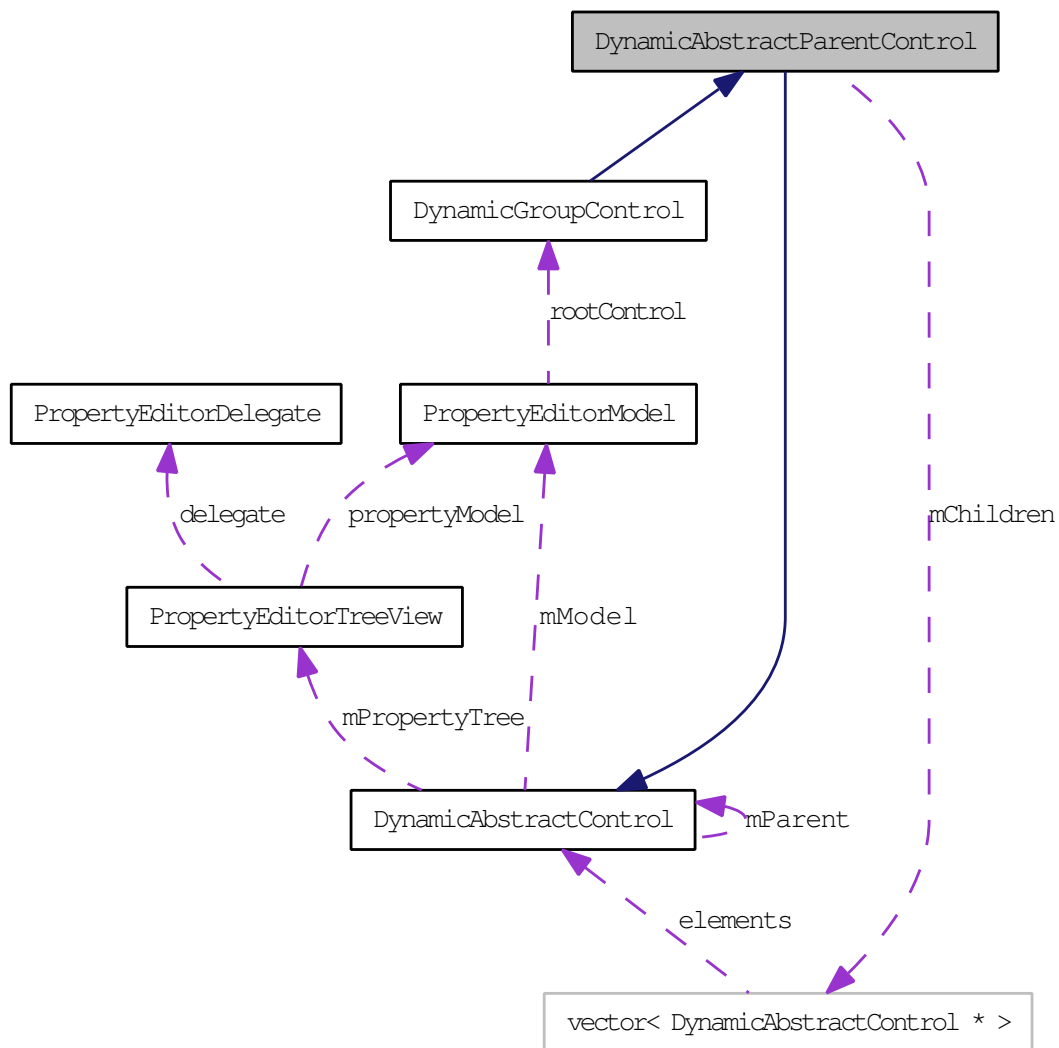
4.5.4.5 `PropertyEditorTreeView* mPropertyTree` [protected]

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

- `dynamicabstractcontrol.h`
- `dynamicabstractcontrol.cpp`



Collaboration diagram for DynamicAbstractParentControl:



## Public Member Functions

- **DynamicAbstractParentControl** ()  
*Constructor.*
- virtual **~DynamicAbstractParentControl** ()  
*Destructor.*
- virtual **DynamicAbstractControl \* getChild** (int index)
- virtual int **getChildCount** ()
- virtual int **getChildIndex** (**DynamicAbstractControl \*child**)
- void **removeAllChildren** (**PropertyEditorModel \*model**)  
*A clean up method you should use when you are planning to reuse this control.*

## Protected Attributes

- std::vector< **DynamicAbstractControl \* >** **mChildren**

### 4.6.1 Detailed Description

This is a base class for any dynamic control that has children. It keeps a vector of **DynamicAbstractControl** (p. 21) \*'s and knows how to work with it. This is common behavior in the dynamic controls that was pulled to this class.

## 4.6.2 Constructor & Destructor Documentation

### 4.6.2.1 DynamicAbstractParentControl ()

Constructor.

### 4.6.2.2 ~DynamicAbstractParentControl () [virtual]

Destructor.

## 4.6.3 Member Function Documentation

### 4.6.3.1 DynamicAbstractControl \* getChild (int *index*) [virtual]

See also `DynamicAbstractControl::getChild` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

### 4.6.3.2 int getChildCount () [virtual]

See also `DynamicAbstractControl::getChildCount` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

### 4.6.3.3 int getChildIndex (DynamicAbstractControl \* *child*) [virtual]

See also `DynamicAbstractControl::getChildIndex` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

### 4.6.3.4 void removeAllChildren (PropertyEditorModel \* *model*)

A clean up method you should use when you are planning to reuse this control.

## 4.6.4 Member Data Documentation

### 4.6.4.1 std::vector<DynamicAbstractControl\*> mChildren [protected]

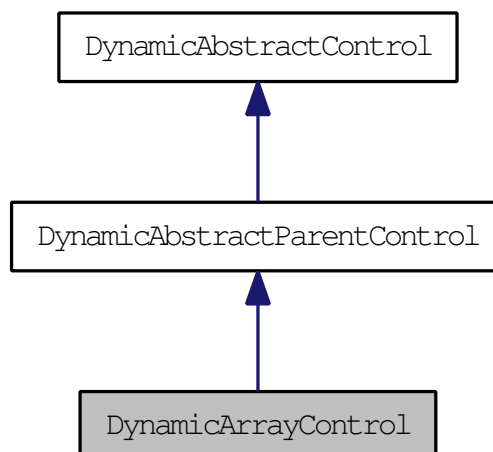
The documentation for this class was generated from the following files:

- `dynamicabstractparentcontrol.h`
- `dynamicabstractparentcontrol.cpp`

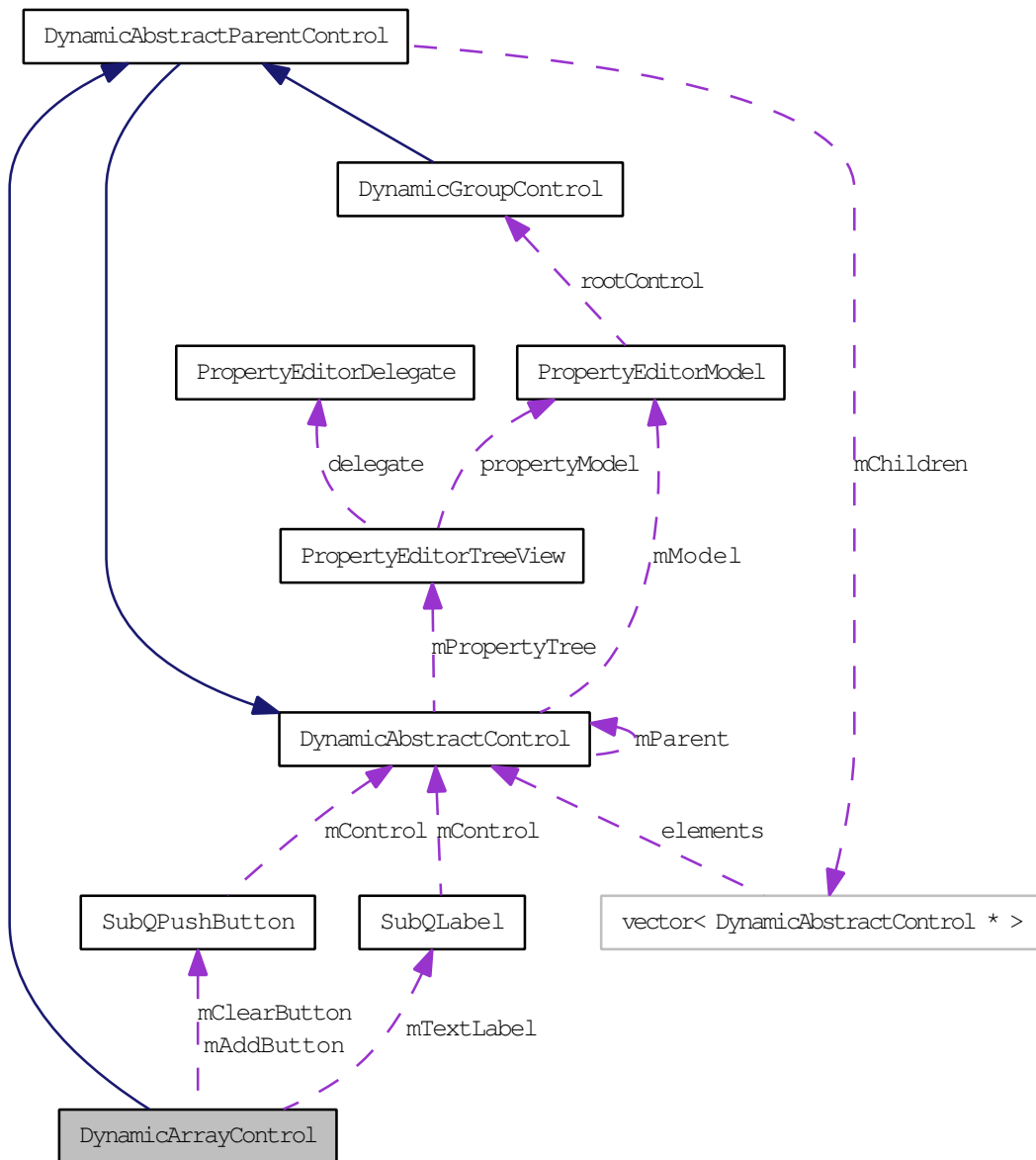
## 4.7 DynamicArrayControl Class Reference

This is the dynamic control for the float data type - used in the property editor.

`#include <inc/dtQt/dynamicarraycontrol.h>`Inheritance diagram for DynamicArrayControl:



Collaboration diagram for DynamicArrayControl:



## Public Slots

- void **onAddClicked** ()  
*Signal when the Add button has been clicked.*
- void **onClearClicked** ()  
*Signal when the Clear button has been clicked.*
- virtual bool **updateData** (QWidget \*widget)  
*Signal when the data is updated.*

## Public Member Functions

- **DynamicArrayControl** ()  
*Constructor.*
- **~DynamicArrayControl** ()

*Destructor.*

- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- virtual const QString **getValueAsString** ()
- virtual void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual void **initializeData** (DynamicAbstractControl \*newParent, PropertyEditorModel \*model, dtDAL::PropertyContainer \*newPC, dtDAL::ActorProperty \*property)
- virtual bool **isEditable** ()
- void **resizeChildren** (bool forceRefresh=false, bool isChild=false, bool initializing=false)

*Resize the number of children in the array based on the actual property.*

### 4.7.1 Detailed Description

This is the dynamic control for the float data type - used in the property editor.

### 4.7.2 Constructor & Destructor Documentation

#### 4.7.2.1 DynamicArrayControl ()

Constructor.

#### 4.7.2.2 ~DynamicArrayControl ()

Destructor.

### 4.7.3 Member Function Documentation

#### 4.7.3.1 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

See also **DynamicAbstractControl::createEditor** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.7.3.2 const QString getDescription () [virtual]

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.7.3.3 const QString getDisplayName () [virtual]

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.7.3.4 const QString getValueAsString () [virtual]

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.7.3.5 void handleSubEditDestroy (QWidget \* *widget*, QAbstractItemDelegate::EndEditHint *hint* = QAbstractItemDelegate::NoHint) [virtual]

See also **DynamicAbstractControl::handleSubEditDestroy** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.7.3.6 void initializeData (DynamicAbstractControl \* *newParent*, PropertyEditorModel \* *model*, dtDAL::PropertyContainer \* *newPC*, dtDAL::ActorProperty \* *property*) [virtual]

See also **DynamicAbstractControl::initializeData** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.7.3.7 bool isEditable () [virtual]**

See also **DynamicAbstractControl::isEditable** (p. 26)

Reimplemented from **DynamicAbstractControl** (p. 26).

**4.7.3.8 void onAddClicked () [slot]**

Signal when the Add button has been clicked.

**4.7.3.9 void onClearClicked () [slot]**

Signal when the Clear button has been clicked.

**4.7.3.10 void resizeChildren (bool *forceRefresh* = false, bool *isChild* = false, bool *initializing* = false)**

Resize the number of children in the array based on the actual property. Parameters

← ***forceRefresh*** True to force the UI to refresh.

← ***isChild*** True if a child control is calling this from its parent.

← ***initializing*** True if we are initializing the list.

**4.7.3.11 bool updateData (QWidget \* *widget*) [virtual, slot]**

Signal when the data is updated.

Implements **DynamicAbstractControl** (p. 27).

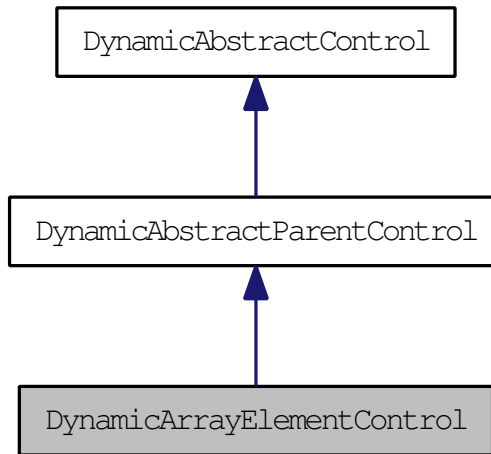
The documentation for this class was generated from the following files:

- **dynamicarraycontrol.h**
- **dynamicarraycontrol.cpp**

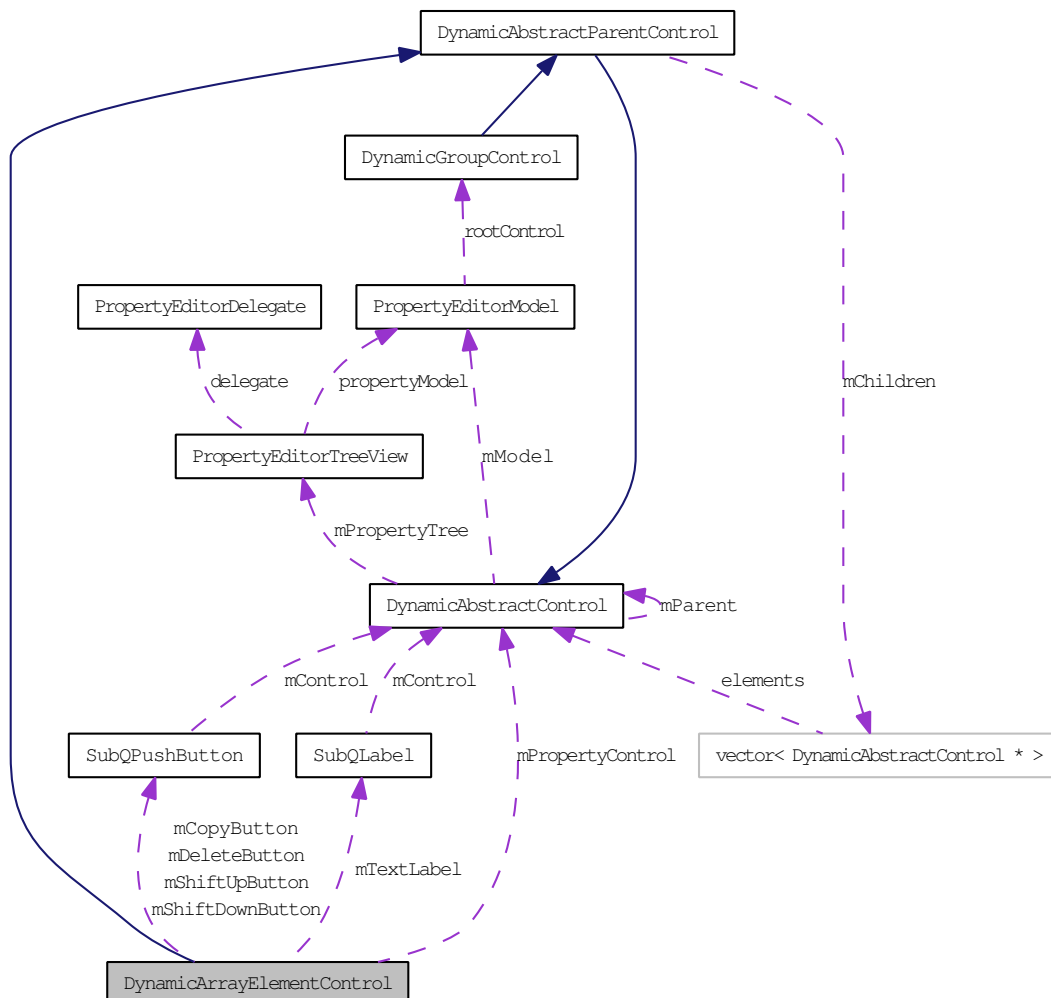
### 4.8 DynamicArrayElementControl Class Reference

This is the dynamic control for the float data type - used in the property editor.

#include <inc/dtQt/dynamicarrayelementcontrol.h> Inheritance diagram for DynamicArrayElementControl:



Collaboration diagram for DynamicArrayElementControl:



## Public Slots

- void **onCopyClicked** ()  
*Signal when the Copy button has been clicked.*
- void **onDeleteClicked** ()  
*Signal when the Delete button has been clicked.*
- void **onShiftDownClicked** ()  
*Signal when the Shift Down button has been clicked.*
- void **onShiftUpClicked** ()  
*Signal when the Shift Up button has been clicked.*
- virtual bool **updateData** (QWidget \*widget)  
*Signal when the data is updated.*

## Public Member Functions

- **DynamicArrayElementControl** (int index)  
*Constructor.*
- **~DynamicArrayElementControl** ()  
*Destructor.*
- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- virtual int **GetIndex** ()  
*Gets the current index of the element.*
- virtual const QString **getValueAsString** ()
- virtual void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual void **InitializeData** (DynamicAbstractControl \*newParent, PropertyEditorModel \*model, dtDAL::PropertyContainer \*newPC, dtDAL::ActorProperty \*property)
- virtual bool **isEditable** ()
- virtual void **OnChildPreUpdate** (DynamicAbstractControl \*child)
- virtual void **SetActive** ()  
*Sets this as the currently active index.*
- virtual void **SetIndex** (int index)  
*Sets the current index of the element.*

### 4.8.1 Detailed Description

This is the dynamic control for the float data type - used in the property editor.

### 4.8.2 Constructor & Destructor Documentation

#### 4.8.2.1 DynamicArrayElementControl (int *index*)

Constructor.

#### 4.8.2.2 ~DynamicArrayElementControl ()

Destructor.

### 4.8.3 Member Function Documentation

#### 4.8.3.1 **QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]**

See also **DynamicAbstractControl::createEditor** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.8.3.2 **const QString getDescription () [virtual]**

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.8.3.3 **const QString getDisplayName () [virtual]**

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.8.3.4 **int GetIndex () [virtual]**

Gets the current index of the element.

#### 4.8.3.5 **const QString getValueAsString () [virtual]**

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.8.3.6 **void handleSubEditDestroy (QWidget \* *widget*, QAbstractItemDelegate::EndEditHint *hint* = QAbstractItemDelegate::NoHint) [virtual]**

See also **DynamicAbstractControl::handleSubEditDestroy** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.8.3.7 **void InitializeData (DynamicAbstractControl \* *newParent*, PropertyEditorModel \* *model*, dtDAL::PropertyContainer \* *newPC*, dtDAL::ActorProperty \* *property*) [virtual]**

See also **DynamicAbstractControl::InitializeData** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.8.3.8 **bool isEditable () [virtual]**

See also **DynamicAbstractControl::isEditable** (p. 26)

Reimplemented from **DynamicAbstractControl** (p. 26).

#### 4.8.3.9 **void OnChildPreUpdate (DynamicAbstractControl \* *child*) [virtual]**

See also **DynamicAbstractControl::OnChildPreUpdate** (p. 26)

Reimplemented from **DynamicAbstractControl** (p. 26).

#### 4.8.3.10 **void onCopyClicked () [slot]**

Signal when the Copy button has been clicked.

#### 4.8.3.11 **void onDeleteClicked () [slot]**

Signal when the Delete button has been clicked.

#### 4.8.3.12 **void onShiftDownClicked () [slot]**

Signal when the Shift Down button has been clicked.

#### 4.8.3.13 **void onShiftUpClicked () [slot]**

Signal when the Shift Up button has been clicked.

#### 4.8.3.14 **void SetActive () [virtual]**

Sets this as the currently active index.

**4.8.3.15 void SetIndex (int *index*) [virtual]**

Sets the current index of the element.

**4.8.3.16 bool updateData (QWidget \* *widget*) [virtual, slot]**

Signal when the data is updated.

Implements **DynamicAbstractControl** (p. 27).

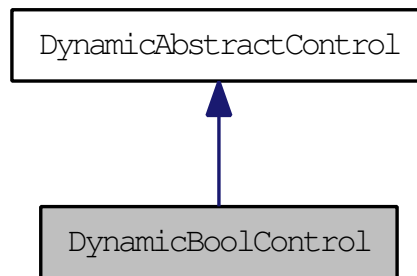
The documentation for this class was generated from the following files:

- **dynamicarrayelementcontrol.h**
- **dynamicarrayelementcontrol.cpp**

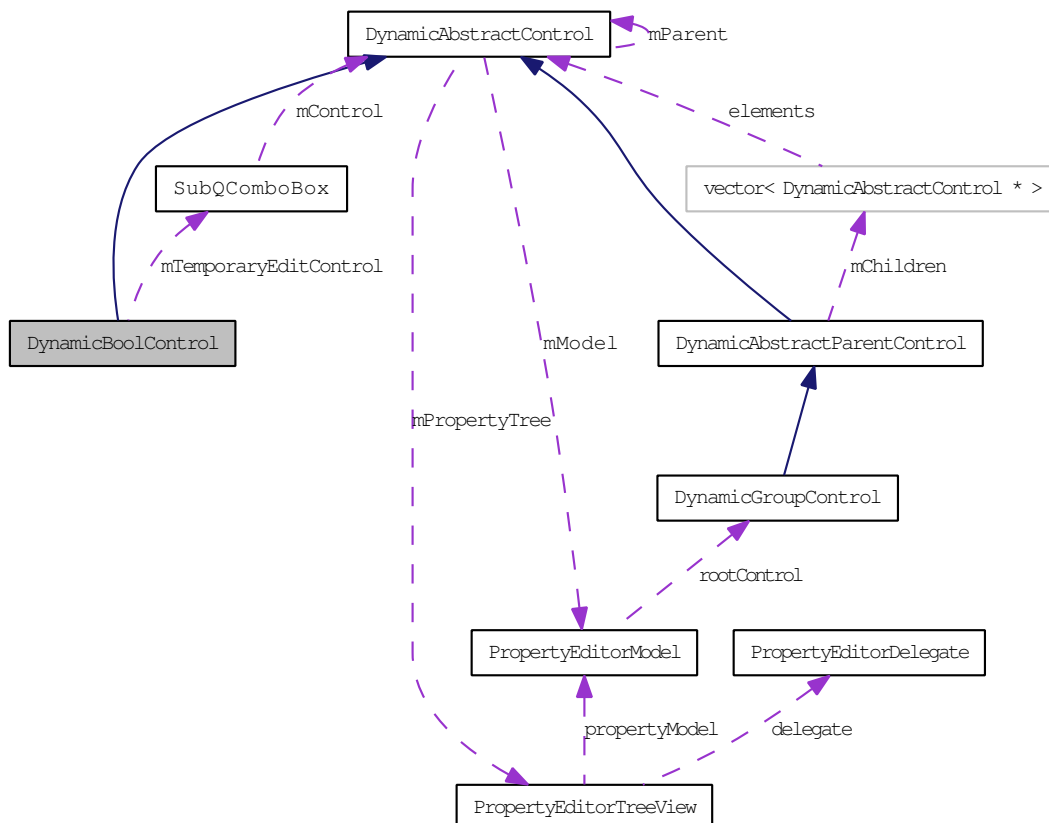
## 4.9 DynamicBoolControl Class Reference

This is the dynamic control for the bool data type - used in the property editor.

#include <inc/dtQt/dynamicboolcontrol.h> Inheritance diagram for DynamicBoolControl:



Collaboration diagram for DynamicBoolControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- void **itemSelected** (int index)  
*Called when the user selects an item in the combo box.*
- virtual bool **updateData** (QWidget \*widget)

## Public Member Functions

- **DynamicBoolControl** ()  
*Constructor.*
- virtual **~DynamicBoolControl** ()  
*Destructor.*
- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- virtual const QString **getValueAsString** ()
- virtual void **initializeData** (DynamicAbstractControl \*newParent, PropertyEditorModel \*model, dtDAL::PropertyContainer \*newPC, dtDAL::ActorProperty \*property)
- virtual bool **isEditable** ()
- virtual void **updateEditorFromModel** (QWidget \*widget)
- virtual bool **updateModelFromEditor** (QWidget \*widget)

## Static Public Attributes

- static const QString **FALSE\_LABEL**
- static const QString **TRUE\_LABEL**

### 4.9.1 Detailed Description

This is the dynamic control for the bool data type - used in the property editor.

### 4.9.2 Constructor & Destructor Documentation

#### 4.9.2.1 DynamicBoolControl ()

Constructor.

#### 4.9.2.2 ~DynamicBoolControl () [virtual]

Destructor.

### 4.9.3 Member Function Documentation

#### 4.9.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.9.3.2 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

See also **DynamicAbstractControl::createEditor** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.9.3.3 const QString getDescription () [virtual]

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.9.3.4 const QString getDisplayName () [virtual]

See also [DynamicAbstractControl::getDisplayName](#) (p. 25)

Reimplemented from [DynamicAbstractControl](#) (p. 25).

#### 4.9.3.5 const QString getValueAsString () [virtual]

See also [DynamicAbstractControl::getValueAsString](#) (p. 25)

Reimplemented from [DynamicAbstractControl](#) (p. 25).

#### 4.9.3.6 void handleSubEditDestroy (QWidget \* *widget*, QAbstractItemDelegate::EndEditHint *hint* = QAbstractItemDelegate::NoHint) [inline, virtual, slot]

See also [DynamicAbstractControl::handleSubEditDestroy](#) (p. 25)

Reimplemented from [DynamicAbstractControl](#) (p. 25).

#### 4.9.3.7 void InitializeData (DynamicAbstractControl \* *newParent*, PropertyEditorModel \* *model*, dtDAL::PropertyContainer \* *newPC*, dtDAL::ActorProperty \* *property*) [virtual]

See also [DynamicAbstractControl::InitializeData](#) (p. 25)

Reimplemented from [DynamicAbstractControl](#) (p. 25).

#### 4.9.3.8 bool isEditable () [virtual]

See also [DynamicAbstractControl::isEditable](#) (p. 26)

Reimplemented from [DynamicAbstractControl](#) (p. 26).

#### 4.9.3.9 void itemSelected (int *index*) [slot]

Called when the user selects an item in the combo box.

#### 4.9.3.10 bool updateData (QWidget \* *widget*) [virtual, slot]

See also [DynamicAbstractControl::updateData](#) (p. 27)

Implements [DynamicAbstractControl](#) (p. 27).

#### 4.9.3.11 void updateEditorFromModel (QWidget \* *widget*) [virtual]

See also [DynamicAbstractControl::updateEditorFromModel](#) (p. 27)

Reimplemented from [DynamicAbstractControl](#) (p. 27).

#### 4.9.3.12 bool updateModelFromEditor (QWidget \* *widget*) [virtual]

See also [DynamicAbstractControl::updateModelFromEditor](#) (p. 27)

Reimplemented from [DynamicAbstractControl](#) (p. 27).

### 4.9.4 Member Data Documentation

#### 4.9.4.1 const QString FALSE\_LABEL [static]

#### 4.9.4.2 const QString TRUE\_LABEL [static]

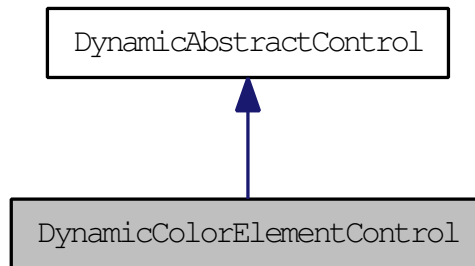
The documentation for this class was generated from the following files:

- [dynamicboolcontrol.h](#)
- [dynamicboolcontrol.cpp](#)

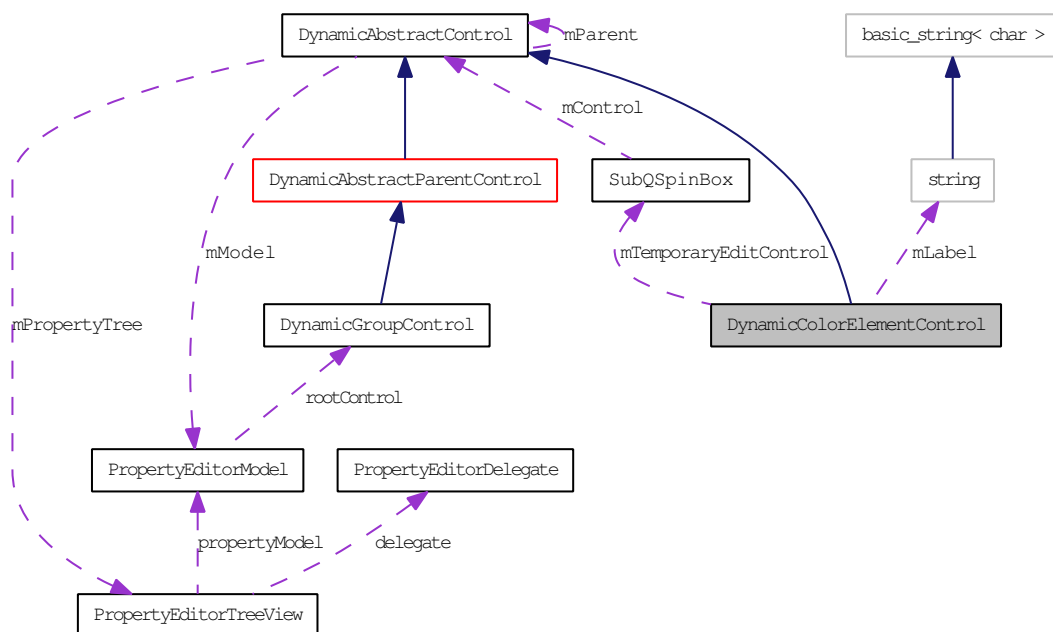
## 4.10 DynamicColorElementControl Class Reference

This is the a sub control used by the various color property classes.

#include <inc/dtQt/dynamiccolorelementcontrol.h> Inheritance diagram for DynamicColorElementControl:



Collaboration diagram for DynamicColorElementControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual bool **updateData** (QWidget \*widget)  
*Called when we should take the data out of the controls and put it into the actor.*

### Public Member Functions

- **DynamicColorElementControl** (dtDAL::ColorRgbaActorProperty \*colorRGBA, int whichIndex, const std::string &newLabel)  
*Constructor - For the ColorRgbaActorProperty property.*

- virtual `~DynamicColorElementControl ()`  
*Constructor - For the ColorRgbaActorProperty property.*
- virtual `QWidget * createEditor (QWidget *parent, const QStyleOptionViewItem &option, const QModelIndex &index)`
- virtual const `QString getDescription ()`
- virtual const `QString getDisplayName ()`
- int `getValue ()`  
*A convenience method to get the value from the associated vector.*
- virtual const `QString getValueAsString ()`
- virtual void `initializeData (DynamicAbstractControl *newParent, PropertyEditorModel *model, dtDAL::PropertyContainer *pc, dtDAL::ActorProperty *property)`
- virtual bool `isEditable ()`
- void `setValue (int value)`  
*Puts the passed in value into the appropriate vector at whichElement index.*
- virtual void `updateEditorFromModel (QWidget *widget)`
- virtual bool `updateModelFromEditor (QWidget *widget)`

### Static Public Member Functions

- static int `convertColorFloatToInt (float value)`  
*Convert the 0.0 to 1.0 float color value to the 0 to 255 int display range.*
- static float `convertColorIntToFloat (int value)`  
*Convert the 0 to 255 int display range to 0.0 to a 1.0 float value.*

#### 4.10.1 Detailed Description

This is the a sub control used by the various color property classes. In order to draw a vector 3 in the property tree, you actually have an X, Y, and Z entry. This class is for that. And, this class actually supports the ColorRgbaActorProperty, and ColorRgbActorProperty with separate constructors. It provides a get and set method to get at the data.

#### 4.10.2 Constructor & Destructor Documentation

##### 4.10.2.1 DynamicColorElementControl (dtDAL::ColorRgbaActorProperty \* *colorRGBA*, int *whichIndex*, const std::string & *newLabel*)

Constructor - For the ColorRgbaActorProperty property. - We can put data in the constructor because aren't using the factory for this.

##### 4.10.2.2 ~DynamicColorElementControl () [virtual]

Constructor - For the ColorRgbaActorProperty property. - We can put data in the constructor because aren't using the factory for this. Destructor

#### 4.10.3 Member Function Documentation

##### 4.10.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from `DynamicAbstractControl` (p. 24).

**4.10.3.2 int convertColorFloatToInt (float *value*) [static]**

Convert the 0.0 to 1.0 float color value to the 0 to 255 int display range.

**4.10.3.3 float convertColorIntToFloat (int *value*) [static]**

Convert the 0 to 255 int display range to 0.0 to a 1.0 float value.

**4.10.3.4 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]**

See also **DynamicAbstractControl::createEditor** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

**4.10.3.5 const QString getDescription () [virtual]**

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

**4.10.3.6 const QString getDisplayName () [virtual]**

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.10.3.7 int getValue ()**

A convenience method to get the value from the associated vector. Which element is the 0 based index into the vector.

**4.10.3.8 const QString getValueAsString () [virtual]**

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.10.3.9 void handleSubEditDestroy (QWidget \* *widget*, QAbstractItemDelegate::EndEditHint *hint* = QAbstractItemDelegate::NoHint) [inline, virtual, slot]**

See also **DynamicAbstractControl::handleSubEditDestroy** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.10.3.10 void InitializeData (DynamicAbstractControl \* *newParent*, PropertyEditorModel \* *model*, dtDAL::PropertyContainer \* *pc*, dtDAL::ActorProperty \* *property*) [virtual]**

See also **DynamicAbstractControl::InitializeData** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.10.3.11 bool isEditable () [virtual]**

See also **DynamicAbstractControl::isEditable** (p. 26)

Reimplemented from **DynamicAbstractControl** (p. 26).

**4.10.3.12 void setValue (int *value*)**

Puts the passed in value into the appropriate vector at whichElement index.

**4.10.3.13 bool updateData (QWidget \* *widget*) [virtual, slot]**

Called when we should take the data out of the controls and put it into the actor. This is typically trapped to a lost focus or return pressed or similar user event behavior. This can also be called by the parent control at the moment we change selection. Returns Returns true if any data was actually changed and successfully set on the control - This is purely virtual

Implements **DynamicAbstractControl** (p. 27).

**4.10.3.14 void updateEditorFromModel (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateEditorFromModel** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

**4.10.3.15 bool updateModelFromEditor (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateModelFromEditor** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

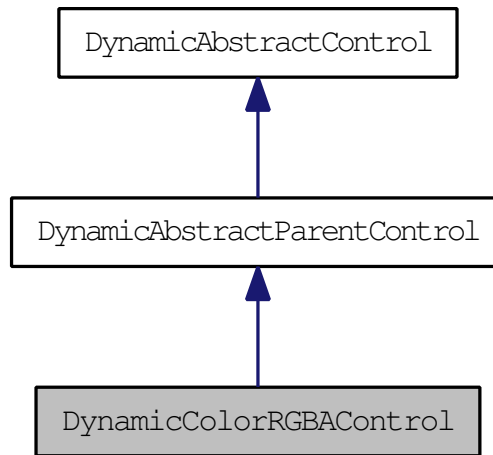
The documentation for this class was generated from the following files:

- **dynamiccolorelementcontrol.h**
- **dynamiccolorelementcontrol.cpp**

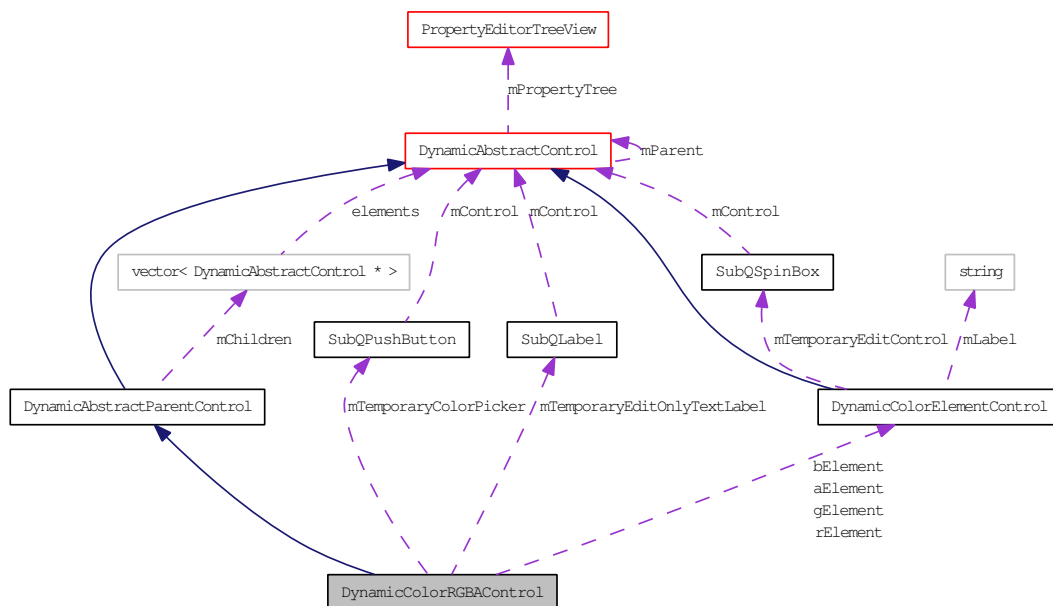
## 4.11 DynamicColorRGBAControl Class Reference

This is the dynamic control for the an RGBA Color picker - used in the property editor It adds a group of child elements to the tree, since you can't edit 3 things in one control easily.

#include <inc/dtQt/dynamiccolorrgbacontrol.h>Inheritance diagram for DynamicColorRGBAControl:



Collaboration diagram for DynamicColorRGBAControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- void **colorPickerPressed** ()  
*Slot - color button is pressed.*
- virtual void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual bool **updateData** (QWidget \*widget)  
*Called when we should take the data out of the controls and put it into the actor.*

## Public Member Functions

- **DynamicColorRGBAControl** ()  
*Constructor.*
- virtual **~DynamicColorRGBAControl** ()  
*Destructor.*
- void **addSelfToParentWidget** (QWidget &parent, QGridLayout &layout, int row)
- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- virtual const QString **getValueAsString** ()
- virtual void **initializeData** (DynamicAbstractControl \*newParent, PropertyEditorModel \*model, dtDAL::PropertyContainer \*pc, dtDAL::ActorProperty \*property)
- virtual void **installEventFilterOnControl** (QObject \*filterObj)
- virtual bool **isEditable** ()
- virtual bool **NeedsPersistentEditor** ()
- virtual void **updateEditorFromModel** (QWidget \*widget)
- virtual bool **updateModelFromEditor** (QWidget \*widget)

## Protected Member Functions

- **DynamicColorElementControl** \* **CreateElementControl** (int index, const std::string &label, PropertyEditorModel \*newModel, dtDAL::PropertyContainer \*newPC)

### 4.11.1 Detailed Description

This is the dynamic control for the an RGBA Color picker - used in the property editor It adds a group of child elements to the tree, since you can't edit 3 things in one control easily.

### 4.11.2 Constructor & Destructor Documentation

#### 4.11.2.1 DynamicColorRGBAControl ()

Constructor.

#### 4.11.2.2 ~DynamicColorRGBAControl () [virtual]

Destructor.

### 4.11.3 Member Function Documentation

#### 4.11.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.11.3.2 void addSelfToParentWidget (QWidget & *parent*, QGridLayout & *layout*, int *row*)

See also `DynamicAbstractControl::addSelfToParentWidget`

#### 4.11.3.3 void colorPickerPressed () [slot]

Slot - color button is pressed.

**4.11.3.4** `QWidget * createEditor (QWidget * parent, const QStyleOptionViewItem & option, const QModelIndex & index) [virtual]`

See also `DynamicAbstractControl::createEditor` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

**4.11.3.5** `DynamicColorElementControl * CreateElementControl (int index, const std::string & label, PropertyEditorModel * newModel, dtDAL::PropertyContainer * newPC) [protected]`

**4.11.3.6** `const QString getDescription () [virtual]`

See also `DynamicAbstractControl::getDescription` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

**4.11.3.7** `const QString getDisplayName () [virtual]`

See also `DynamicAbstractControl::getDisplayName` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.11.3.8** `const QString getValueAsString () [virtual]`

See also `DynamicAbstractControl::getValueAsString` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.11.3.9** `void handleSubEditDestroy (QWidget * widget, QAbstractItemDelegate::EndEditHint hint = QAbstractItemDelegate::NoHint) [virtual, slot]`

See also `DynamicAbstractControl::handleSubEditDestroy` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.11.3.10** `void InitializeData (DynamicAbstractControl * newParent, PropertyEditorModel * model, dtDAL::PropertyContainer * pc, dtDAL::ActorProperty * property) [virtual]`

See also `DynamicAbstractControl::InitializeData` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.11.3.11** `void installEventFilterOnControl (QObject * filterObj) [virtual]`

See also `DynamicAbstractControl::installEventFilterOnControl`

**4.11.3.12** `bool isEditable () [virtual]`

See also `DynamicAbstractControl::isEditable` (p. 26)

Reimplemented from `DynamicAbstractControl` (p. 26).

**4.11.3.13** `bool NeedsPersistentEditor () [virtual]`

See also `DynamicAbstractControl::NeedsPersistentEditor` (p. 26)

Reimplemented from `DynamicAbstractControl` (p. 26).

**4.11.3.14** `bool updateData (QWidget * widget) [virtual, slot]`

Called when we should take the data out of the controls and put it into the actor. This is typically trapped to a lost focus or return pressed or similar user event behavior. This can also be called by the parent control at the moment we change selection. Returns Returns true if any data was actually changed and successfully set on the control - This is purely virtual

Implements `DynamicAbstractControl` (p. 27).

**4.11.3.15** `void updateEditorFromModel (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateEditorFromModel` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

**4.11.3.16 bool updateModelFromEditor (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateModelFromEditor** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

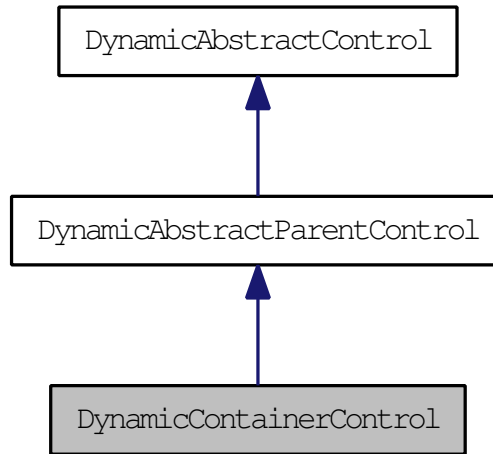
The documentation for this class was generated from the following files:

- **dynamiccolorrrgbacontrol.h**
- **dynamiccolorrrgbacontrol.cpp**

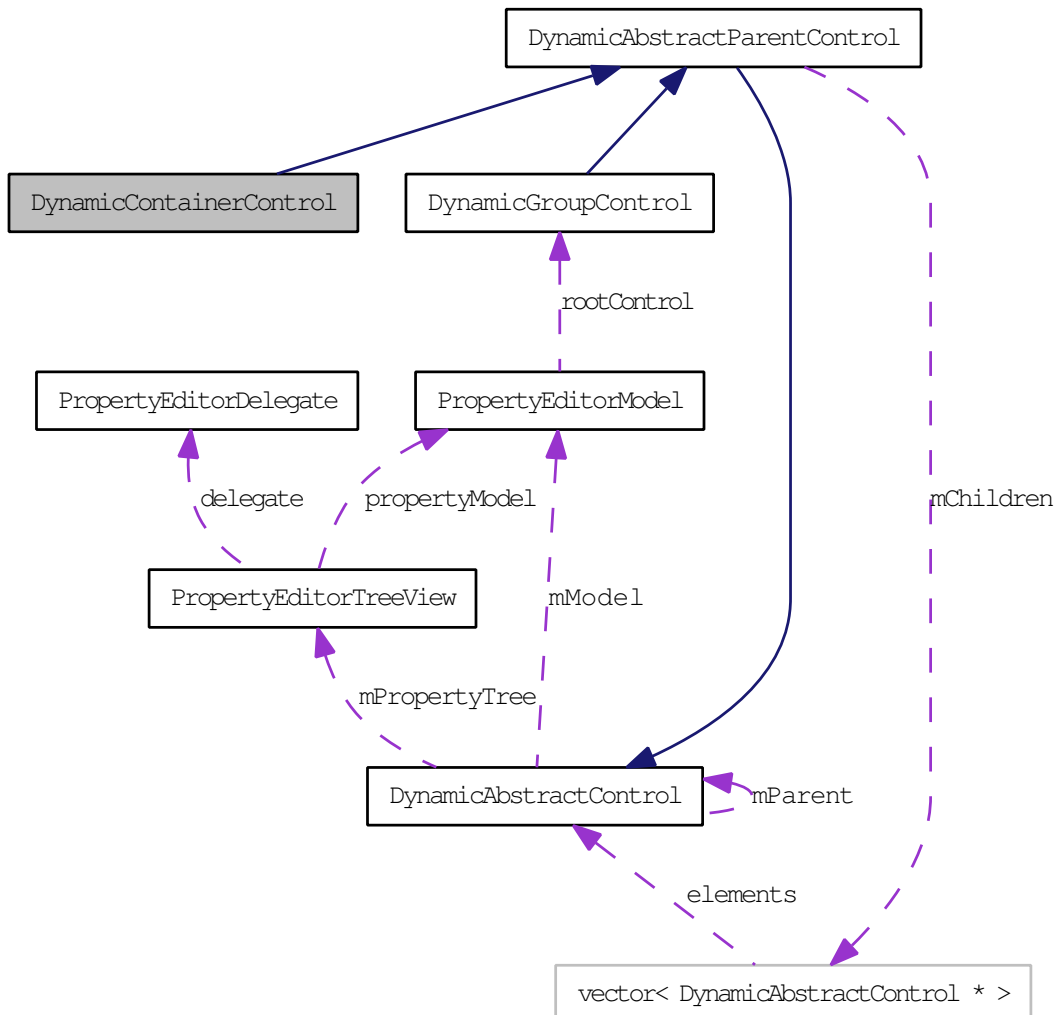
## 4.12 DynamicContainerControl Class Reference

This is the dynamic control for the float data type - used in the property editor.

`#include <inc/dtQt/dynamiccontainercontrol.h>`Inheritance diagram for DynamicContainerControl:



Collaboration diagram for DynamicContainerControl:



## Public Slots

- virtual bool **updateData** (QWidget \*widget)  
*Signal when the data is updated.*

## Public Member Functions

- **DynamicContainerControl** ()  
*Constructor.*
- **~DynamicContainerControl** ()  
*Destructor.*
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- virtual const QString **getValueAsString** ()
- virtual void **initializeData** (DynamicAbstractControl \*newParent, PropertyEditorModel \*model, dtDAL::PropertyContainer \*newPC, dtDAL::ActorProperty \*property)
- virtual bool **isEditable** ()

### 4.12.1 Detailed Description

This is the dynamic control for the float data type - used in the property editor.

### 4.12.2 Constructor & Destructor Documentation

#### 4.12.2.1 DynamicContainerControl ()

Constructor.

#### 4.12.2.2 ~DynamicContainerControl ()

Destructor.

### 4.12.3 Member Function Documentation

#### 4.12.3.1 const QString getDescription () [virtual]

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.12.3.2 const QString getDisplayName () [virtual]

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.12.3.3 const QString getValueAsString () [virtual]

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.12.3.4 void initializeData (DynamicAbstractControl \* newParent, PropertyEditorModel \* model, dtDAL::PropertyContainer \* newPC, dtDAL::ActorProperty \* property) [virtual]

See also **DynamicAbstractControl::initializeData** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.12.3.5 bool isEditable () [virtual]

See also **DynamicAbstractControl::isEditable** (p. 26)

Reimplemented from **DynamicAbstractControl** (p. 26).

**4.12.3.6 bool updateData (QWidget \* *widget*) [virtual, slot]**

Signal when the data is updated.

Implements **DynamicAbstractControl** (p. 27).

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

- **dynamiccontainercontrol.h**
- **dynamiccontainercontrol.cpp**

## 4.13 DynamicControlFactory Class Reference

```
#include <inc/dtQt/dynamicabstractcontrol.h>
```

### Public Member Functions

- **DynamicControlFactory** ()
- **DynamicAbstractControl \* CreateDynamicControl** (const dtDAL::ActorProperty &prop)
- **template<typename DynControlType >**  
void **RegisterControlForDataType** (dtDAL::DataType &dataType)

### 4.13.1 Constructor & Destructor Documentation

#### 4.13.1.1 DynamicControlFactory ()

### 4.13.2 Member Function Documentation

#### 4.13.2.1 DynamicAbstractControl \* CreateDynamicControl (const dtDAL::ActorProperty & *prop*)

#### 4.13.2.2 void RegisterControlForDataType (dtDAL::DataType & *dataType*) [inLine]

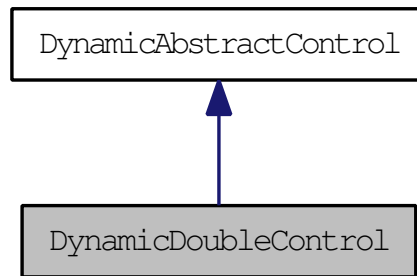
The documentation for this class was generated from the following files:

- **dynamicabstractcontrol.h**
- **dynamicabstractcontrol.cpp**

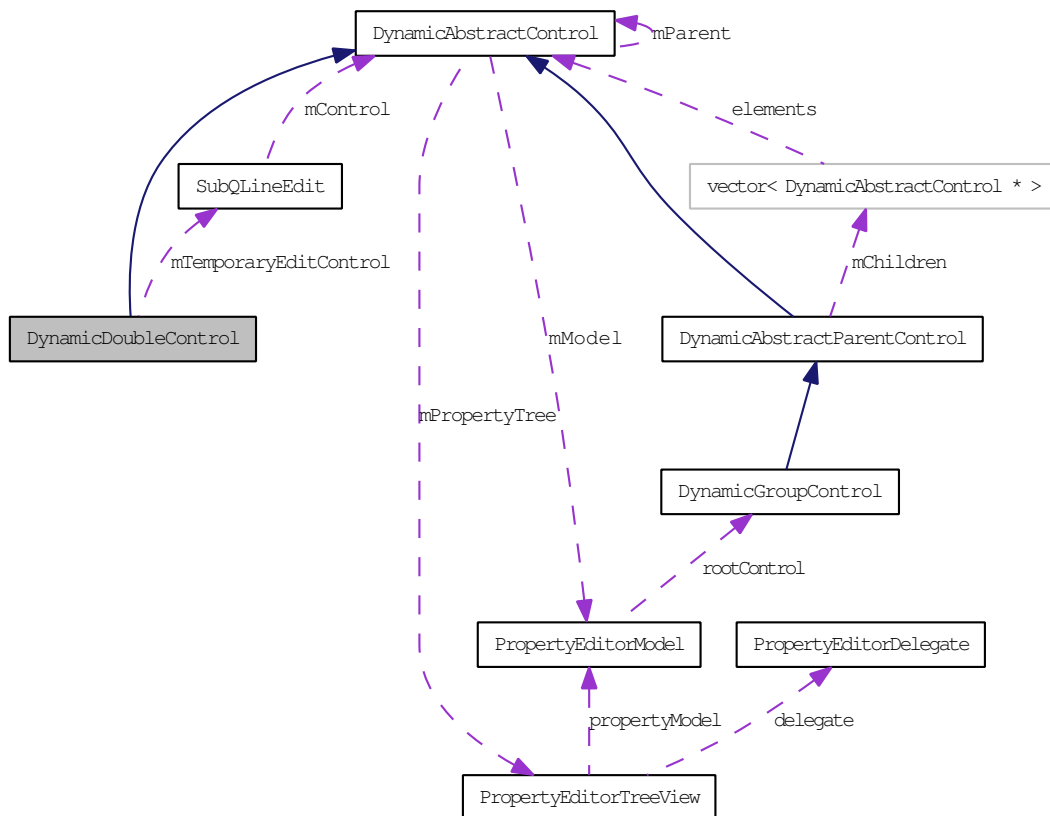
## 4.14 DynamicDoubleControl Class Reference

This is the dynamic control for the double data type - used in the property editor.

#include <inc/dtQt/dynamicdoublecontrol.h> Inheritance diagram for DynamicDoubleControl:



Collaboration diagram for DynamicDoubleControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual bool **updateData** (QWidget \*widget)

### Public Member Functions

- **DynamicDoubleControl** ()

*Constructor.*

- virtual `~DynamicDoubleControl ()`

*Destructor.*

- virtual `QWidget * createEditor (QWidget *parent, const QStyleOptionViewItem &option, const QModelIndex &index)`
- virtual const `QString getDescription ()`
- virtual const `QString getDisplayName ()`
- virtual const `QString getValueAsString ()`
- virtual void `InitializeData (DynamicAbstractControl *newParent, PropertyEditorModel *model, dtDAL::PropertyContainer *pc, dtDAL::ActorProperty *property)`
- virtual bool `isEditable ()`
- virtual void `updateEditorFromModel (QWidget *widget)`
- virtual bool `updateModelFromEditor (QWidget *widget)`

#### 4.14.1 Detailed Description

This is the dynamic control for the double data type - used in the property editor. Note This is almost identical to the float control, but has different data types.

#### 4.14.2 Constructor & Destructor Documentation

##### 4.14.2.1 DynamicDoubleControl ()

Constructor.

##### 4.14.2.2 ~DynamicDoubleControl () [virtual]

Destructor.

#### 4.14.3 Member Function Documentation

##### 4.14.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from `DynamicAbstractControl` (p. 24).

##### 4.14.3.2 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

See also `DynamicAbstractControl::createEditor` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

##### 4.14.3.3 const QString getDescription () [virtual]

See also `DynamicAbstractControl::getDescription` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

##### 4.14.3.4 const QString getDisplayName () [virtual]

See also `DynamicAbstractControl::getDisplayName` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

##### 4.14.3.5 const QString getValueAsString () [virtual]

See also `DynamicAbstractControl::getValueAsString` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.14.3.6** `void handleSubEditDestroy (QWidget * widget, QAbstractItemDelegate::EndEditHint hint = QAbstractItemDelegate::NoHint) [inline, virtual, slot]`

See also `DynamicAbstractControl::handleSubEditDestroy` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.14.3.7** `void InitializeData (DynamicAbstractControl * newParent, PropertyEditorModel * model, dtDAL::PropertyContainer * pc, dtDAL::ActorProperty * property) [virtual]`

See also `DynamicAbstractControl::InitializeData` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.14.3.8** `bool isEditable () [virtual]`

See also `DynamicAbstractControl::isEditable` (p. 26)

Reimplemented from `DynamicAbstractControl` (p. 26).

**4.14.3.9** `bool updateData (QWidget * widget) [virtual, slot]`

See also `DynamicAbstractControl::updateData` (p. 27)

Implements `DynamicAbstractControl` (p. 27).

**4.14.3.10** `void updateEditorFromModel (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateEditorFromModel` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

**4.14.3.11** `bool updateModelFromEditor (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateModelFromEditor` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

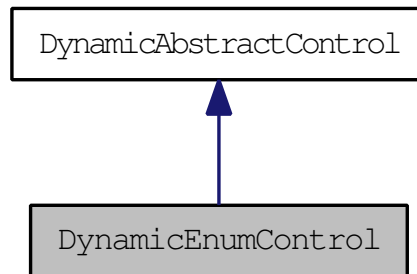
The documentation for this class was generated from the following files:

- `dynamicdoublecontrol.h`
- `dynamicdoublecontrol.cpp`

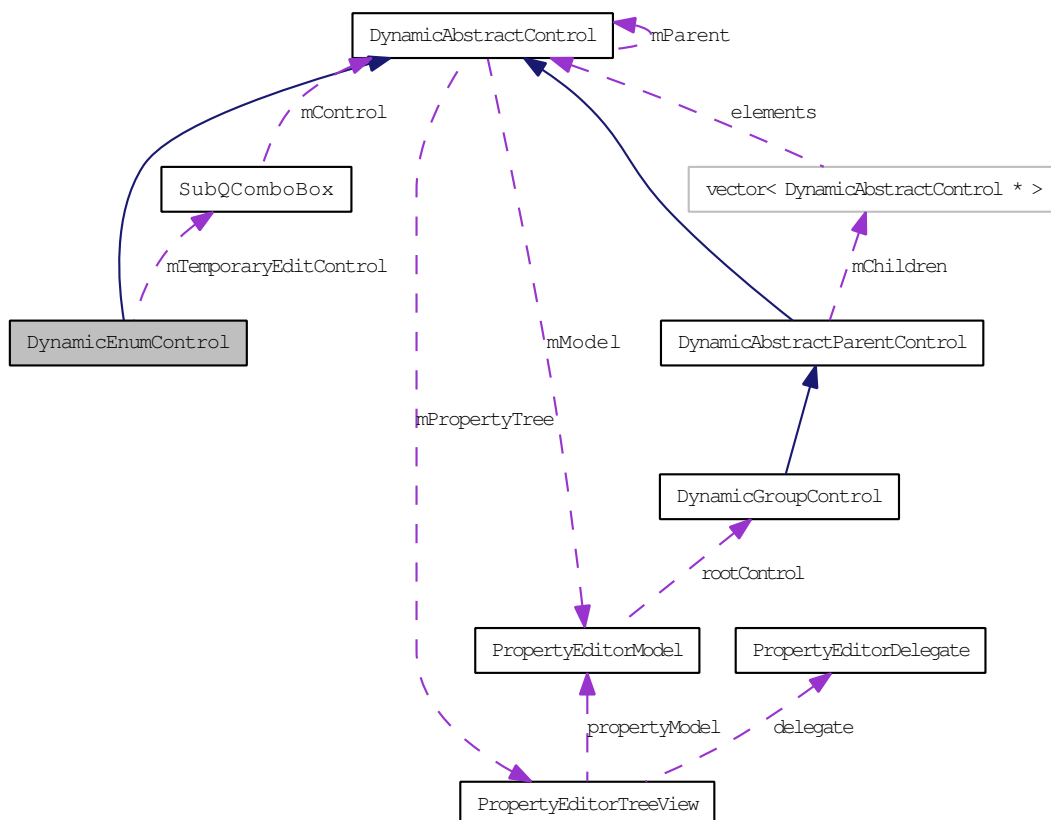
## 4.15 DynamicEnumControl Class Reference

This is the dynamic control for the enum data type - used in the property editor.

#include <inc/dtQt/dynamicenumcontrol.h> Inheritance diagram for DynamicEnumControl:



Collaboration diagram for DynamicEnumControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
When a property changes, we have to update our editor.
- void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- void **itemSelected** (int index)  
Called when the user selects an item in the combo box.
- virtual bool **updateData** (QWidget \*widget)

## Public Member Functions

- **DynamicEnumControl** ()  
*Constructor.*
- virtual **~DynamicEnumControl** ()  
*Destructor.*
- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- virtual const QString **getValueAsString** ()
- virtual void **initializeData** (DynamicAbstractControl \*newParent, PropertyEditorModel \*model, dtDAL::PropertyContainer \*pc, dtDAL::ActorProperty \*property)
- virtual bool **isEditable** ()
- virtual void **updateEditorFromModel** (QWidget \*widget)
- virtual bool **updateModelFromEditor** (QWidget \*widget)

### 4.15.1 Detailed Description

This is the dynamic control for the enum data type - used in the property editor.

### 4.15.2 Constructor & Destructor Documentation

#### 4.15.2.1 DynamicEnumControl ()

Constructor.

#### 4.15.2.2 ~DynamicEnumControl () [virtual]

Destructor.

### 4.15.3 Member Function Documentation

#### 4.15.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & propCon, dtDAL::ActorProperty & property) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.15.3.2 QWidget \* createEditor (QWidget \* parent, const QStyleOptionViewItem & option, const QModelIndex & index) [virtual]

See also **DynamicAbstractControl::createEditor** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.15.3.3 const QString getDescription () [virtual]

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.15.3.4 const QString getDisplayName () [virtual]

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.15.3.5 const QString getValueAsString () [virtual]**

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.15.3.6 void handleSubEditDestroy (QWidget \* *widget*, QAbstractItemDelegate::EndEditHint *hint* = QAbstractItemDelegate::NoHint) [inline, virtual, slot]**

See also **DynamicAbstractControl::handleSubEditDestroy** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.15.3.7 void InitializeData (DynamicAbstractControl \* *newParent*, PropertyEditorModel \* *model*, dtDAL::PropertyContainer \* *pc*, dtDAL::ActorProperty \* *property*) [virtual]**

See also **DynamicAbstractControl::InitializeData** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.15.3.8 bool isEditable () [virtual]**

See also **DynamicAbstractControl::isEditable** (p. 26)

Reimplemented from **DynamicAbstractControl** (p. 26).

**4.15.3.9 void itemSelected (int *index*) [slot]**

Called when the user selects an item in the combo box.

**4.15.3.10 bool updateData (QWidget \* *widget*) [virtual, slot]**

See also **DynamicAbstractControl::updateData** (p. 27)

Implements **DynamicAbstractControl** (p. 27).

**4.15.3.11 void updateEditorFromModel (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateEditorFromModel** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

**4.15.3.12 bool updateModelFromEditor (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateModelFromEditor** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

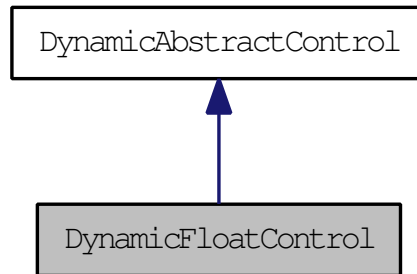
The documentation for this class was generated from the following files:

- **dynamicenumcontrol.h**
- **dynamicenumcontrol.cpp**

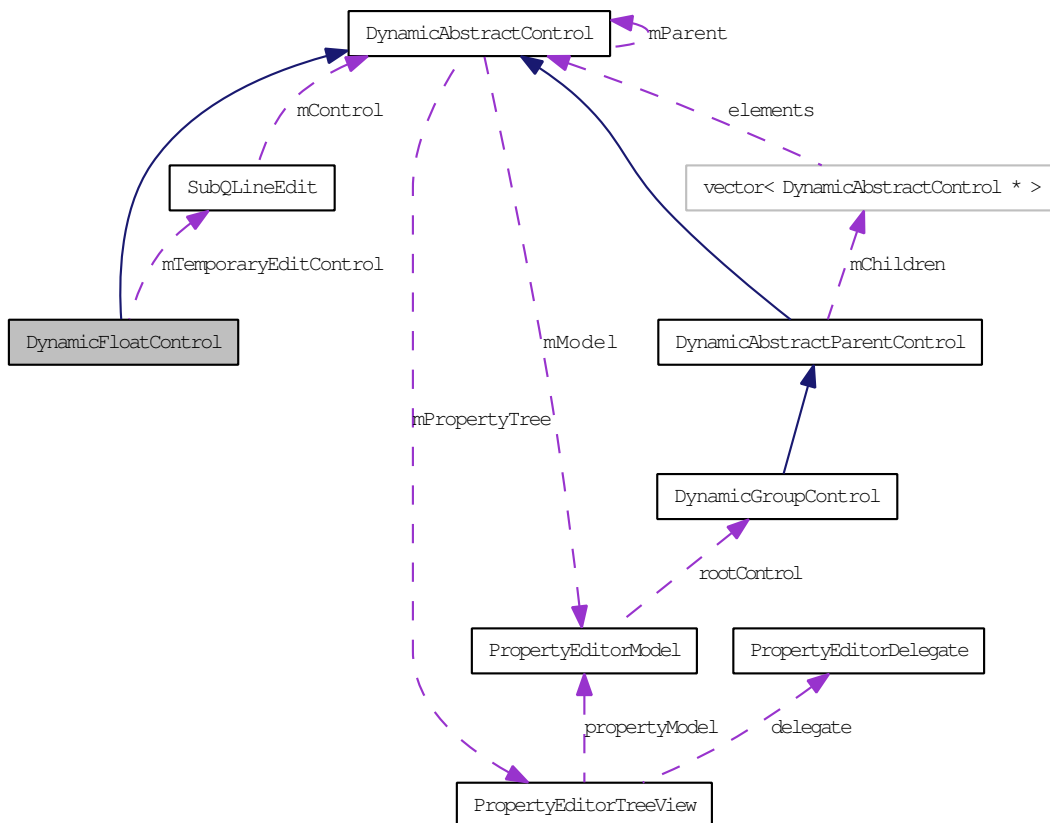
## 4.16 DynamicFloatControl Class Reference

This is the dynamic control for the float data type - used in the property editor.

#include <inc/dtQt/dynamicfloatcontrol.h> Inheritance diagram for DynamicFloatControl:



Collaboration diagram for DynamicFloatControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- virtual bool **updateData** (QWidget \*widget)

### Public Member Functions

- **DynamicFloatControl** ()  
*Constructor.*

- virtual `~DynamicFloatControl ()`  
*Destructor.*
- virtual `QWidget * createEditor (QWidget *parent, const QStyleOptionViewItem &option, const QModelIndex &index)`
- virtual const `QString getDescription ()`
- virtual const `QString getDisplayName ()`
- virtual const `QString getValueAsString ()`
- void `handleSubEditDestroy (QWidget *widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)`
- virtual void `initializeData (DynamicAbstractControl *newParent, PropertyEditorModel *model, dtDAL::PropertyContainer *newPC, dtDAL::ActorProperty *property)`
- virtual bool `isEditable ()`
- virtual void `updateEditorFromModel (QWidget *widget)`
- virtual bool `updateModelFromEditor (QWidget *widget)`

#### 4.16.1 Detailed Description

This is the dynamic control for the float data type - used in the property editor.

#### 4.16.2 Constructor & Destructor Documentation

##### 4.16.2.1 DynamicFloatControl ()

Constructor.

##### 4.16.2.2 ~DynamicFloatControl () [virtual]

Destructor.

#### 4.16.3 Member Function Documentation

##### 4.16.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from `DynamicAbstractControl` (p. 24).

##### 4.16.3.2 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

See also `DynamicAbstractControl::createEditor` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

##### 4.16.3.3 const QString getDescription () [virtual]

See also `DynamicAbstractControl::getDescription` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

##### 4.16.3.4 const QString getDisplayName () [virtual]

See also `DynamicAbstractControl::getDisplayName` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

##### 4.16.3.5 const QString getValueAsString () [virtual]

See also `DynamicAbstractControl::getValueAsString` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.16.3.6** `void handleSubEditDestroy (QWidget * widget, QAbstractItemDelegate::EndEditHint hint = QAbstractItemDelegate::NoHint) [inline, virtual]`

See also `DynamicAbstractControl::handleSubEditDestroy` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.16.3.7** `void InitializeData (DynamicAbstractControl * newParent, PropertyEditorModel * model, dtDAL::PropertyContainer * newPC, dtDAL::ActorProperty * property) [virtual]`

See also `DynamicAbstractControl::InitializeData` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.16.3.8** `bool isEditable () [virtual]`

See also `DynamicAbstractControl::isEditable` (p. 26)

Reimplemented from `DynamicAbstractControl` (p. 26).

**4.16.3.9** `bool updateData (QWidget * widget) [virtual, slot]`

See also `DynamicAbstractControl::updateData` (p. 27)

Implements `DynamicAbstractControl` (p. 27).

**4.16.3.10** `void updateEditorFromModel (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateEditorFromModel` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

**4.16.3.11** `bool updateModelFromEditor (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateModelFromEditor` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

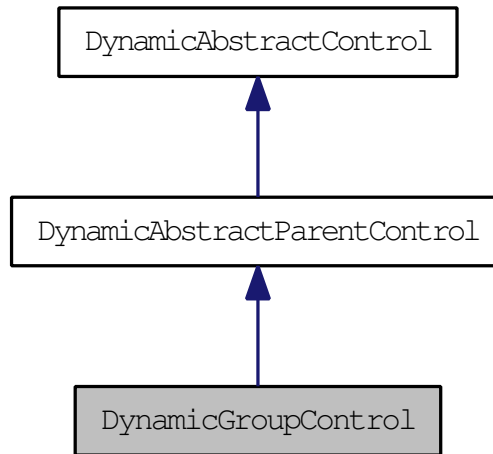
The documentation for this class was generated from the following files:

- `dynamicfloatcontrol.h`
- `dynamicfloatcontrol.cpp`

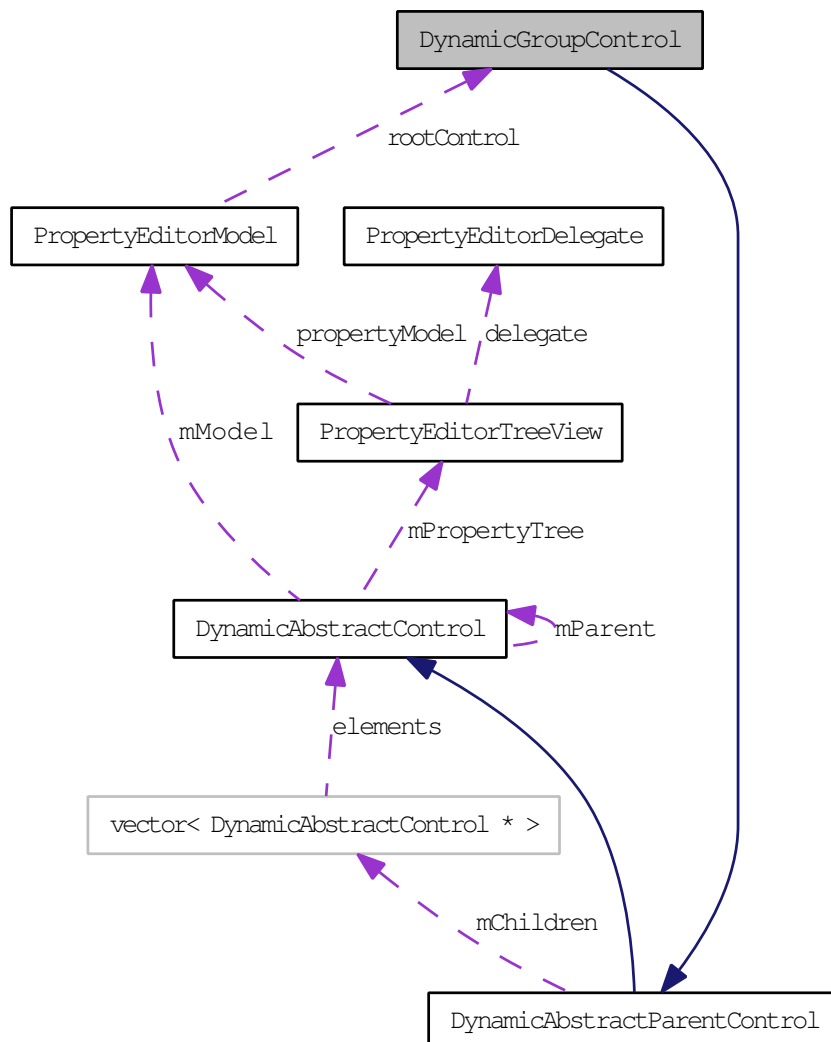
## 4.17 DynamicGroupControl Class Reference

This is the dynamic control for the Group data type - used in the property editor. The primary purpose of the group control is to provide a visual grouping of property types so that they aren't all laid out together.

#include <inc/dtQt/dynamicgroupcontrol.h> Inheritance diagram for DynamicGroupControl:



Collaboration diagram for DynamicGroupControl:



## Public Slots

- virtual bool **updateData** (QWidget \*widget)

## Public Member Functions

- **DynamicGroupControl** (const std::string &newName)  
*Constructor.*
- virtual ~**DynamicGroupControl** ()  
*Destructor.*
- void **addChildControl** (**DynamicAbstractControl** \*child, **PropertyEditorModel** \*model)  
*Groups can have children.*
- void **addSelfToParentWidget** (QWidget &parent, QGridLayout &layout, int row)
- **DynamicGroupControl** \* **getChildGroupControl** (QString name)  
*Attempt to find a group control with the passed in name.*
- virtual const QString **getDisplayName** ()

### 4.17.1 Detailed Description

This is the dynamic control for the Group data type - used in the property editor The primary purpose of the group control is to provide a visual grouping of property types so that they aren't all laid out together.

### 4.17.2 Constructor & Destructor Documentation

#### 4.17.2.1 DynamicGroupControl (const std::string & newName)

Constructor.

#### 4.17.2.2 ~DynamicGroupControl () [virtual]

Destructor.

### 4.17.3 Member Function Documentation

#### 4.17.3.1 void addChildControl (DynamicAbstractControl \* child, PropertyEditorModel \* model)

Groups can have children. This is how you add children to the group. Note that you can't remove a child once it's added.

#### 4.17.3.2 void addSelfToParentWidget (QWidget & parent, QGridLayout & layout, int row)

See also `DynamicAbstractControl::addSelfToParentWidget`

#### 4.17.3.3 DynamicGroupControl \* getChildGroupControl (QString name)

Attempt to find a group control with the passed in name. This is used primarily on the root object to find an existing group. However, it could easily be used for nested groups once that is supported.

#### 4.17.3.4 const QString getDisplayName () [virtual]

See also `DynamicAbstractControl::getDisplayName` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

#### 4.17.3.5 bool updateData (QWidget \* widget) [virtual, slot]

See also `DynamicAbstractControl::updateData` (p. 27)

Implements `DynamicAbstractControl` (p. 27).

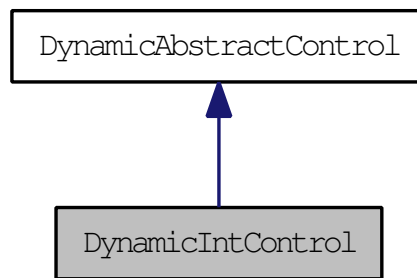
The documentation for this class was generated from the following files:

- `dynamicgroupcontrol.h`
- `dynamicgroupcontrol.cpp`

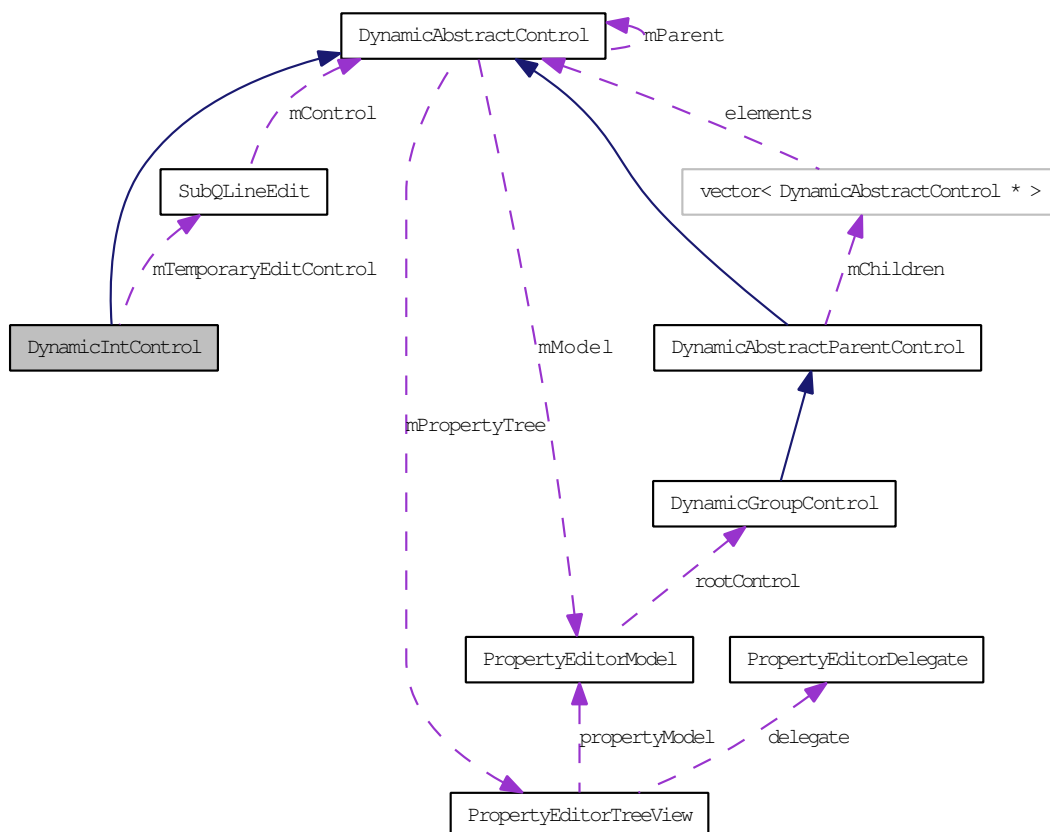
## 4.18 DynamicIntControl Class Reference

This is the dynamic control for the int data type - used in the property editor.

#include <inc/dtQt/dynamicintcontrol.h> Inheritance diagram for DynamicIntControl:



Collaboration diagram for DynamicIntControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
When a property changes, we have to update our editor.
- void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual bool **updateData** (QWidget \*widget)

### Public Member Functions

- **DynamicIntControl** ()

*Constructor.*

- virtual `~DynamicIntControl ()`

*Destructor.*

- virtual `QWidget * createEditor (QWidget *parent, const QStyleOptionViewItem &option, const QModelIndex &index)`
- virtual const `QString getDescription ()`
- virtual const `QString getDisplayName ()`
- virtual const `QString getValueAsString ()`
- virtual void `initializeData (DynamicAbstractControl *newParent, PropertyEditorModel *model, dtDAL::PropertyContainer *newPC, dtDAL::ActorProperty *property)`
- virtual bool `isEditable ()`
- virtual void `updateEditorFromModel (QWidget *widget)`
- virtual bool `updateModelFromEditor (QWidget *widget)`

### 4.18.1 Detailed Description

This is the dynamic control for the int data type - used in the property editor.

### 4.18.2 Constructor & Destructor Documentation

#### 4.18.2.1 DynamicIntControl ()

Constructor.

#### 4.18.2.2 ~DynamicIntControl () [virtual]

Destructor.

### 4.18.3 Member Function Documentation

#### 4.18.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from `DynamicAbstractControl` (p. 24).

#### 4.18.3.2 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

See also `DynamicAbstractControl::createEditor` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

#### 4.18.3.3 const QString getDescription () [virtual]

See also `DynamicAbstractControl::getDescription` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

#### 4.18.3.4 const QString getDisplayName () [virtual]

See also `DynamicAbstractControl::getDisplayName` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

#### 4.18.3.5 const QString getValueAsString () [virtual]

See also `DynamicAbstractControl::getValueAsString` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.18.3.6** `void handleSubEditDestroy (QWidget * widget, QAbstractItemDelegate::EndEditHint hint = QAbstractItemDelegate::NoHint) [inline, virtual, slot]`

See also `DynamicAbstractControl::handleSubEditDestroy` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.18.3.7** `void InitializeData (DynamicAbstractControl * newParent, PropertyEditorModel * model, dtDAL::PropertyContainer * newPC, dtDAL::ActorProperty * property) [virtual]`

See also `DynamicAbstractControl::InitializeData` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.18.3.8** `bool isEditable () [virtual]`

See also `DynamicAbstractControl::isEditable` (p. 26)

Reimplemented from `DynamicAbstractControl` (p. 26).

**4.18.3.9** `bool updateData (QWidget * widget) [virtual, slot]`

See also `DynamicAbstractControl::updateData` (p. 27)

Implements `DynamicAbstractControl` (p. 27).

**4.18.3.10** `void updateEditorFromModel (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateEditorFromModel` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

**4.18.3.11** `bool updateModelFromEditor (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateModelFromEditor` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

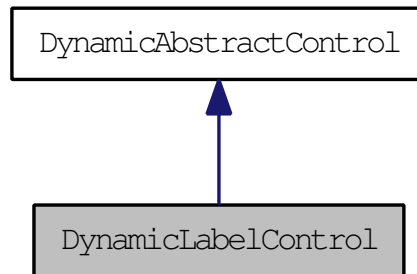
The documentation for this class was generated from the following files:

- `dynamicintcontrol.h`
- `dynamicintcontrol.cpp`

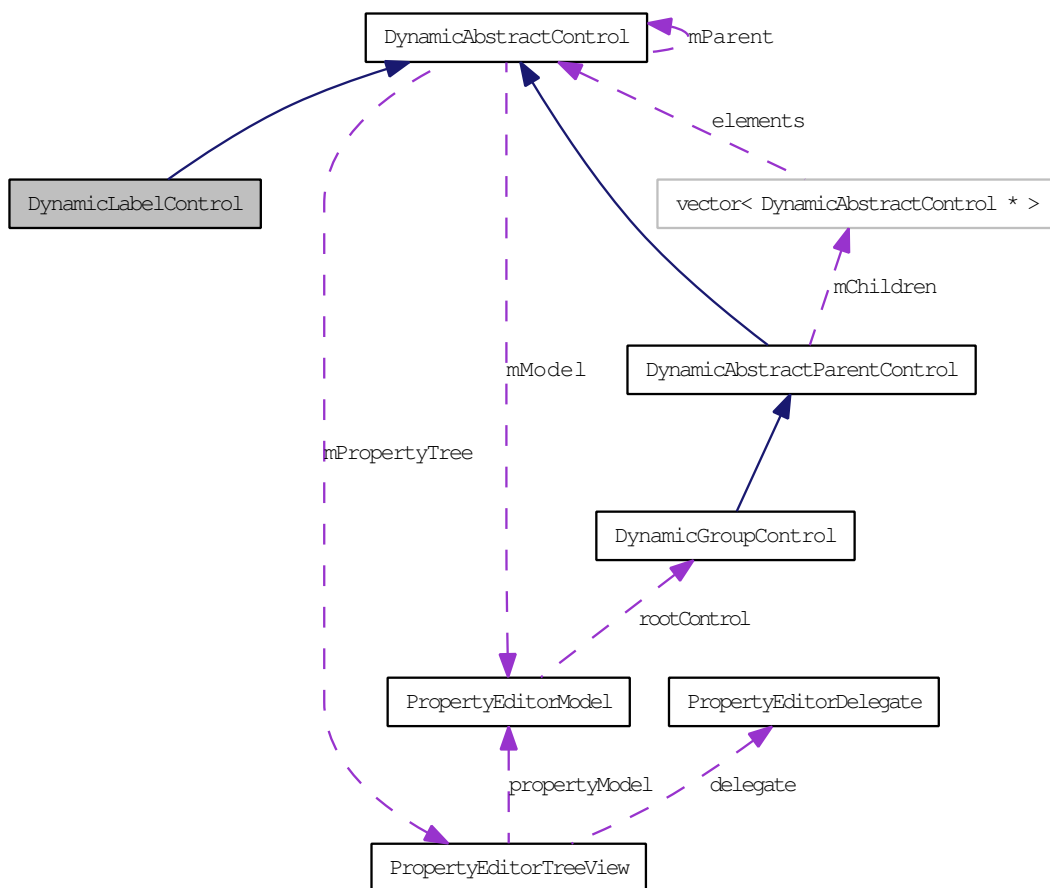
## 4.19 DynamicLabelControl Class Reference

This is the dynamic control that is a bit odd.

`#include <inc/dtQt/dynamiclabelcontrol.h>`Inheritance diagram for DynamicLabelControl:



Collaboration diagram for DynamicLabelControl:



### Public Slots

- virtual bool `updateData` (QWidget \*widget)

### Public Member Functions

- `DynamicLabelControl ()`  
*Constructor.*
- virtual `~DynamicLabelControl ()`

*Destructor.*

- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- virtual const QString **getValueAsString** ()
- virtual void **initializeData** (**DynamicAbstractControl** \*newParent, **PropertyEditorModel** \*model, dtDAL::PropertyContainer \*newPC, dtDAL::ActorProperty \*property)
- void **setDisplayValues** (QString name, QString description, QString valueAsString)

*Use this for the 2nd use of this control, when you don't have a real property, but instead have some static strings you are working with.*

### 4.19.1 Detailed Description

This is the dynamic control that is a bit odd. It serves a dual purpose. The first purpose is to work with a label data type (once it exists). The second is just a placeholder to add non-editable strings with no underlying data type.

### 4.19.2 Constructor & Destructor Documentation

#### 4.19.2.1 DynamicLabelControl ()

Constructor.

#### 4.19.2.2 ~DynamicLabelControl () [virtual]

Destructor.

### 4.19.3 Member Function Documentation

#### 4.19.3.1 const QString getDescription () [virtual]

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.19.3.2 const QString getDisplayName () [virtual]

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.19.3.3 const QString getValueAsString () [virtual]

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.19.3.4 void initializeData (DynamicAbstractControl \* newParent, PropertyEditorModel \* model, dtDAL::PropertyContainer \* newPC, dtDAL::ActorProperty \* property) [virtual]

See also **DynamicAbstractControl::initializeData** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.19.3.5 void setDisplayValues (QString name, QString description, QString valueAsString)

Use this for the 2nd use of this control, when you don't have a real property, but instead have some static strings you are working with. If you call this AFTER **initializeData()** (p. 70), these values will override the property values. Note It'd be nice if this could be put in the constructor, however we have to be careful because the factory uses the constructor. Also, using the constructor would prevent overriding the values for a real property

#### 4.19.3.6 bool updateData (QWidget \* widget) [virtual, slot]

See also **DynamicAbstractControl::updateData** (p. 27)

Implements **DynamicAbstractControl** (p. 27).

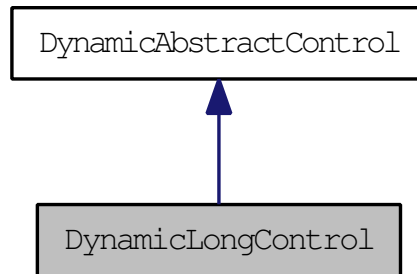
The documentation for this class was generated from the following files:

- [dynamiclabelcontrol.h](#)
- [dynamiclabelcontrol.cpp](#)

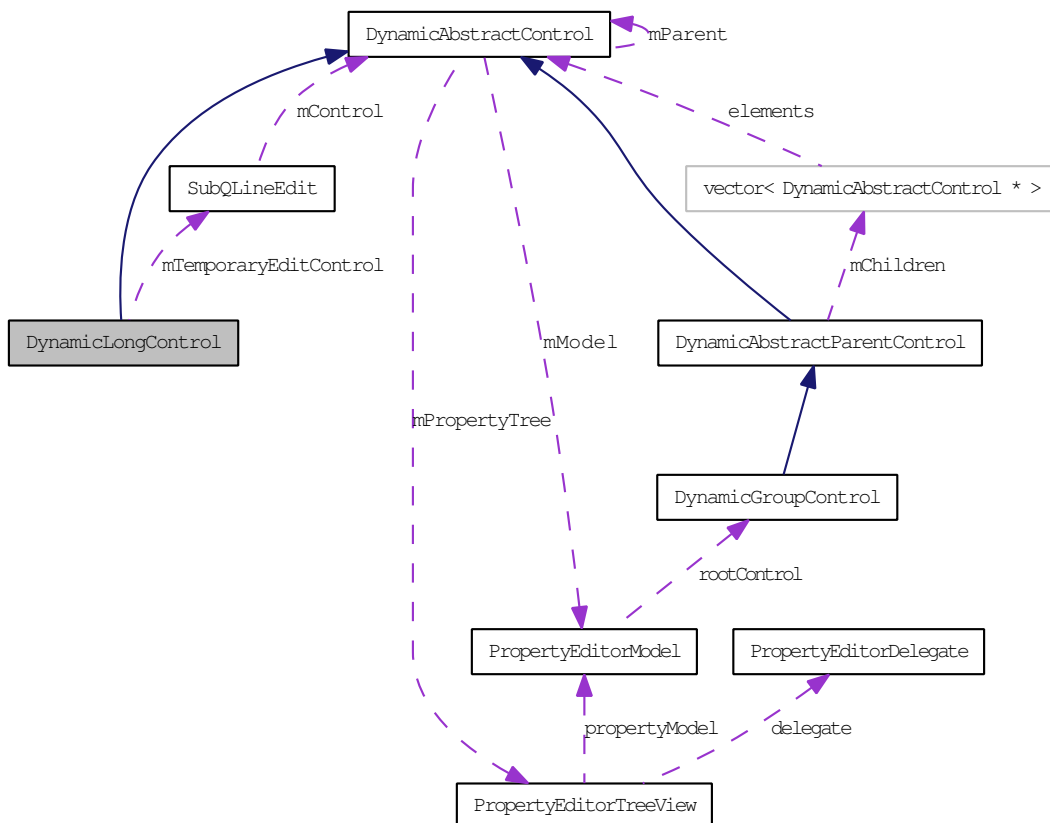
## 4.20 DynamicLongControl Class Reference

This is the dynamic control for the long data type - used in the property editor.

#include <inc/dtQt/dynamiclongcontrol.h> Inheritance diagram for DynamicLongControl:



Collaboration diagram for DynamicLongControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
When a property changes, we have to update our editor.
- void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual bool **updateData** (QWidget \*widget)

### Public Member Functions

- **DynamicLongControl** ()

*Constructor.*

- virtual `~DynamicLongControl ()`

*Destructor.*

- virtual `QWidget * createEditor (QWidget *parent, const QStyleOptionViewItem &option, const QModelIndex &index)`
- virtual const `QString getDescription ()`
- virtual const `QString getDisplayName ()`
- virtual const `QString getValueAsString ()`
- virtual void `InitializeData (DynamicAbstractControl *newParent, PropertyEditorModel *model, dtDAL::PropertyContainer *newPC, dtDAL::ActorProperty *property)`
- virtual bool `isEditable ()`
- virtual void `updateEditorFromModel (QWidget *widget)`
- virtual bool `updateModelFromEditor (QWidget *widget)`

### 4.20.1 Detailed Description

This is the dynamic control for the long data type - used in the property editor. Note This is really the same as int, but we need a separate control anyway.

### 4.20.2 Constructor & Destructor Documentation

#### 4.20.2.1 DynamicLongControl ()

Constructor.

#### 4.20.2.2 ~DynamicLongControl () [virtual]

Destructor.

### 4.20.3 Member Function Documentation

#### 4.20.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from `DynamicAbstractControl` (p. 24).

#### 4.20.3.2 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

See also `DynamicAbstractControl::createEditor` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

#### 4.20.3.3 const QString getDescription () [virtual]

See also `DynamicAbstractControl::getDescription` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

#### 4.20.3.4 const QString getDisplayName () [virtual]

See also `DynamicAbstractControl::getDisplayName` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

#### 4.20.3.5 const QString getValueAsString () [virtual]

See also `DynamicAbstractControl::getValueAsString` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.20.3.6** `void handleSubEditDestroy (QWidget * widget, QAbstractItemDelegate::EndEditHint hint = QAbstractItemDelegate::NoHint) [inline, virtual, slot]`

See also `DynamicAbstractControl::handleSubEditDestroy` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.20.3.7** `void InitializeData (DynamicAbstractControl * newParent, PropertyEditorModel * model, dtDAL::PropertyContainer * newPC, dtDAL::ActorProperty * property) [virtual]`

See also `DynamicAbstractControl::InitializeData` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.20.3.8** `bool isEditable () [virtual]`

See also `DynamicAbstractControl::isEditable` (p. 26)

Reimplemented from `DynamicAbstractControl` (p. 26).

**4.20.3.9** `bool updateData (QWidget * widget) [virtual, slot]`

See also `DynamicAbstractControl::updateData` (p. 27)

Implements `DynamicAbstractControl` (p. 27).

**4.20.3.10** `void updateEditorFromModel (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateEditorFromModel` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

**4.20.3.11** `bool updateModelFromEditor (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateModelFromEditor` (p. 27)

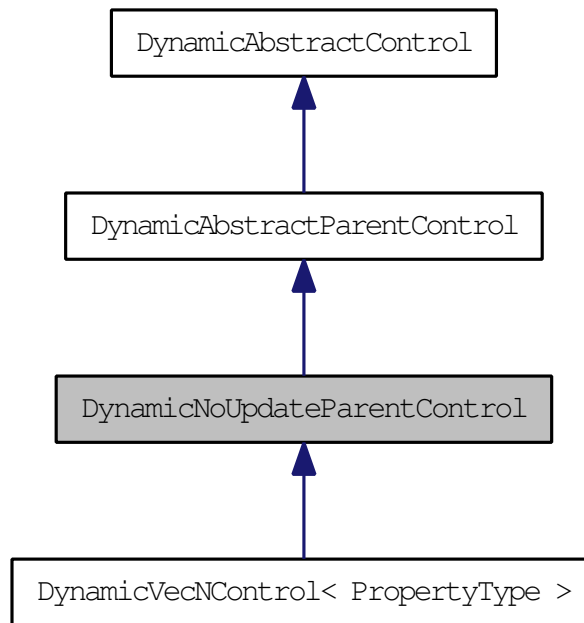
Reimplemented from `DynamicAbstractControl` (p. 27).

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

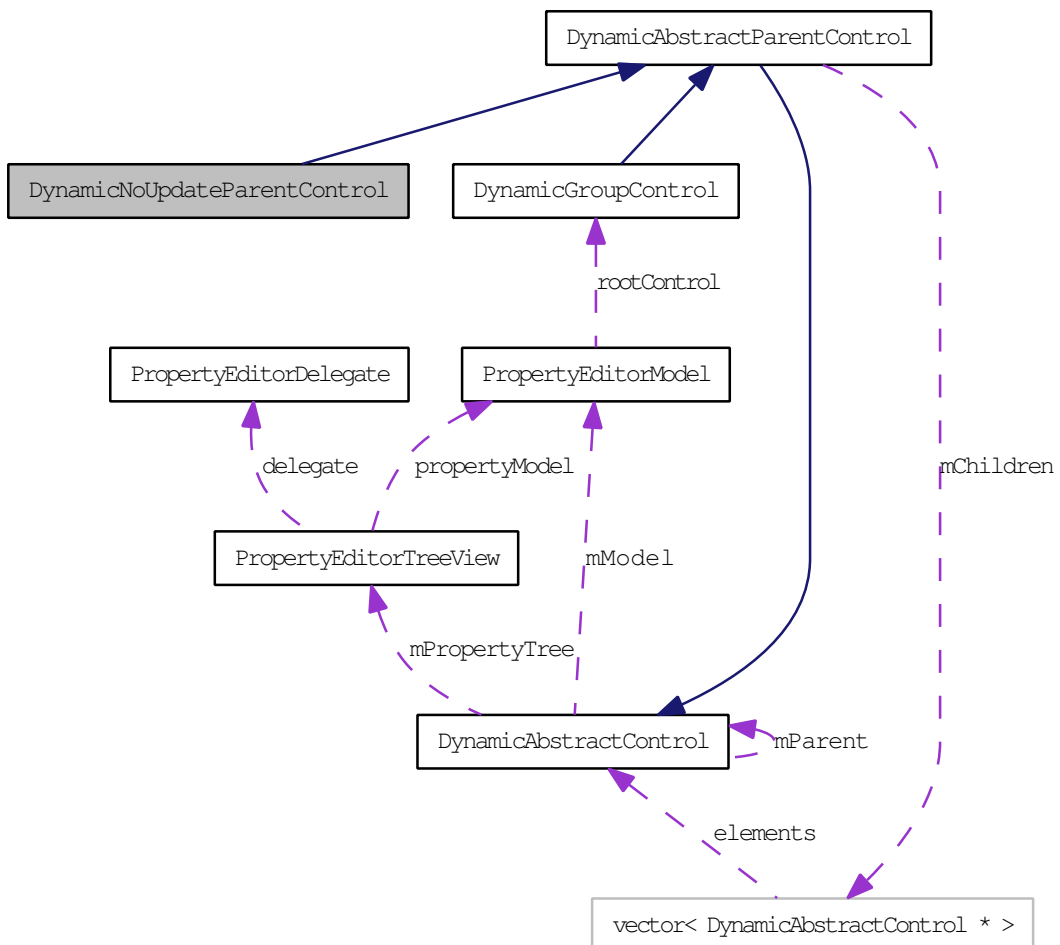
- `dynamiclongcontrol.h`
- `dynamiclongcontrol.cpp`

## 4.21 DynamicNoUpdateParentControl Class Reference

#include <inc/dtQt/dynamicvecncontrol.h> Inheritance diagram for DynamicNoUpdateParentControl:



Collaboration diagram for DynamicNoUpdateParentControl:



## Public Slots

- virtual bool **updateData** (QWidget \*widget)

*Called when we should take the data out of the controls and put it into the actor.*

### 4.21.1 Member Function Documentation

#### 4.21.1.1 virtual bool updateData (QWidget \* widget) [inline, virtual, slot]

Called when we should take the data out of the controls and put it into the actor. This is typically trapped to a lost focus or return pressed or similar user event behavior. This can also be called by the parent control at the moment we change selection. Returns Returns true if any data was actually changed and sucessfully set on the control - This is purely virtual

Implements **DynamicAbstractControl** (p. 27).

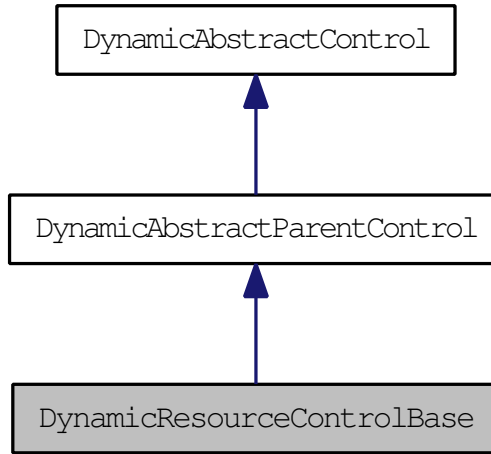
The documentation for this class was generated from the following file:

- **dynamicvecncontrol.h**

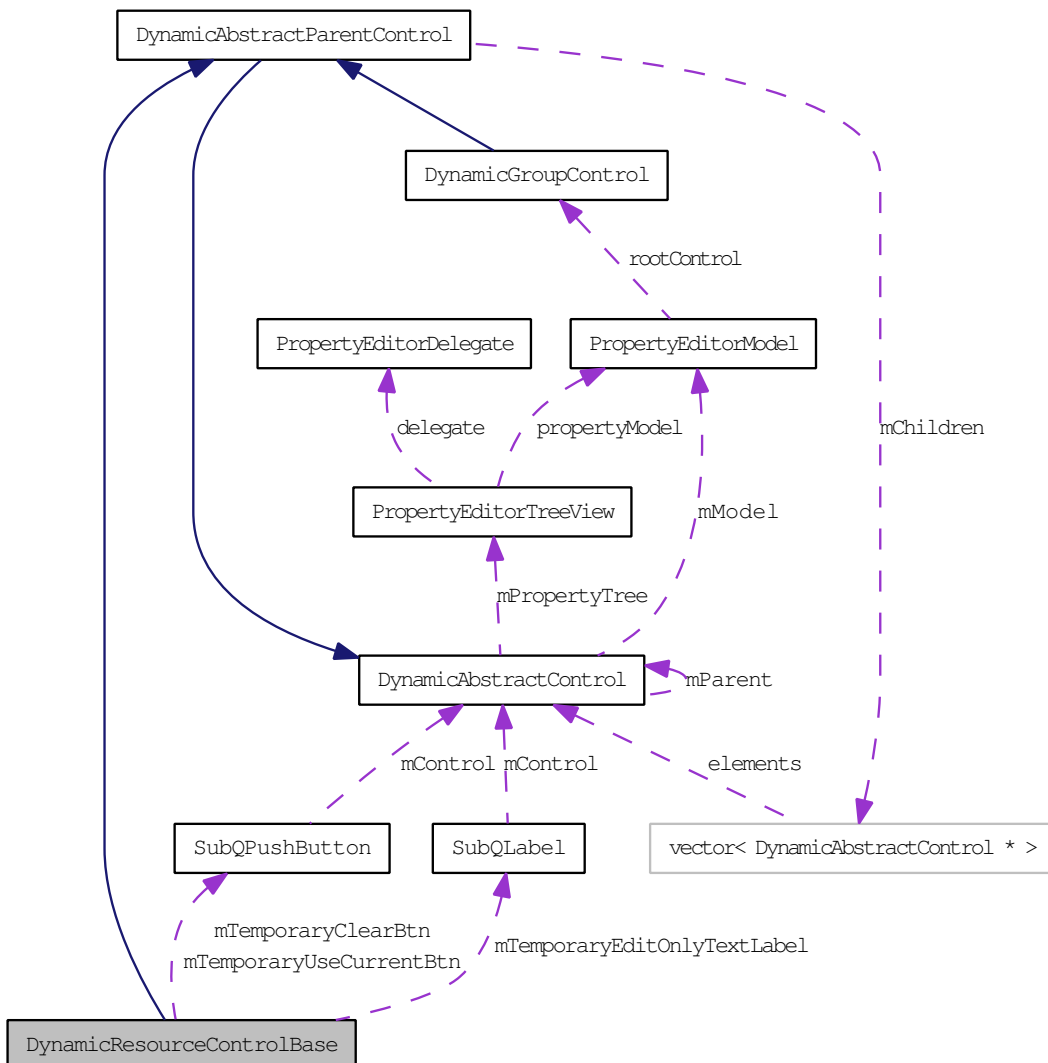
## 4.22 DynamicResourceControlBase Class Reference

This is the resource actor property.

#include <inc/dtQt/dynamicresourcecontrolbase.h> Inheritance diagram for DynamicResourceControlBase:



Collaboration diagram for DynamicResourceControlBase:



## Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- void **clearPressed** ()  
*The user pressed the 'Clear' Button.*
- virtual void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual bool **updateData** (QWidget \*widget)  
*Called when we should take the data out of the controls and put it into the actor.*
- void **useCurrentPressed** ()  
*The user pressed the 'Use Current' Button.*

## Public Member Functions

- **DynamicResourceControlBase** ()  
*Constructor.*
- virtual ~**DynamicResourceControlBase** ()  
*Destructor.*
- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- dtDAL::ResourceActorProperty & **GetProperty** ()
- virtual const QString **getValueAsString** ()
- virtual void **initializeData** (**DynamicAbstractControl** \*newParent, **PropertyEditorModel** \*model, dtDAL::PropertyContainer \*newPC, dtDAL::ActorProperty \*property)
- virtual void **installEventFilterOnControl** (QObject \*filterObj)
- virtual bool **isEditable** ()
- virtual void **updateEditorFromModel** (QWidget \*widget)
- virtual bool **updateModelFromEditor** (QWidget \*widget)

## Protected Member Functions

- virtual dtDAL::ResourceDescriptor **getCurrentResource** ()=0  
*Figure out which resource descriptor to get from EditorData and get it.*

### 4.22.1 Detailed Description

This is the resource actor property. It knows how to work with the various resource data types (Terrain, Character, Mesh, Texture, sound, ...) from DataTypes.h This control is not editable, but has several child controls and some of them are editable.

### 4.22.2 Constructor & Destructor Documentation

#### 4.22.2.1 DynamicResourceControlBase ()

Constructor.

#### 4.22.2.2 ~DynamicResourceControlBase () [virtual]

Destructor.

### 4.22.3 Member Function Documentation

#### 4.22.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.22.3.2 void clearPressed () [slot]

The user pressed the 'Clear' Button. Clear out the resource.

#### 4.22.3.3 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

See also **DynamicAbstractControl::createEditor** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.22.3.4 virtual dtDAL::ResourceDescriptor getCurrentResource () [protected, pure virtual]

Figure out which resource descriptor to get from EditorData and get it. Returns the current resource descriptor for our type, else an empty one if type is invalid.

#### 4.22.3.5 const QString getDescription () [virtual]

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.22.3.6 const QString getDisplayName () [virtual]

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.22.3.7 dtDAL::ResourceActorProperty & GetProperty ()

#### 4.22.3.8 const QString getValueAsString () [virtual]

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.22.3.9 void handleSubEditDestroy (QWidget \* *widget*, QAbstractItemDelegate::EndEditHint *hint* = QAbstractItemDelegate::NoHint) [virtual, slot]

See also **DynamicAbstractControl::handleSubEditDestroy** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.22.3.10 void InitializeData (DynamicAbstractControl \* *newParent*, PropertyEditorModel \* *model*, dtDAL::PropertyContainer \* *newPC*, dtDAL::ActorProperty \* *property*) [virtual]

See also **DynamicAbstractControl::InitializeData** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.22.3.11 void installEventFilterOnControl (QObject \* *filterObj*) [virtual]

See also **DynamicAbstractControl::installEventFilterOnControl**

#### 4.22.3.12 bool isEditable () [virtual]

See also **DynamicAbstractControl::isEditable** (p. 26)

Reimplemented from **DynamicAbstractControl** (p. 26).

**4.22.3.13 bool updateData (QWidget \* *widget*) [virtual, slot]**

Called when we should take the data out of the controls and put it into the actor. This is typically trapped to a lost focus or return pressed or similar user event behavior. This can also be called by the parent control at the moment we change selection. Returns Returns true if any data was actually changed and sucessfully set on the control - This is purely virtual

Implements **DynamicAbstractControl** (p. 27).

**4.22.3.14 void updateEditorFromModel (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateEditorFromModel** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

**4.22.3.15 bool updateModelFromEditor (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateModelFromEditor** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

**4.22.3.16 void useCurrentPressed () [slot]**

The user pressed the 'Use Current' Button. Grab the current resource.

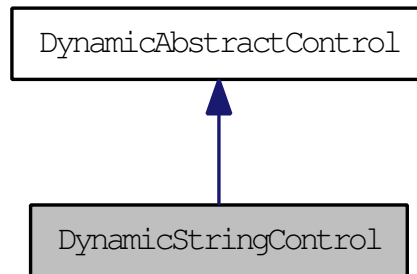
The documentation for this class was generated from the following files:

- **dynamicresourcecontrolbase.h**
- **dynamicresourcecontrolbase.cpp**

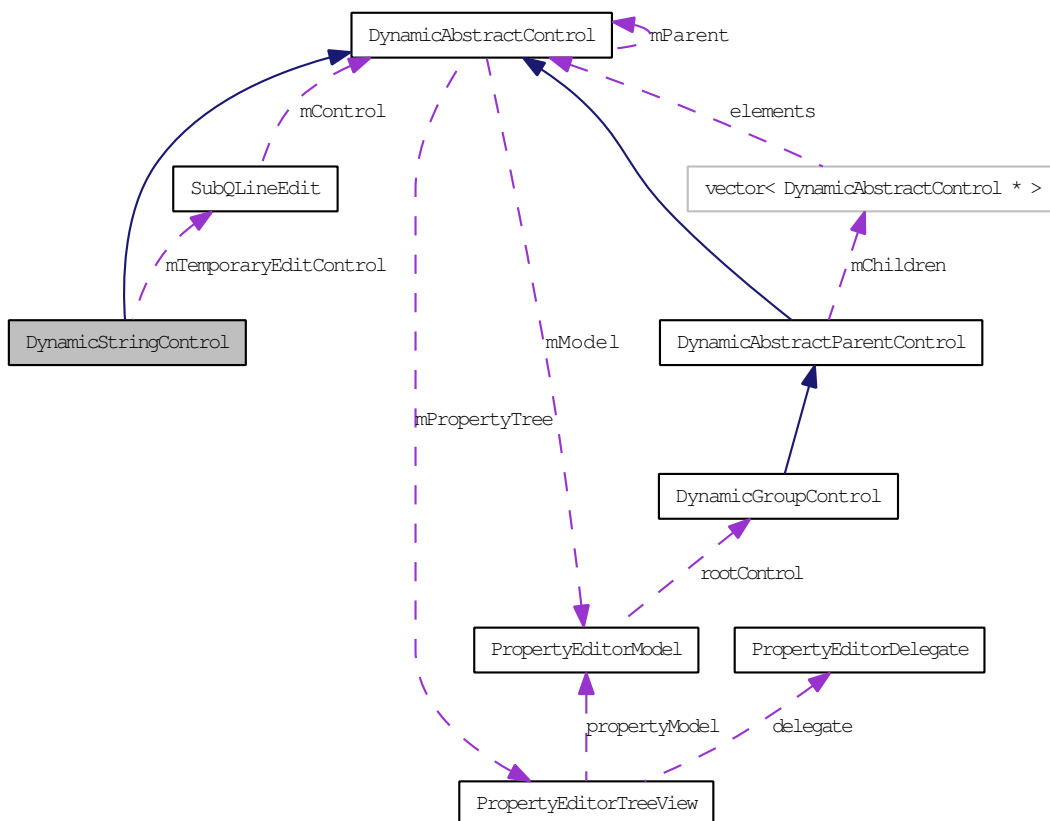
## 4.23 DynamicStringControl Class Reference

This is the dynamic control for the String data type - used in the property editor.

#include <inc/dtQt/dynamicstringcontrol.h> Inheritance diagram for DynamicStringControl:



Collaboration diagram for DynamicStringControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual bool **updateData** (QWidget \*widget)

### Public Member Functions

- **DynamicStringControl** ()

*Constructor.*

- virtual **~DynamicStringControl** ()

*Destructor.*

- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- virtual const QString **getValueAsString** ()
- virtual void **initializeData** (DynamicAbstractControl \*newParent, PropertyEditorModel \*model, dtDAL::PropertyContainer \*newPC, dtDAL::ActorProperty \*property)
- virtual bool **isEditable** ()
- virtual void **updateEditorFromModel** (QWidget \*widget)
- virtual bool **updateModelFromEditor** (QWidget \*widget)

### 4.23.1 Detailed Description

This is the dynamic control for the String data type - used in the property editor.

### 4.23.2 Constructor & Destructor Documentation

#### 4.23.2.1 DynamicStringControl ()

Constructor.

#### 4.23.2.2 ~DynamicStringControl () [virtual]

Destructor.

### 4.23.3 Member Function Documentation

#### 4.23.3.1 void actorPropertyChanged (dtDAL::PropertyContainer & *propCon*, dtDAL::ActorProperty & *property*) [virtual, slot]

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.23.3.2 QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]

See also **DynamicAbstractControl::createEditor** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.23.3.3 const QString getDescription () [virtual]

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.23.3.4 const QString getDisplayName () [virtual]

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.23.3.5 const QString getValueAsString () [virtual]

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

**4.23.3.6** `void handleSubEditDestroy (QWidget * widget, QAbstractItemDelegate::EndEditHint hint = QAbstractItemDelegate::NoHint) [inline, virtual, slot]`

See also `DynamicAbstractControl::handleSubEditDestroy` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.23.3.7** `void InitializeData (DynamicAbstractControl * newParent, PropertyEditorModel * model, dtDAL::PropertyContainer * newPC, dtDAL::ActorProperty * property) [virtual]`

See also `DynamicAbstractControl::InitializeData` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.23.3.8** `bool isEditable () [virtual]`

See also `DynamicAbstractControl::isEditable` (p. 26)

Reimplemented from `DynamicAbstractControl` (p. 26).

**4.23.3.9** `bool updateData (QWidget * widget) [virtual, slot]`

See also `DynamicAbstractControl::updateData` (p. 27)

Implements `DynamicAbstractControl` (p. 27).

**4.23.3.10** `void updateEditorFromModel (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateEditorFromModel` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

**4.23.3.11** `bool updateModelFromEditor (QWidget * widget) [virtual]`

See also `DynamicAbstractControl::updateModelFromEditor` (p. 27)

Reimplemented from `DynamicAbstractControl` (p. 27).

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

- `dynamicstringcontrol.h`
- `dynamicstringcontrol.cpp`



## Protected Member Functions

- **DynamicVectorElementControl \* CreateElementControl** (PropertyType \*prop, int index, const std::string &label, PropertyEditorModel \*newModel, dtDAL::PropertyContainer \*newPC)

### 4.24.1 Detailed Description

`template<typename PropertyType> class dtQt::DynamicVecNControl< PropertyType >`

This is the dynamic control for the N dimensional vector data type - used in the property editor It adds a group of child elements to the tree, since you can't edit multiple things in one control easily.

### 4.24.2 Member Typedef Documentation

**4.24.2.1** `typedef PropertyType::GetValueType PropertyGetValueType`

### 4.24.3 Constructor & Destructor Documentation

**4.24.3.1** `DynamicVecNControl () [inline]`

Constructor.

**4.24.3.2** `~DynamicVecNControl () [inline]`

Destructor.

### 4.24.4 Member Function Documentation

**4.24.4.1** `DynamicVectorElementControl * CreateElementControl (PropertyType * prop, int index, const std::string & label, PropertyEditorModel * newModel, dtDAL::PropertyContainer * newPC) [inline, protected]`

**4.24.4.2** `const QString getDescription () [inline, virtual]`

See also `DynamicAbstractControl::getDescription` (p. 24)

Reimplemented from `DynamicAbstractControl` (p. 24).

**4.24.4.3** `const QString getDisplayName () [inline, virtual]`

See also `DynamicAbstractControl::getDisplayName` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.24.4.4** `const QString getValueAsString () [inline, virtual]`

See also `DynamicAbstractControl::getValueAsString` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.24.4.5** `void InitializeData (DynamicAbstractControl * newParent, PropertyEditorModel * model, dtDAL::PropertyContainer * newPC, dtDAL::ActorProperty * property) [inline, virtual]`

See also `DynamicAbstractControl::InitializeData` (p. 25)

Reimplemented from `DynamicAbstractControl` (p. 25).

**4.24.4.6** `bool isEditable () [inline, virtual]`

See also `DynamicAbstractControl::isEditable` (p. 26)

Reimplemented from `DynamicAbstractControl` (p. 26).

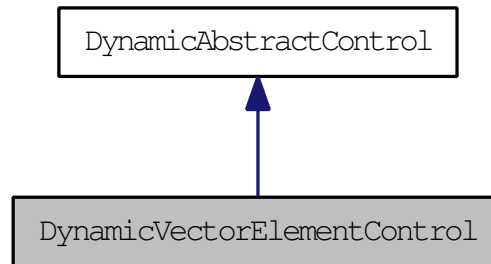
The documentation for this class was generated from the following file:

- `dynamicvecncontrol.h`

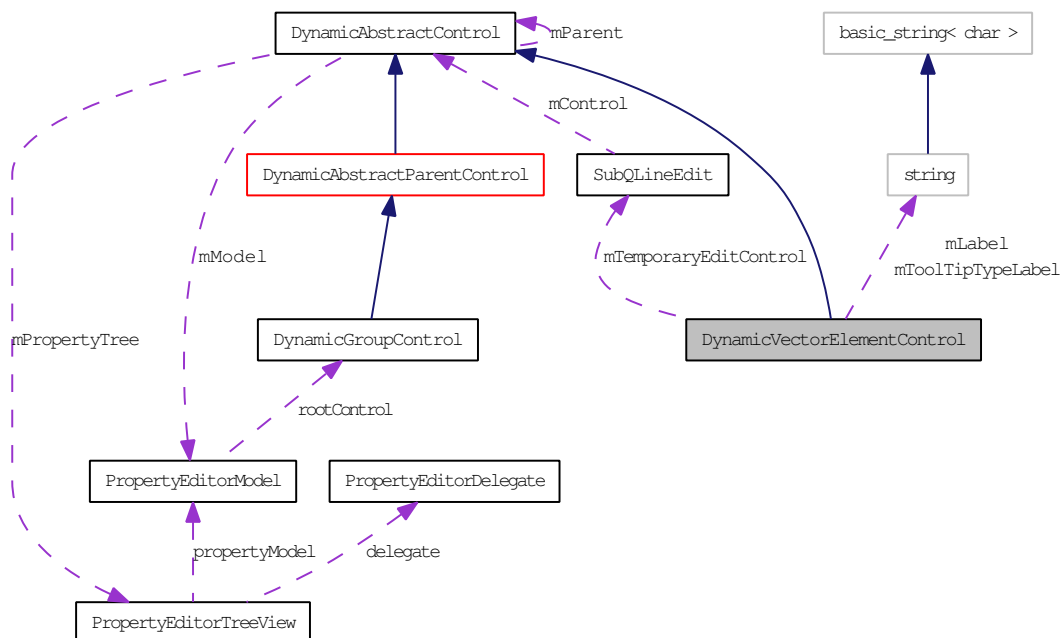
## 4.25 DynamicVectorElementControl Class Reference

This is the a sub control used by the various vector property classes.

#include <inc/dtQt/dynamicvectorelementcontrol.h> Inheritance diagram for DynamicVectorElementControl:



Collaboration diagram for DynamicVectorElementControl:



### Public Slots

- void **actorPropertyChanged** (dtDAL::PropertyContainer &propCon, dtDAL::ActorProperty &property)  
*When a property changes, we have to update our editor.*
- void **handleSubEditDestroy** (QWidget \*widget, QAbstractItemDelegate::EndEditHint hint=QAbstractItemDelegate::NoHint)
- virtual bool **updateData** (QWidget \*widget)  
*Called when we should take the data out of the controls and put it into the actor.*

### Public Member Functions

- **DynamicVectorElementControl** (dtDAL::Vec4dActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec4d property.*

- **DynamicVectorElementControl** (dtDAL::Vec4fActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec4f property.*
- **DynamicVectorElementControl** (dtDAL::Vec4ActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec4 property.*
- **DynamicVectorElementControl** (dtDAL::Vec3dActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec3d property.*
- **DynamicVectorElementControl** (dtDAL::Vec3fActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec3f property.*
- **DynamicVectorElementControl** (dtDAL::Vec3ActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec3 property.*
- **DynamicVectorElementControl** (dtDAL::Vec2dActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec2d property.*
- **DynamicVectorElementControl** (dtDAL::Vec2fActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec2f property.*
- **DynamicVectorElementControl** (dtDAL::Vec2ActorProperty \*newVectorProp, int whichIndex, const std::string &newLabel)  
*Constructor - For the Vec2 property.*
- virtual ~**DynamicVectorElementControl** ()  
*Destructor.*
- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index)
- virtual const QString **getDescription** ()
- virtual const QString **getDisplayName** ()
- double **getValue** ()  
*A convenience method to get the value from the associated vector.*
- virtual const QString **getValueAsString** ()
- virtual void **initializeData** (**DynamicAbstractControl** \*newParent, **PropertyEditorModel** \*model, dtDAL::PropertyContainer \*pc, dtDAL::ActorProperty \*property)
- virtual bool **isEditable** ()
- void **setValue** (double value)  
*Puts the passed in value into the appropriate vector at whichElement index.*
- virtual void **updateEditorFromModel** (QWidget \*widget)
- virtual bool **updateModelFromEditor** (QWidget \*widget)

#### 4.25.1 Detailed Description

This is the a sub control used by the various vector property classes. In order to draw a vector 3 in the property tree, you actually have an X, Y, and Z entry. This class is for that. And, this class actually supports the Vec2, Vec3, and Vec4 with separate constructors. It provides a get and set method to get at the data.

## 4.25.2 Constructor & Destructor Documentation

### 4.25.2.1 `DynamicVectorElementControl (dtDAL::Vec2ActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec2 property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.2 `DynamicVectorElementControl (dtDAL::Vec2fActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec2f property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.3 `DynamicVectorElementControl (dtDAL::Vec2dActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec2d property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.4 `DynamicVectorElementControl (dtDAL::Vec3ActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec3 property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.5 `DynamicVectorElementControl (dtDAL::Vec3fActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec3f property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.6 `DynamicVectorElementControl (dtDAL::Vec3dActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec3d property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.7 `DynamicVectorElementControl (dtDAL::Vec4ActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec4 property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.8 `DynamicVectorElementControl (dtDAL::Vec4fActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec4f property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.9 `DynamicVectorElementControl (dtDAL::Vec4dActorProperty * newVectorProp, int whichIndex, const std::string & newLabel)`

Constructor - For the Vec4d property. - We can put data in the constructor because aren't using the factory for this.

### 4.25.2.10 `~DynamicVectorElementControl () [virtual]`

Destructor.

## 4.25.3 Member Function Documentation

### 4.25.3.1 `void actorPropertyChanged (dtDAL::PropertyContainer & propCon, dtDAL::ActorProperty & property) [virtual, slot]`

When a property changes, we have to update our editor. It is likely that many properties will change with no effect, but if the user is using undo/redo or is moving an actor in the viewport, then it is possible that they will also be sitting on the editor for one of the affected values. This gives us a chance to reflect the change in our editor. The default implementation does nothing.

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.25.3.2 **QWidget \* createEditor (QWidget \* *parent*, const QStyleOptionViewItem & *option*, const QModelIndex & *index*) [virtual]**

See also **DynamicAbstractControl::createEditor** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.25.3.3 **const QString getDescription () [virtual]**

See also **DynamicAbstractControl::getDescription** (p. 24)

Reimplemented from **DynamicAbstractControl** (p. 24).

#### 4.25.3.4 **const QString getDisplayName () [virtual]**

See also **DynamicAbstractControl::getDisplayName** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.25.3.5 **double getValue ()**

A convenience method to get the value from the associated vector. Which element is the 0 based index into the vector.

#### 4.25.3.6 **const QString getValueAsString () [virtual]**

See also **DynamicAbstractControl::getValueAsString** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.25.3.7 **void handleSubEditDestroy (QWidget \* *widget*, QAbstractItemDelegate::EndEditHint *hint* = QAbstractItemDelegate::NoHint) [inline, virtual, slot]**

See also **DynamicAbstractControl::handleSubEditDestroy** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.25.3.8 **void InitializeData (DynamicAbstractControl \* *newParent*, PropertyEditorModel \* *model*, dtDAL::PropertyContainer \* *pc*, dtDAL::ActorProperty \* *property*) [virtual]**

See also **DynamicAbstractControl::InitializeData** (p. 25)

Reimplemented from **DynamicAbstractControl** (p. 25).

#### 4.25.3.9 **bool isEditable () [virtual]**

See also **DynamicAbstractControl::isEditable** (p. 26)

Reimplemented from **DynamicAbstractControl** (p. 26).

#### 4.25.3.10 **void setValue (double *value*)**

Puts the passed in value into the appropriate vector at whichElement index.

#### 4.25.3.11 **bool updateData (QWidget \* *widget*) [virtual, slot]**

Called when we should take the data out of the controls and put it into the actor. This is typically trapped to a lost focus or return pressed or similar user event behavior. This can also be called by the parent control at the moment we change selection. Returns Returns true if any data was actually changed and successfully set on the control - This is purely virtual

Implements **DynamicAbstractControl** (p. 27).

#### 4.25.3.12 **void updateEditorFromModel (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateEditorFromModel** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

**4.25.3.13 bool updateModelFromEditor (QWidget \* *widget*) [virtual]**

See also **DynamicAbstractControl::updateModelFromEditor** (p. 27)

Reimplemented from **DynamicAbstractControl** (p. 27).

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

- **dynamicvectorelementcontrol.h**
- **dynamicvectorelementcontrol.cpp**

## 4.26 GLWidgetFactory Class Reference

Abstract factory interface used to create specialized **OSGAdapterWidget** (p. 94) for custom application requirements.

```
#include <inc/dtQt/glwidgetfactory.h>
```

### Public Member Functions

- **GLWidgetFactory** ()
- virtual **OSGAdapterWidget \* CreateWidget** (bool drawOnSeparateThread, QWidget \*parent=NULL, const QGLWidget \*shareWidget=NULL, Qt::WindowFlags f=NULL)=0  
*Overwrite to generate a custom **OSGAdapterWidget** (p. 94).*

### Protected Member Functions

- **~GLWidgetFactory** ()

#### 4.26.1 Detailed Description

Abstract factory interface used to create specialized **OSGAdapterWidget** (p. 94) for custom application requirements. See also **QtGuiWindowSystemWrapper::SetGLWidgetFactory()** (p. 107)

#### 4.26.2 Constructor & Destructor Documentation

4.26.2.1 **GLWidgetFactory** () [inline]

4.26.2.2 **~GLWidgetFactory** () [inline, protected]

#### 4.26.3 Member Function Documentation

4.26.3.1 virtual **OSGAdapterWidget\* CreateWidget** (bool *drawOnSeparateThread*, **QWidget \* parent** = NULL, const **QGLWidget \* shareWidget** = NULL, **Qt::WindowFlags f** = NULL) [pure virtual]

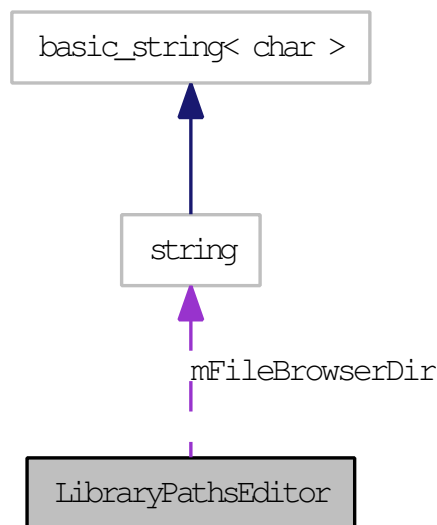
Overwrite to generate a custom **OSGAdapterWidget** (p. 94).

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

- **glwidgetfactory.h**

## 4.27 LibraryPathsEditor Class Reference

#include <inc/dtQt/librarypathseditor.h> Collaboration diagram for LibraryPathsEditor:



### Public Slots

- void **RefreshButtons** ()  
*Received when the path selection state changes.*
- void **ShiftPathDown** ()  
*Shift the current path down 1 position.*
- void **ShiftPathUp** ()  
*Shift the current path up 1 position.*
- void **SpawnDeleteConfirmation** ()  
*Confirm deletion of libraries.*
- void **SpawnFileBrowser** ()  
*Pop up the file browser for libraries.*

### Public Member Functions

- **LibraryPathsEditor** (QWidget \*parent, const std::string &fileBrowserStartDir)  
*Constructor.*
- virtual **~LibraryPathsEditor** ()  
*Destructor.*
- bool **AnyItemsSelected** () const  
*Are any items selected?*

#### 4.27.1 Constructor & Destructor Documentation

##### 4.27.1.1 LibraryPathsEditor (QWidget \* parent, const std::string & fileBrowserStartDir)

Constructor.

**4.27.1.2** `~LibraryPathsEditor () [virtual]`

Destructor.

**4.27.2 Member Function Documentation****4.27.2.1** `bool AnyItemsSelected () const`

Are any items selected?

**4.27.2.2** `void RefreshButtons () [slot]`

Received when the path selection state changes.

**4.27.2.3** `void ShiftPathDown () [slot]`

Shift the current path down 1 position.

**4.27.2.4** `void ShiftPathUp () [slot]`

Shift the current path up 1 position.

**4.27.2.5** `void SpawnDeleteConfirmation () [slot]`

Confirm deletion of libraries.

**4.27.2.6** `void SpawnFileBrowser () [slot]`

Pop up the file browser for libraries.

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

- **librarypathseditor.h**
- **librarypathseditor.cpp**

## 4.28 OSGAdapterWidget Class Reference

```
#include <inc/dtQt/osgadapterwidget.h>
```

### Public Slots

- void **ThreadedUpdateGL** ()

### Public Member Functions

- **OSGAdapterWidget** (bool drawOnSeparateThread, QWidget \*parent=NULL, const QGLWidget \*shareWidget=NULL, Qt::WindowFlags f=NULL)
- virtual ~**OSGAdapterWidget** ()
- const osgViewer::GraphicsWindow & **GetGraphicsWindow** () const
- osgViewer::GraphicsWindow & **GetGraphicsWindow** ()
- void **SetGraphicsWindow** (osgViewer::GraphicsWindow &newWindow)
- void **ThreadedInitializeGL** ()
- void **ThreadedMakeCurrent** ()

### Protected Member Functions

- virtual void **initializeGL** ()
- virtual void **keyPressEvent** (QKeyEvent \*event)
- virtual void **keyReleaseEvent** (QKeyEvent \*event)
- virtual void **mouseMoveEvent** (QMouseEvent \*event)
- virtual void **mousePressEvent** (QMouseEvent \*event)
- virtual void **mouseReleaseEvent** (QMouseEvent \*event)
- virtual void **paintGL** ()
- void **paintGLImpl** ()
- virtual void **resizeGL** (int width, int height)
- void **resizeGLImpl** (int width, int height)
- virtual void **wheelEvent** (QWheelEvent \*event)

### Protected Attributes

- volatile bool **mDoResize**
- bool **mDrawOnSeparateThread**
- dtCore::RefPtr< osgViewer::GraphicsWindow > **mGraphicsWindow**
- QGLContext \* **mThreadGLContext**
- QTimer **mTimer**

### 4.28.1 Constructor & Destructor Documentation

4.28.1.1 **OSGAdapterWidget** (bool *drawOnSeparateThread*, QWidget \* *parent* = NULL, const QGLWidget \* *shareWidget* = NULL, Qt::WindowFlags *f* = NULL)

4.28.1.2 **~OSGAdapterWidget** () [virtual]

### 4.28.2 Member Function Documentation

4.28.2.1 **const osgViewer::GraphicsWindow & GetGraphicsWindow** () const

4.28.2.2 **osgViewer::GraphicsWindow & GetGraphicsWindow** ()

4.28.2.3 **void initializeGL** () [protected, virtual]

4.28.2.4 **void keyPressEvent** (QKeyEvent \* *event*) [protected, virtual]

4.28.2.5 **void keyReleaseEvent** (QKeyEvent \* *event*) [protected, virtual]

4.28.2.6 **void mouseMoveEvent** (QMouseEvent \* *event*) [protected, virtual]

4.28.2.7 **void mousePressEvent** (QMouseEvent \* *event*) [protected, virtual]

4.28.2.8 **void mouseReleaseEvent** (QMouseEvent \* *event*) [protected, virtual]

4.28.2.9 **void paintGL** () [protected, virtual]

4.28.2.10 **void paintGLImpl** () [protected]

4.28.2.11 **void resizeGL** (int *width*, int *height*) [protected, virtual]

set message to resize. this is actually a race condition. It needs to be locked.

4.28.2.12 **void resizeGLImpl** (int *width*, int *height*) [protected]

4.28.2.13 **void SetGraphicsWindow** (osgViewer::GraphicsWindow & *newWindow*)

4.28.2.14 **void ThreadedInitializeGL** ()

4.28.2.15 **void ThreadedMakeCurrent** ()

4.28.2.16 **void ThreadedUpdateGL** () [slot]

4.28.2.17 **void wheelEvent** (QWheelEvent \* *event*) [protected, virtual]

### 4.28.3 Member Data Documentation

4.28.3.1 **volatile bool mDoResize** [protected]

4.28.3.2 **bool mDrawOnSeparateThread** [protected]

4.28.3.3 **dtCore::RefPtr<osgViewer::GraphicsWindow> mGraphicsWindow** [protected]

4.28.3.4 **QGLContext\* mThreadGLContext** [protected]

4.28.3.5 **QTimer mTimer** [protected]

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

- **osgadapterwidget.h**
- **osgadapterwidget.cpp**

## 4.29 OSGGraphicsWindowQt Class Reference

```
#include <inc/dtQt/osggraphicswindowqt.h>
```

### Public Types

- typedef osgViewer::GraphicsWindow **BaseClass**

### Public Member Functions

- **OSGGraphicsWindowQt** (osg::GraphicsContext::Traits \*traits, dtQt::GLWidgetFactory \*factory=NULL, dtQt::OSGAdapterWidget \*adapter=NULL)
- virtual ~**OSGGraphicsWindowQt** ()
- virtual void **checkEvents** ()  
*Check to see if any events have been generated.*
- virtual const char \* **className** () const
- virtual void **closeImplementation** ()  
*Close the graphics context.*
- const QGLWidget \* **GetQGLWidget** () const
- QGLWidget \* **GetQGLWidget** ()
- virtual void **getWindowRectangle** (int &x, int &y, int &width, int &height)  
*Get the window's position and size.*
- virtual void **grabFocus** ()  
*Get focus.*
- virtual void **grabFocusIfPointerInWindow** ()  
*Get focus on if the pointer is in this window.*
- virtual bool **isRealizedImplementation** () const  
*Return true if the graphics context has been realised and is ready to use.*
- virtual bool **isSameKindAs** (const Object \*object) const
- virtual const char \* **libraryName** () const
- virtual bool **makeCurrentImplementation** ()  
*Make this graphics context current.*
- virtual bool **realizeImplementation** ()  
*Realise the GraphicsContext.*
- virtual bool **releaseContextImplementation** ()  
*Release the graphics context.*
- void **requestClose** ()
- virtual void **resizedImplementation** (int x, int y, int width, int height)
- virtual void **setCursor** (osgViewer::GraphicsWindow::MouseCursor mouseCursor)
- void **SetQGLWidget** (QGLWidget \*qwidget)
- virtual bool **setWindowDecorationImplementation** (bool flag)  
*Set Window decoration.*
- virtual void **setWindowName** (const std::string &name)
- virtual bool **setWindowRectangleImplementation** (int x, int y, int width, int height)  
*Set the window's position and size.*
- virtual void **swapBuffersImplementation** ()

Swap the front and back buffers.

- virtual void **useCursor** (bool cursorOn)
- virtual bool **valid** () const

## 4.29.1 Member Typedef Documentation

### 4.29.1.1 typedef osgViewer::GraphicsWindow BaseClass

## 4.29.2 Constructor & Destructor Documentation

### 4.29.2.1 OSGGraphicsWindowQt (osg::GraphicsContext::Traits \* *traits*, dtQt::GLWidgetFactory \* *factory* = NULL, dtQt::OSGAdapterWidget \* *adapter* = NULL)

### 4.29.2.2 ~OSGGraphicsWindowQt () [virtual]

## 4.29.3 Member Function Documentation

### 4.29.3.1 void checkEvents () [virtual]

Check to see if any events have been generated.

### 4.29.3.2 const char \* className () const [virtual]

### 4.29.3.3 void closeImplementation () [virtual]

Close the graphics context.

### 4.29.3.4 const QGLWidget \* GetQGLWidget () const

### 4.29.3.5 QGLWidget \* GetQGLWidget ()

### 4.29.3.6 void getWindowRectangle (int & *x*, int & *y*, int & *width*, int & *height*) [virtual]

Get the window's position and size.

### 4.29.3.7 void grabFocus () [virtual]

Get focus.

### 4.29.3.8 void grabFocusIfPointerInWindow () [virtual]

Get focus on if the pointer is in this window.

### 4.29.3.9 bool isRealizedImplementation () const [virtual]

Return true if the graphics context has been realised and is ready to use.

### 4.29.3.10 bool isSameKindAs (const Object \* *object*) const [virtual]

### 4.29.3.11 const char \* libraryName () const [virtual]

### 4.29.3.12 bool makeCurrentImplementation () [virtual]

Make this graphics context current.

### 4.29.3.13 bool realizeImplementation () [virtual]

Realise the GraphicsContext.

### 4.29.3.14 bool releaseContextImplementation () [virtual]

Release the graphics context.

### 4.29.3.15 void requestClose ()

### 4.29.3.16 void resizedImplementation (int *x*, int *y*, int *width*, int *height*) [virtual]

### 4.29.3.17 void setCursor (osgViewer::GraphicsWindow::MouseCursor *mouseCursor*) [virtual]

### 4.29.3.18 void SetQGLWidget (QGLWidget \* *qwidget*)

### 4.29.3.19 bool setWindowDecorationImplementation (bool *flag*) [virtual]

Set Window decoration.

**4.29.3.20 void setWindowName (const std::string & *name*) [virtual]**

**4.29.3.21 bool setWindowRectangleImplementation (int *x*, int *y*, int *width*, int *height*) [virtual]**

Set the window's position and size.

**4.29.3.22 void swapBuffersImplementation () [virtual]**

Swap the front and back buffers.

**4.29.3.23 void useCursor (bool *cursorOn*) [virtual]**

**4.29.3.24 bool valid () const [virtual]**

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

- **osggraphicswindowqt.h**
- **osggraphicswindowqt.cpp**

## 4.30 ProjectContextDialog Class Reference

```
#include <inc/dtQt/projectcontextdialog.h>
```

### Public Slots

- void **spawnFileBrowser** ()

### Public Member Functions

- **ProjectContextDialog** (QWidget \*parent=NULL)  
*Constructor.*
- virtual **~ProjectContextDialog** ()  
*Destructor.*
- QString **getProjectPath** () const  
*Returns the path to the new project specified by the user.*

### 4.30.1 Constructor & Destructor Documentation

#### 4.30.1.1 ProjectContextDialog (QWidget \* *parent* = NULL)

Constructor.

#### 4.30.1.2 ~ProjectContextDialog () [virtual]

Destructor.

### 4.30.2 Member Function Documentation

#### 4.30.2.1 QString getProjectPath () const

Returns the path to the new project specified by the user. Returns The directory path.

#### 4.30.2.2 void spawnFileBrowser () [slot]

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

- **projectcontextdialog.h**
- **projectcontextdialog.cpp**

## 4.31 PropertyEditorDelegate Class Reference

This class is a delegate for editing commands between the model and the dynamic control.

```
#include <inc/dtQt/propertyeditordelegate.h>
```

### Public Member Functions

- **PropertyEditorDelegate** (QObject \*parent=0)

*Constructor.*

- virtual **~PropertyEditorDelegate** ()

*Destructor.*

- virtual QWidget \* **createEditor** (QWidget \*parent, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual bool **eventFilter** (QObject \*object, QEvent \*event)
- virtual void **paint** (QPainter \*painter, const QStyleOptionViewItem &option, const QModelIndex &index) const
- virtual void **setEditorData** (QWidget \*editor, const QModelIndex &index) const
- virtual void **setModelData** (QWidget \*editor, QAbstractItemModel \*model, const QModelIndex &index) const
- virtual QSize **sizeHint** (const QStyleOptionViewItem &option, const QModelIndex &index) const

### Protected Member Functions

- virtual void **drawDecoration** (QPainter \*painter, const QStyleOptionViewItem &option, const QRect &rect, const QPixmap &pixmap) const

#### 4.31.1 Detailed Description

This class is a delegate for editing commands between the model and the dynamic control. Essentially, this is the 'control' part of the MVC architecture, but QT intentionally doesn't call it that.

The base idea for this class was taken from qpropertyeditor\_delegate.h from the QT Beta2 Designer application.

Goto: `QT\tools\designer\src\components\propertyeditor\qpropertyeditor_delegate.h` to see the original. The original class is distributable according to the standard QT license (GPL).

#### 4.31.2 Constructor & Destructor Documentation

##### 4.31.2.1 PropertyEditorDelegate (QObject \* parent = 0)

Constructor.

##### 4.31.2.2 ~PropertyEditorDelegate () [virtual]

Destructor.

### 4.31.3 Member Function Documentation

- 4.31.3.1 `QWidget * createEditor (QWidget * parent, const QStyleOptionViewItem & option, const QModelIndex & index) const` [virtual]
- 4.31.3.2 `void drawDecoration (QPainter * painter, const QStyleOptionViewItem & option, const QRect & rect, const QPixmap & pixmap) const` [protected, virtual]
- 4.31.3.3 `bool eventFilter (QObject * object, QEvent * event)` [virtual]
- 4.31.3.4 `void paint (QPainter * painter, const QStyleOptionViewItem & option, const QModelIndex & index) const` [virtual]
- 4.31.3.5 `void setEditorData (QWidget * editor, const QModelIndex & index) const` [virtual]
- 4.31.3.6 `void setModelData (QWidget * editor, QAbstractItemModel * model, const QModelIndex & index) const` [virtual]
- 4.31.3.7 `QSize sizeHint (const QStyleOptionViewItem & option, const QModelIndex & index) const` [virtual]

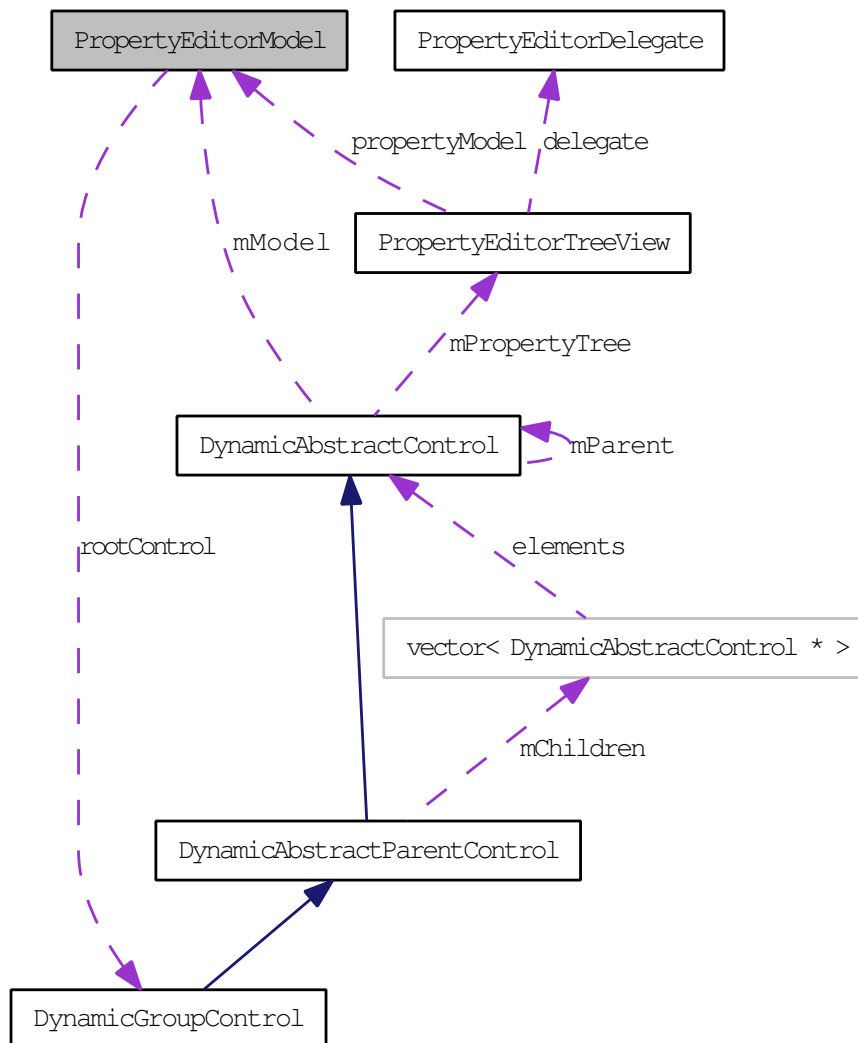
The documentation for this class was generated from the following files:

- `propertyeditordelegate.h`
- `propertyeditordelegate.cpp`

## 4.32 PropertyEditorModel Class Reference

This class is the model used to represent the properties for a selected object.

#include <inc/dtQt/propertyeditormodel.h> Collaboration diagram for PropertyEditorModel:



### Public Member Functions

- **PropertyEditorModel** (QObject \*parent=0)  
*Constructor.*
- **~PropertyEditorModel** ()  
*Destructor.*
- virtual int **columnCount** (const QModelIndex &parent) const
- virtual QVariant **data** (const QModelIndex &index, int role) const
- virtual Qt::ItemFlags **flags** (const QModelIndex &index) const
- **DynamicAbstractControl \* GetAbstractControlFromIndex** (const QModelIndex &index) const  
*Convenience method - This returns the **DynamicAbstractControl** (p. 21) from the QModelIndex.*
- **DynamicGroupControl \* GetRootControl** () const  
*Remove the children of the root group object.*
- virtual bool **hasChildren** (const QModelIndex &parent) const

- virtual QVariant **headerData** (int section, Qt::Orientation orientation, int role) const
- virtual QModelIndex **index** (int row, int column, const QModelIndex &parent) const
- QModelIndex **IndexOf** (**DynamicAbstractControl** \*property, int column=0) const

*This class returns the QModelIndex for the property.*

- virtual bool **insertRows** (int row, int count, const QModelIndex &parent=QModelIndex())
- virtual bool **isEditable** (const QModelIndex &index) const
- virtual QModelIndex **parent** (const QModelIndex &index) const
- virtual bool **removeRows** (int position, int rows, const QModelIndex &parent)
- virtual int **rowCount** (const QModelIndex &parent) const
- virtual bool **setData** (const QModelIndex &index, const QVariant &value, int role)
- virtual void **setDescription** (**DynamicAbstractControl** \*control)
- void **SetRootControl** (**DynamicGroupControl** \*newRoot)

## Friends

- class **DynamicAbstractParentControl**
- class **DynamicGroupControl**
- class **PropertyEditor**

### 4.32.1 Detailed Description

This class is the model used to represent the properties for a selected object. This class is fairly complex. It follows the model/view architecture for QT. The base idea for this class was taken from qpropertyeditor\_model.h from the QT Beta2 Designer application.

Goto: QT\tools\designer\src\components\propertyeditor\qpropertyeditor\_model.h to see the original. The original class is distributable according to the standard QT license (GPL).

### 4.32.2 Constructor & Destructor Documentation

#### 4.32.2.1 PropertyEditorModel (QObject \* parent = 0)

Constructor.

#### 4.32.2.2 ~PropertyEditorModel ()

Destructor.

### 4.32.3 Member Function Documentation

#### 4.32.3.1 int columnCount (const QModelIndex & parent) const [virtual]

#### 4.32.3.2 QVariant data (const QModelIndex & index, int role) const [virtual]

#### 4.32.3.3 Qt::ItemFlags flags (const QModelIndex & index) const [virtual]

#### 4.32.3.4 DynamicAbstractControl \* GetAbstractControlFromIndex (const QModelIndex & index) const

Convenience method - This returns the **DynamicAbstractControl** (p. 21) from the QModelIndex. It's kinda wonky, but QT allows you to construct a QModelIndex with a void \* pointer but doesn't allow you to access the data as a void \* later. Instead, we have to static cast it out from the QVariant in **data()** (p. 103).

#### 4.32.3.5 DynamicGroupControl \* GetRootControl () const

Remove the children of the root group object. This is necessary because the model needs to tell the underlying QAbstractItemModel to update itself. If we don't remove items safely, then the base class will eventually try to access an object that doesn't exist.

4.32.3.6 virtual bool hasChildren (const QModelIndex & *parent*) const [inline, virtual]

4.32.3.7 QVariant headerData (int *section*, Qt::Orientation *orientation*, int *role*) const [virtual]

4.32.3.8 QModelIndex index (int *row*, int *column*, const QModelIndex & *parent*) const [virtual]

4.32.3.9 QModelIndex IndexOf (DynamicAbstractControl \* *property*, int *column* = 0) const

This class returns the QModelIndex for the property. The index comes from getting the parent of the property and then finding this properties row in the parent.

4.32.3.10 bool insertRows (int *row*, int *count*, const QModelIndex & *parent* = QModelIndex()) [virtual]

4.32.3.11 bool isEditable (const QModelIndex & *index*) const [virtual]

4.32.3.12 QModelIndex parent (const QModelIndex & *index*) const [virtual]

4.32.3.13 bool removeRows (int *position*, int *rows*, const QModelIndex & *parent*) [virtual]

4.32.3.14 int rowCount (const QModelIndex & *parent*) const [virtual]

4.32.3.15 virtual bool setData (const QModelIndex & *index*, const QVariant & *value*, int *role*) [inline, virtual]

4.32.3.16 void setDescription (DynamicAbstractControl \* *control*) [virtual]

4.32.3.17 void SetRootControl (DynamicGroupControl \* *newRoot*)

#### 4.32.4 Friends And Related Function Documentation

4.32.4.1 friend class DynamicAbstractParentControl [friend]

4.32.4.2 friend class DynamicGroupControl [friend]

4.32.4.3 friend class PropertyEditor [friend]

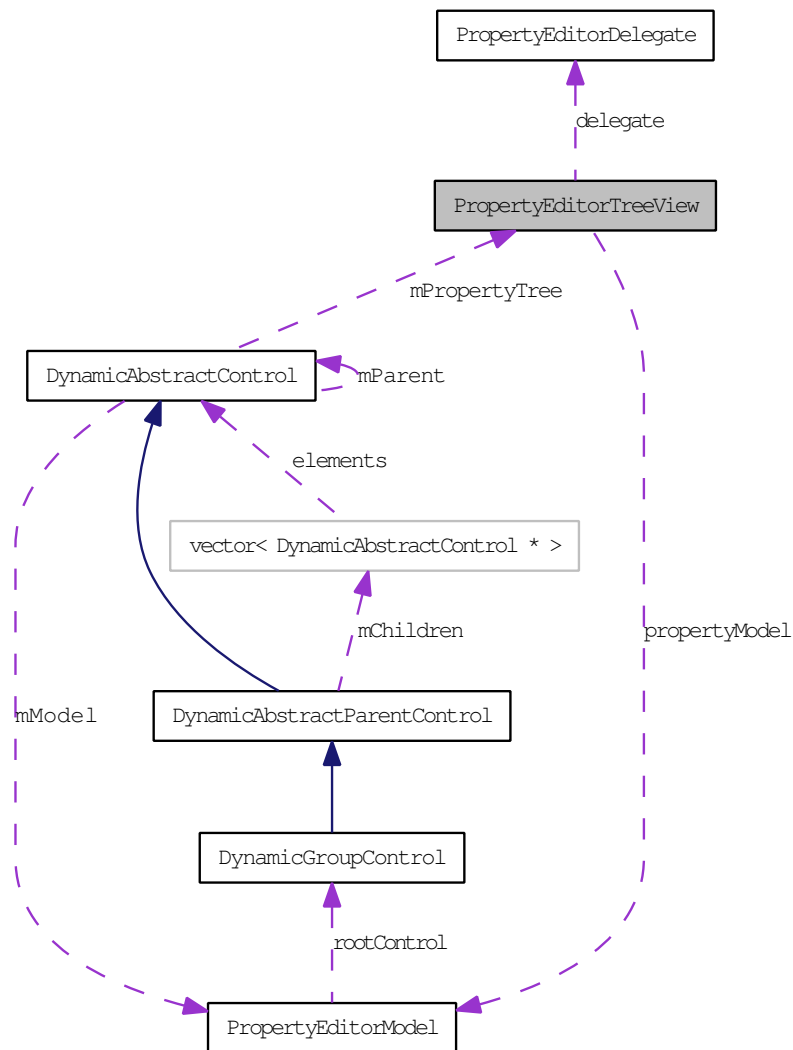
The documentation for this class was generated from the following files:

- propertyeditormodel.h
- propertyeditormodel.cpp

## 4.33 PropertyEditorTreeView Class Reference

This class is the tree control for the properties of a proxy.

#include <inc/dtQt/propertyeditortreeview.h> Collaboration diagram for PropertyEditorTreeView:



### Public Slots

- virtual void `reset ()`

### Public Member Functions

- **PropertyEditorTreeView** (**PropertyEditorModel** \*model, QWidget \*parent=0)  
*Constructor.*
- **~PropertyEditorTreeView** ()  
*Destructor.*
- bool **isReadOnly** () const  
*Is the tree view read only? Yes it is.*
- void **setRoot** (**DynamicGroupControl** \*newRoot)  
*Used to set the root item.*

## Static Public Attributes

- static const QColor **ROW\_COLOR\_EVEN**  
*The row colors are made public constants so that controls that are embedded inside the table can determine what color backgrounds to use.*
- static const QColor **ROW\_COLOR\_ODD**

## Protected Slots

- virtual void **closeEditor** (QWidget \*editor, QAbstractItemDelegate::EndEditHint hint)
- virtual void **currentChanged** (const QModelIndex &current, const QModelIndex &previous)
- virtual void **selectionChanged** (const QItemSelection &selected, const QItemSelection &deselected)

### 4.33.1 Detailed Description

This class is the tree control for the properties of a proxy. It knows how to draw itself based on the underlying data in the **PropertyEditorModel** (p. 102). This class follows the model/view architecture for QT. The base idea for this class was taken from qpropertyeditor.h from the QT Beta2 Designer application.

Goto: QT\tools\designer\src\components\propertyeditor\qpropertyeditor.h to see the original. The original class is distributable according to the standard QT license (GPL).

### 4.33.2 Constructor & Destructor Documentation

#### 4.33.2.1 PropertyEditorTreeView (PropertyEditorModel \* model, QWidget \* parent = 0)

Constructor.

#### 4.33.2.2 ~PropertyEditorTreeView ()

Destructor.

### 4.33.3 Member Function Documentation

#### 4.33.3.1 void closeEditor (QWidget \* editor, QAbstractItemDelegate::EndEditHint hint) [protected, virtual, slot]

#### 4.33.3.2 void currentChanged (const QModelIndex & current, const QModelIndex & previous) [protected, virtual, slot]

#### 4.33.3.3 bool isReadOnly () const [inline]

Is the tree view read only? Yes it is.

#### 4.33.3.4 void reset () [virtual, slot]

#### 4.33.3.5 void selectionChanged (const QItemSelection & selected, const QItemSelection & deselected) [protected, virtual, slot]

#### 4.33.3.6 void setRoot (DynamicGroupControl \* newRoot)

Used to set the root item. This causes several settings to occur.

### 4.33.4 Member Data Documentation

#### 4.33.4.1 const QColor ROW\_COLOR\_EVEN [static]

The row colors are made public constants so that controls that are embedded inside the table can determine what color backgrounds to use.

#### 4.33.4.2 const QColor ROW\_COLOR\_ODD [static]

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

- **propertyeditortreeview.h**
- **propertyeditortreeview.cpp**

## 4.34 QtGuiWindowSystemWrapper Class Reference

```
#include <inc/dtQt/qtguiwindowssystemwrapper.h>
```

### Public Member Functions

- **QtGuiWindowSystemWrapper** (osg::GraphicsContext::WindowingSystemInterface &oldInterface)
- virtual osg::GraphicsContext \* **createGraphicsContext** (osg::GraphicsContext::Traits \*traits)
- virtual unsigned int **getNumScreens** (const osg::GraphicsContext::ScreenIdentifier &screenIdentifier=osg::GraphicsContext::ScreenIdentifier())
- virtual void **getScreenResolution** (const osg::GraphicsContext::ScreenIdentifier &screenIdentifier, unsigned int &width, unsigned int &height)
- void **SetGLWidgetFactory** (**GLWidgetFactory** \*factory)  
*Supply a custom factory to create custom QGLWidgets.*
- virtual bool **setScreenRefreshRate** (const osg::GraphicsContext::ScreenIdentifier &screenIdentifier, double refreshRate)
- virtual bool **setScreenResolution** (const osg::GraphicsContext::ScreenIdentifier &screenIdentifier, unsigned int width, unsigned int height)

### Static Public Member Functions

- static void **EnableQtGUIWrapper** ()

### Protected Member Functions

- virtual ~**QtGuiWindowSystemWrapper** ()

#### 4.34.1 Constructor & Destructor Documentation

4.34.1.1 **QtGuiWindowSystemWrapper** (osg::GraphicsContext::WindowingSystemInterface &*oldInterface*)

4.34.1.2 virtual ~**QtGuiWindowSystemWrapper** () [inline, protected, virtual]

#### 4.34.2 Member Function Documentation

4.34.2.1 osg::GraphicsContext \* **createGraphicsContext** (osg::GraphicsContext::Traits \* *traits*) [virtual]

4.34.2.2 void **EnableQtGUIWrapper** () [static]

4.34.2.3 unsigned int **getNumScreens** (const osg::GraphicsContext::ScreenIdentifier & *screenIdentifier* = osg::GraphicsContext::ScreenIdentifier()) [virtual]

4.34.2.4 void **getScreenResolution** (const osg::GraphicsContext::ScreenIdentifier & *screenIdentifier*, unsigned int & *width*, unsigned int & *height*) [virtual]

4.34.2.5 void **SetGLWidgetFactory** (**GLWidgetFactory** \* *factory*)

Supply a custom factory to create custom QGLWidgets.

4.34.2.6 bool **setScreenRefreshRate** (const osg::GraphicsContext::ScreenIdentifier & *screenIdentifier*, double *refreshRate*) [virtual]

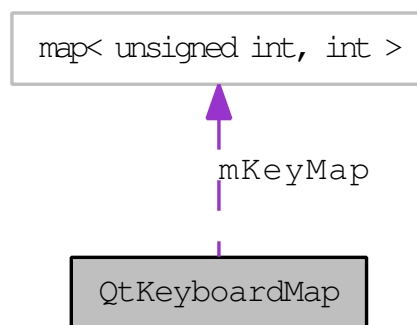
4.34.2.7 bool **setScreenResolution** (const osg::GraphicsContext::ScreenIdentifier & *screenIdentifier*, unsigned int *width*, unsigned int *height*) [virtual]

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

- **qtguiwindowssystemwrapper.h**
- **qtguiwindowssystemwrapper.cpp**

## 4.35 QtKeyboardMap Class Reference

Collaboration diagram for QtKeyboardMap:



### Public Member Functions

- **QtKeyboardMap** ()
- **~QtKeyboardMap** ()
- int **remapKey** (QKeyEvent \*event)

#### 4.35.1 Constructor & Destructor Documentation

4.35.1.1 **QtKeyboardMap** () [inline]

4.35.1.2 **~QtKeyboardMap** () [inline]

#### 4.35.2 Member Function Documentation

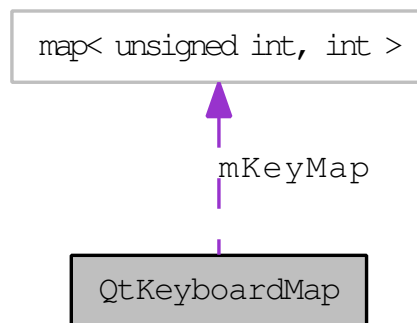
4.35.2.1 int **remapKey** (QKeyEvent \* *event*) [inline]

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

- **viewwindow.cpp**

## 4.36 QtKeyboardMap Class Reference

Collaboration diagram for QtKeyboardMap:



### Public Member Functions

- **QtKeyboardMap** ()
- **~QtKeyboardMap** ()
- int **remapKey** (QKeyEvent \*event)

#### 4.36.1 Constructor & Destructor Documentation

4.36.1.1 **QtKeyboardMap** () [inline]

4.36.1.2 **~QtKeyboardMap** () [inline]

#### 4.36.2 Member Function Documentation

4.36.2.1 int **remapKey** (QKeyEvent \* *event*) [inline]

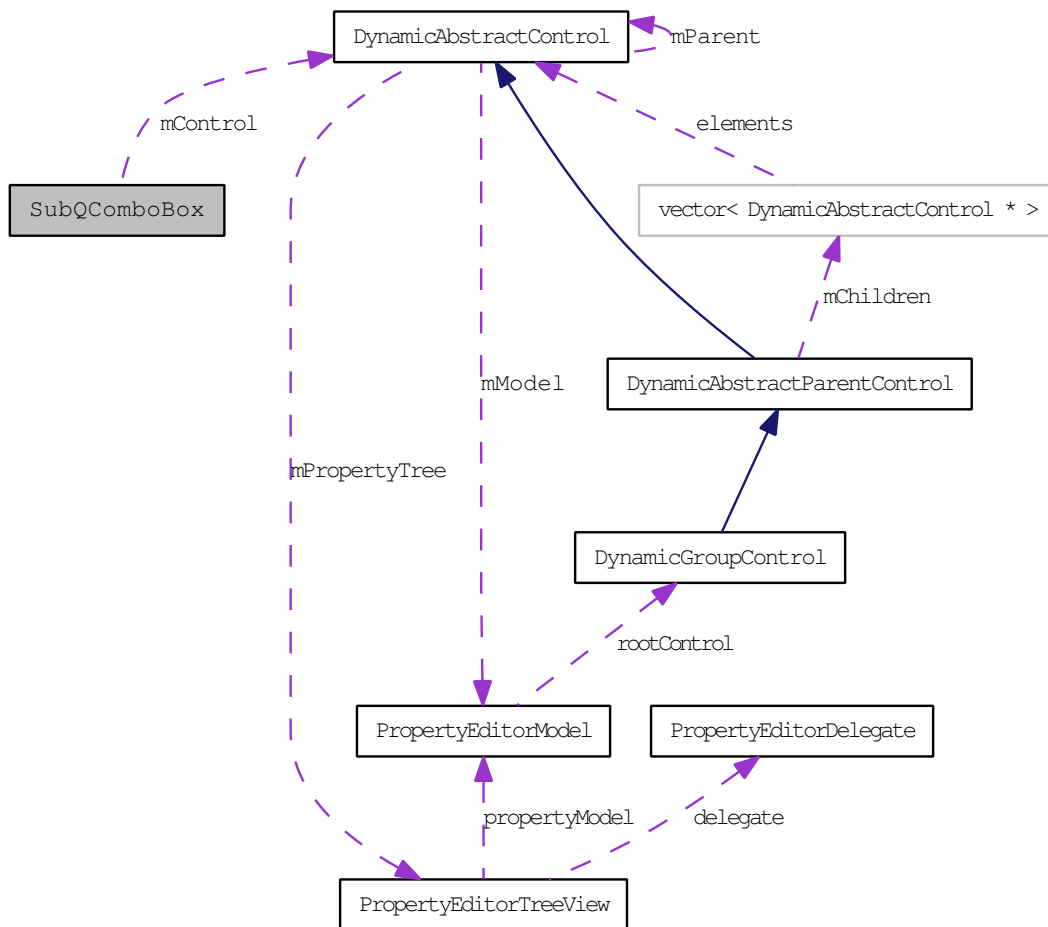
The documentation for this class was generated from the following file:

- **osgadapterwidget.cpp**

## 4.37 SubQComboBox Class Reference

Subclass of QComboBox.

#include <inc/dtQt/dynamicwidgets.h> Collaboration diagram for SubQComboBox:



### Public Member Functions

- **SubQComboBox** (QWidget \*parent, **DynamicAbstractControl** \*newControl)  
*Constructor.*
- virtual ~**SubQComboBox** ()  
*Destructor.*

#### 4.37.1 Detailed Description

Subclass of QComboBox. Destructor updates the parent **DynamicAbstractControl** (p. 21). If you use this call-back to clear out any handles to this control, then it will be safe to hold onto a control created in the createEditor() method.

#### 4.37.2 Constructor & Destructor Documentation

##### 4.37.2.1 SubQComboBox (QWidget \* *parent*, DynamicAbstractControl \* *newControl*) [inline]

Constructor.

##### 4.37.2.2 virtual ~SubQComboBox () [inline, virtual]

Destructor.

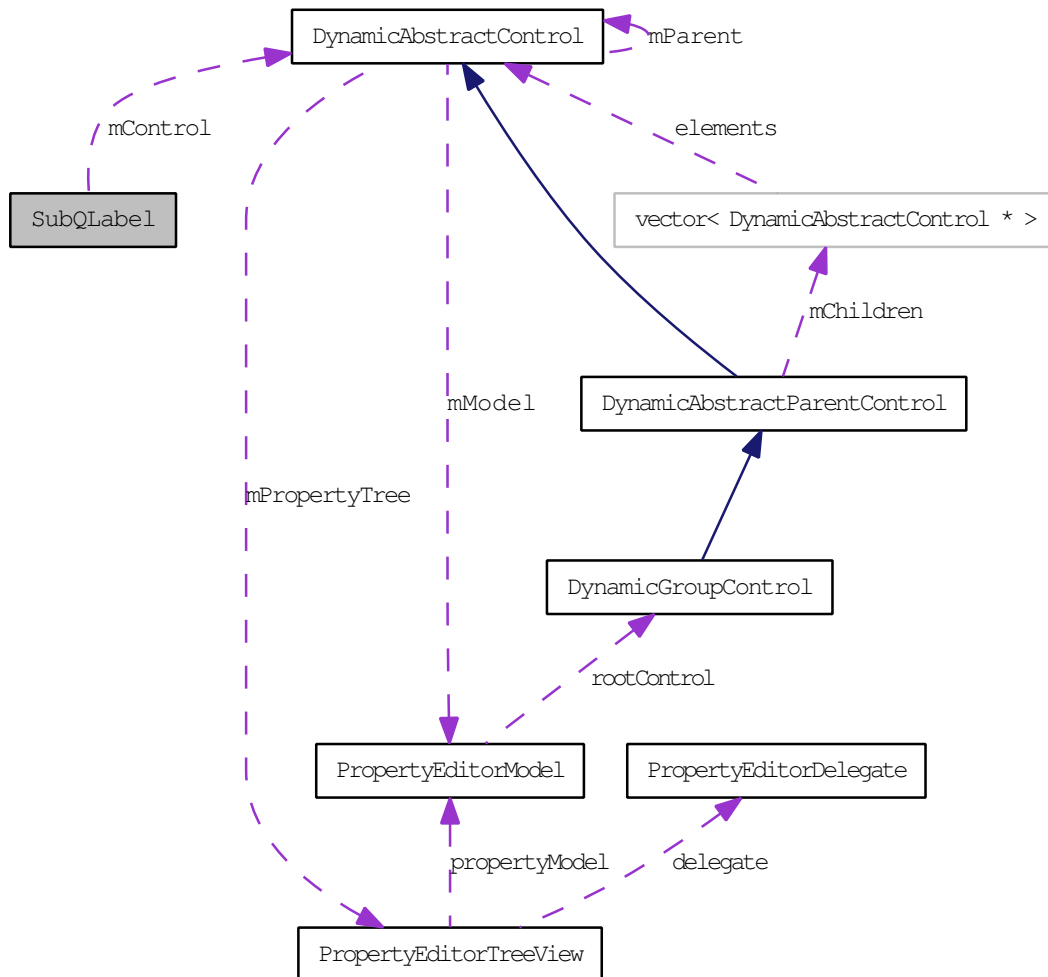
The documentation for this class was generated from the following file:

- **dynamicsubwidgets.h**

## 4.38 SubQLabel Class Reference

Subclass of QLabel.

#include <inc/dtQt/dynamicwidgets.h> Collaboration diagram for SubQLabel:



### Public Member Functions

- **SubQLabel** (QString name, QWidget \*parent, **DynamicAbstractControl** \*newControl)  
*Constructor.*
- virtual ~**SubQLabel** ()  
*Destructor.*

#### 4.38.1 Detailed Description

Subclass of QLabel. Destructor calls the handleSubEditDestroy. If you use this callback to clear out any handles to this control, then it will be safe to hold onto a control created in the createEditor() method.

#### 4.38.2 Constructor & Destructor Documentation

##### 4.38.2.1 SubQLabel (QString name, QWidget \* parent, DynamicAbstractControl \* newControl) [inline]

Constructor.

#### 4.38.2.2 virtual ~SubQLabel () [inline, virtual]

Destructor.

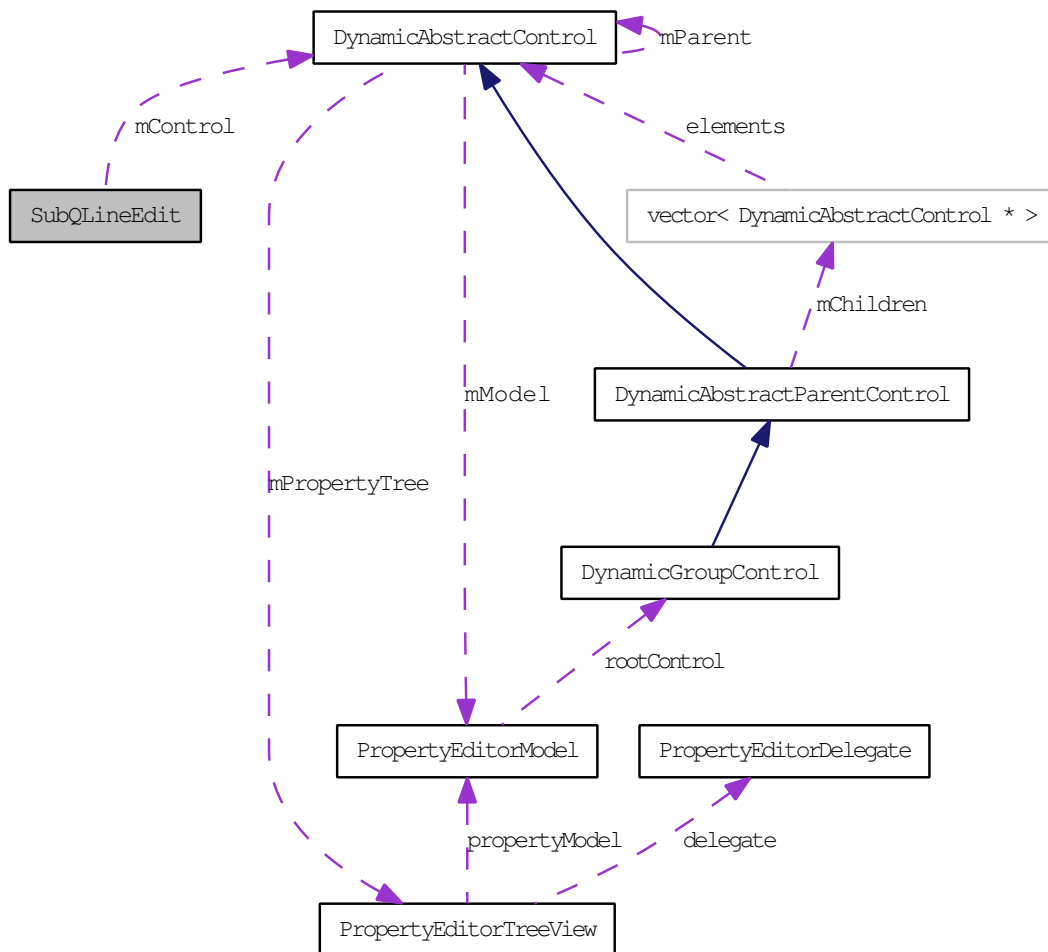
The documentation for this class was generated from the following file:

- **dynamicsubwidgets.h**

## 4.39 SubQLineEdit Class Reference

This is collection of subclasses of QWidgets that updates a dynamic abstract control on destruction.

#include <inc/dtQt/dynamicsubwidgets.h> Collaboration diagram for SubQLineEdit:



### Public Member Functions

- **SubQLineEdit** (QWidget \*parent, **DynamicAbstractControl** \*newControl)  
*Constructor.*
- virtual ~**SubQLineEdit** ()  
*Destructor.*

#### 4.39.1 Detailed Description

This is collection of subclasses of QWidgets that updates a dynamic abstract control on destruction. It's necessary because when you QT's Model View architecture, it has a bug. The original bug occurred when you selected a new object in a viewport while you had changed a value in the property editor. To cause it, select an object, and edit one of the values. While you're editing, go back to the viewport and select a new actor. This generates an event to change the property editor, but also causes the Tree/View architecture to delete your editor QWidget. We lose any changes we made right before we did the new selection.

So, to fix it, we trap the destruction event of our QWidget objects and tell them to update the model data. Nasty, and class heavy, but it was the only way sure fire way around the problem. Subclass of QLineEdit. Destructor updates the parent **DynamicAbstractControl** (p.21). If you use this callback to clear out any handles to this control, then it will be safe to hold onto a control created in the createEditor() method.

## 4.39.2 Constructor & Destructor Documentation

### 4.39.2.1 SubQLineEdit (QWidget \* *parent*, DynamicAbstractControl \* *newControl*) [inline]

Constructor.

### 4.39.2.2 virtual ~SubQLineEdit () [inline, virtual]

Destructor.

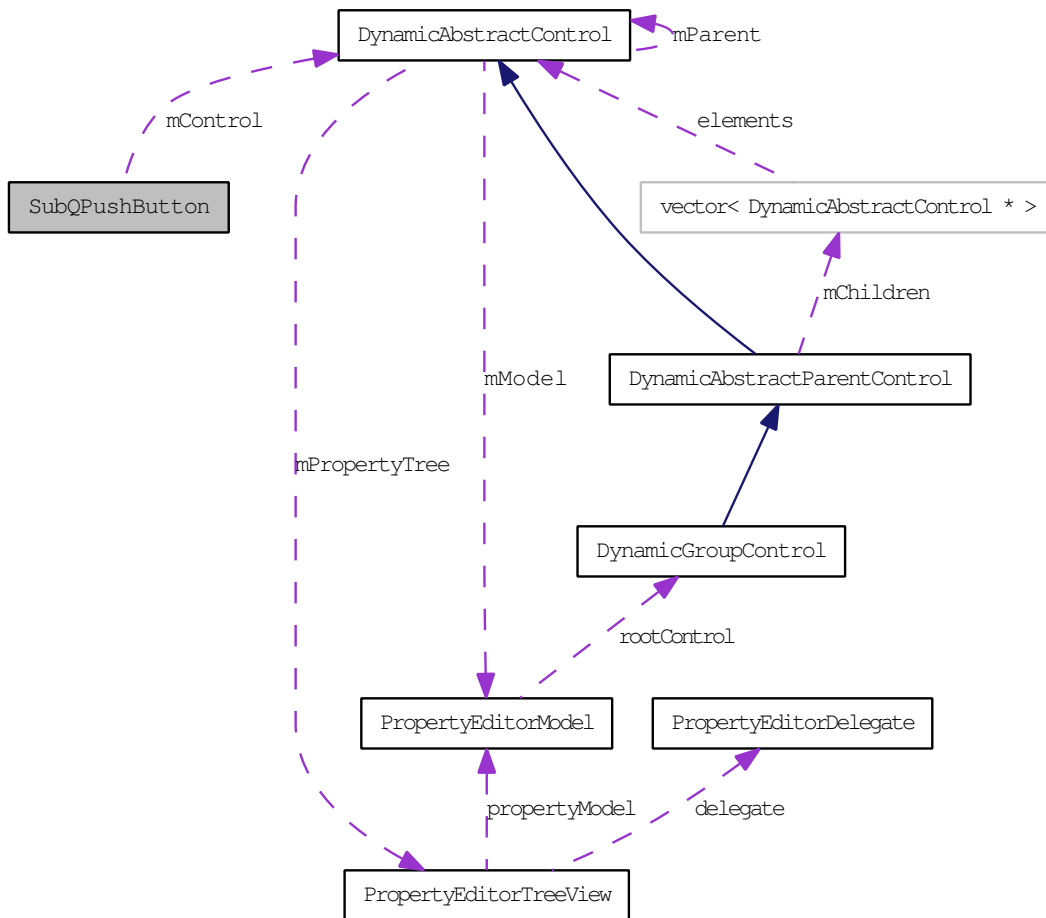
The documentation for this class was generated from the following file:

- **dynamicsubwidgets.h**

## 4.40 SubQPushButton Class Reference

Subclass of QPushButton.

#include <inc/dtQt/dynamicwidgets.h> Collaboration diagram for SubQPushButton:



### Public Member Functions

- **SubQPushButton** (QString name, QWidget \*parent, **DynamicAbstractControl** \*newControl)  
*Constructor.*
- virtual ~**SubQPushButton** ()  
*Destructor.*

#### 4.40.1 Detailed Description

Subclass of QPushButton. Destructor calls the handleSubEditDestroy(). If you use this callback to clear out any handles to this control, then it will be safe to hold onto a control created in the createEditor() method.

#### 4.40.2 Constructor & Destructor Documentation

##### 4.40.2.1 SubQPushButton (QString *name*, QWidget \* *parent*, **DynamicAbstractControl** \* *newControl*) [inline]

Constructor.

##### 4.40.2.2 virtual ~SubQPushButton () [inline, virtual]

Destructor.

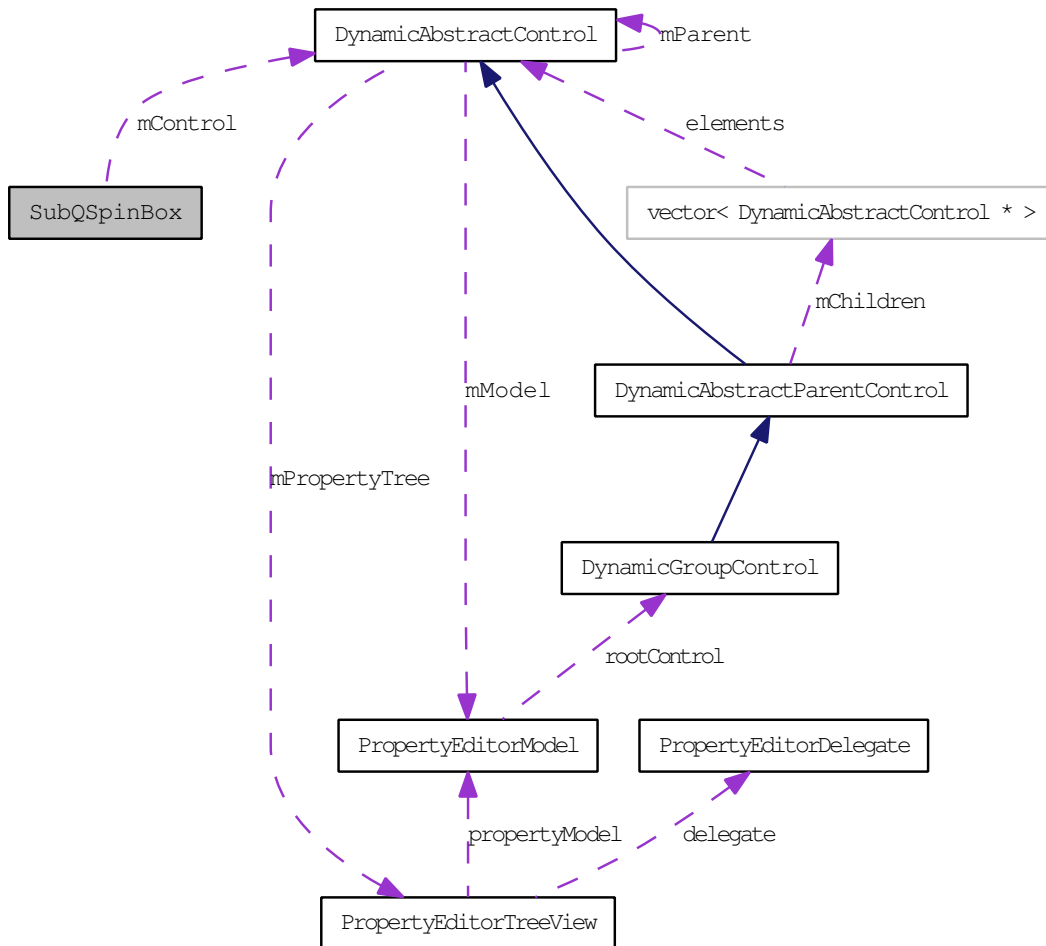
The documentation for this class was generated from the following file:

- **dynamicsubwidgets.h**

## 4.41 SubQSpinBox Class Reference

Subclass of QSpinBox.

#include <inc/dtQt/dynamicwidgets.h> Collaboration diagram for SubQSpinBox:



### Public Member Functions

- **SubQSpinBox** (QWidget \*parent, **DynamicAbstractControl** \*newControl)  
*Constructor.*
- virtual ~**SubQSpinBox** ()  
*Destructor.*

#### 4.41.1 Detailed Description

Subclass of QSpinBox. Destructor updates the parent **DynamicAbstractControl** (p. 21). If you use this callback to clear out any handles to this control, then it will be safe to hold onto a control created in the createEditor() method.

#### 4.41.2 Constructor & Destructor Documentation

##### 4.41.2.1 SubQSpinBox (QWidget \* *parent*, DynamicAbstractControl \* *newControl*) [inline]

Constructor.

#### 4.41.2.2 virtual ~SubQSpinBox () [inline, virtual]

Destructor.

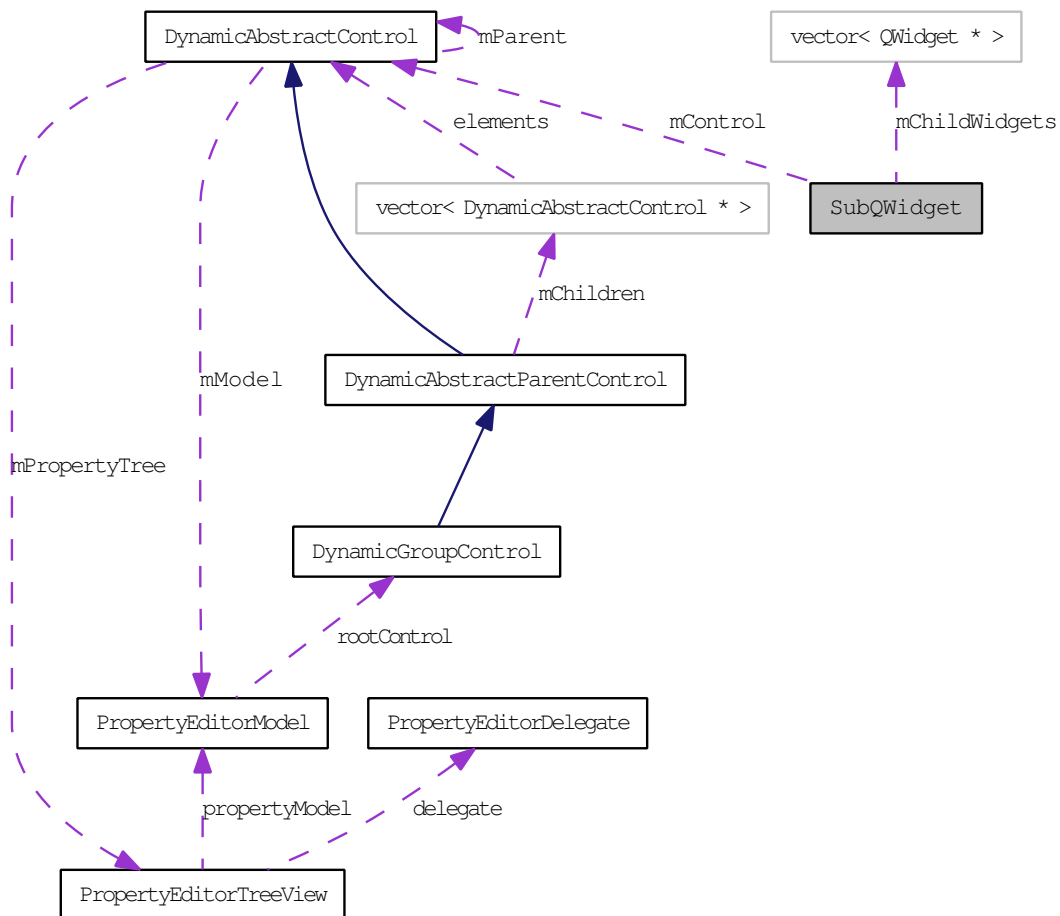
The documentation for this class was generated from the following file:

- **dynamicsubwidgets.h**

## 4.42 SubQWidget Class Reference

Subclass of QWidget.

#include <inc/dtQt/dynamicwidgets.h> Collaboration diagram for SubQWidget:



### Public Member Functions

- **SubQWidget** (QWidget \*parent, **DynamicAbstractControl** \*newControl)  
*Constructor.*
- virtual ~**SubQWidget** ()  
*Destructor.*
- void **addManagedChild** (QWidget \*child)  
*Add a child widget that we should track for palette changes.*

#### 4.42.1 Detailed Description

Subclass of QWidget. Used to trap multiple children if needed. Unlike the other SubXYZ classes, this one does NOT work with the parent control

#### 4.42.2 Constructor & Destructor Documentation

##### 4.42.2.1 SubQWidget (QWidget \* parent, DynamicAbstractControl \* newControl) [inline]

Constructor.

**4.42.2.2 virtual ~SubQWidget () [inline, virtual]**

Destructor.

**4.42.3 Member Function Documentation****4.42.3.1 void addManagedChild (QWidget \* *child*) [inline]**

Add a child widget that we should track for palette changes.

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

- **dynamicsubwidgets.h**

## 4.43 ViewWindow Class Reference

Little class used to hold the Delta3D rendering surface.

```
#include <inc/dtQt/viewwindow.h>
```

### Public Slots

- void **ThreadedUpdateGL** ()

### Public Member Functions

- **ViewWindow** (bool drawOnSeparateThread, QWidget \*parent=NULL, const QGLWidget \*shareWidget=NULL, Qt::WindowFlags f=NULL)
- **~ViewWindow** ()
- const osgViewer::GraphicsWindow & **GetGraphicsWindow** () const
- osgViewer::GraphicsWindow & **GetGraphicsWindow** ()
- virtual void **OnMessage** (MessageData \*)
- void **SetGraphicsWindow** (osgViewer::GraphicsWindow &newWindow)
- void **ThreadedInitializeGL** ()
- void **ThreadedMakeCurrent** ()

### Protected Member Functions

- virtual void **glDraw** ()
- virtual void **initializeGL** ()
- virtual void **keyPressEvent** (QKeyEvent \*event)
- virtual void **keyReleaseEvent** (QKeyEvent \*event)
- virtual void **mouseMoveEvent** (QMouseEvent \*event)
- virtual void **mousePressEvent** (QMouseEvent \*event)
- virtual void **mouseReleaseEvent** (QMouseEvent \*event)
- virtual void **paintGL** ()
- void **paintGLImpl** ()
- virtual void **resizeGL** (int width, int height)
- void **resizeGLImpl** (int width, int height)
- virtual void **wheelEvent** (QWheelEvent \*event)

### Protected Attributes

- volatile bool **mDoResize**
- bool **mDrawOnSeparateThread**
- dtCore::RefPtr< osgViewer::GraphicsWindow > **mGraphicsWindow**
- QGLContext \* **mThreadGLContext**
- QTimer **mTimer**

#### 4.43.1 Detailed Description

Little class used to hold the Delta3D rendering surface.

### 4.43.2 Constructor & Destructor Documentation

4.43.2.1 **ViewWindow** (*bool drawOnSeparateThread*, *QWidget \* parent = NULL*, *const QGLWidget \* shareWidget = NULL*, *Qt::WindowFlags f = NULL*)

4.43.2.2 **~ViewWindow** ()

### 4.43.3 Member Function Documentation

4.43.3.1 **const osgViewer::GraphicsWindow & GetGraphicsWindow** () **const**

4.43.3.2 **osgViewer::GraphicsWindow & GetGraphicsWindow** ()

4.43.3.3 **void glDraw** () [**protected**, **virtual**]

4.43.3.4 **void initializeGL** () [**protected**, **virtual**]

4.43.3.5 **void keyPressEvent** (*QKeyEvent \* event*) [**protected**, **virtual**]

4.43.3.6 **void keyReleaseEvent** (*QKeyEvent \* event*) [**protected**, **virtual**]

4.43.3.7 **void mouseMoveEvent** (*QMouseEvent \* event*) [**protected**, **virtual**]

4.43.3.8 **void mousePressEvent** (*QMouseEvent \* event*) [**protected**, **virtual**]

4.43.3.9 **void mouseReleaseEvent** (*QMouseEvent \* event*) [**protected**, **virtual**]

4.43.3.10 **void OnMessage** (*MessageData \* data*) [**virtual**]

4.43.3.11 **void paintGL** () [**protected**, **virtual**]

4.43.3.12 **void paintGLImpl** () [**protected**]

4.43.3.13 **void resizeGL** (*int width*, *int height*) [**protected**, **virtual**]

set message to resize. this is actually a race condition. It needs to be locked.

4.43.3.14 **void resizeGLImpl** (*int width*, *int height*) [**protected**]

4.43.3.15 **void SetGraphicsWindow** (*osgViewer::GraphicsWindow & newWindow*)

4.43.3.16 **void ThreadedInitializeGL** ()

4.43.3.17 **void ThreadedMakeCurrent** ()

4.43.3.18 **void ThreadedUpdateGL** () [**slot**]

4.43.3.19 **void wheelEvent** (*QWheelEvent \* event*) [**protected**, **virtual**]

### 4.43.4 Member Data Documentation

4.43.4.1 **volatile bool mDoResize** [**protected**]

4.43.4.2 **bool mDrawOnSeparateThread** [**protected**]

4.43.4.3 **dtCore::RefPtr<osgViewer::GraphicsWindow> mGraphicsWindow** [**protected**]

4.43.4.4 **QGLContext\* mThreadGLContext** [**protected**]

4.43.4.5 **QTimer mTimer** [**protected**]

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

- **viewwindow.h**
- **viewwindow.cpp**



## File Documentation

---

### 5.1 baselibrarylisteditor.cpp File Reference

```
#include <QtGui/QVBoxLayout>
#include <QtGui/QHBoxLayout>
#include <QtGui/QGridLayout>
#include <QtGui/QPushButton>
#include <QtGui/QMessageBox>
#include <QtGui/QFileDialog>
#include <QtGui/QListWidgetItem>
#include <QtGui/QListWidget>
#include <QtCore/QStringList>
#include <QtGui/QMainWindow>
#include <QtGui/QGroupBox>
#include <dtQt/baselibrarylisteditor.h>
#include <dtUtil/log.h>
#include <dtUtil/librarysharingmanager.h>
#include <osgDB/FileNameUtils>
```

#### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

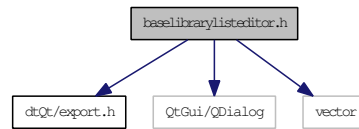
## 5.2 baselibrarylisteditor.h File Reference

```
#include <dtQt/export.h>
```

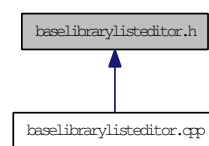
```
#include <QtGui/QDialog>
```

```
#include <vector>
```

Include dependency graph for baselibrarylisteditor.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **BaseLibraryListEditor**

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.3 basepropertyeditor.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/basepropertyeditor.h>
#include <dtQt/dynamicabstractcontrol.h>
#include <dtQt/dynamicgroupcontrol.h>
#include <dtQt/dynamiclabelcontrol.h>
#include <dtQt/propertyeditormodel.h>
#include <dtQt/propertyeditortreeview.h>
#include <dtQt/dynamiccontainercontrol.h>
#include <QtCore/QStringList>
#include <QtGui/QLabel>
#include <QtGui/QGridLayout>
#include <QtGui/QScrollArea>
#include <QtGui/QScrollBar>
#include <QtGui/QTreeWidgetItem>
#include <QtGui/QMainWindow>
#include <QtGui/QHeaderView>
#include <QtGui/QGroupBox>
#include <QtGui/QTreeView>
#include <QtGui/QAction>
#include <dtCore/deltadrawable.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtDAL/exceptionenum.h>
#include <dtDAL/datatype.h>
#include <dtDAL/librarymanager.h>
#include <dtDAL/map.h>
#include <dtUtil/log.h>
#include <dtUtil/tree.h>
#include <osg/Referenced>
#include <vector>
#include <cmath>
```

### Namespaces

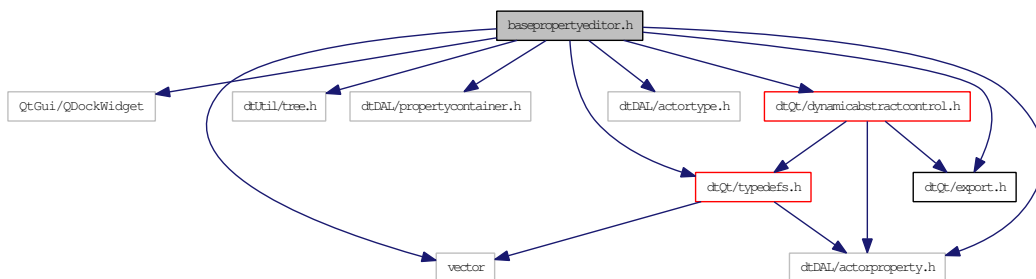
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

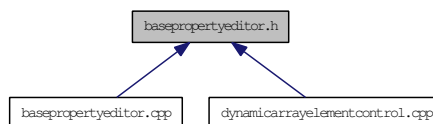
## 5.4 basepropertyeditor.h File Reference

```
#include <QtGui/QDockWidget>
#include <vector>
#include <dtUtil/tree.h>
#include <dtDAL/propertycontainer.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/actortype.h>
#include <dtQt/typedefs.h>
#include <dtQt/dynamicabstractcontrol.h>
#include <dtQt/export.h>
```

Include dependency graph for basepropertyeditor.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **BasePropertyEditor**

*This class is the property editor for displaying and editing properties of selected objects.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

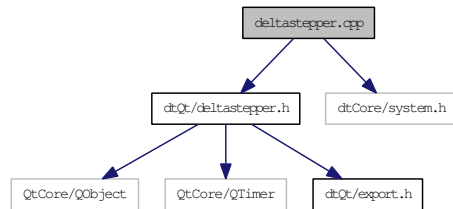
*Contains classes that help integrate Delta3D with Qt.*

## 5.5 deltastepper.cpp File Reference

```
#include <dtQt/deltastepper.h>
```

```
#include <dtCore/system.h>
```

Include dependency graph for deltastepper.cpp:



### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

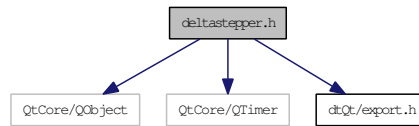
## 5.6 deltastepper.h File Reference

```
#include <QtCore/QObject>
```

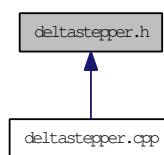
```
#include <QtCore/QTimer>
```

```
#include <dtQt/export.h>
```

Include dependency graph for deltastepper.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DeltaStepper**

### Namespaces

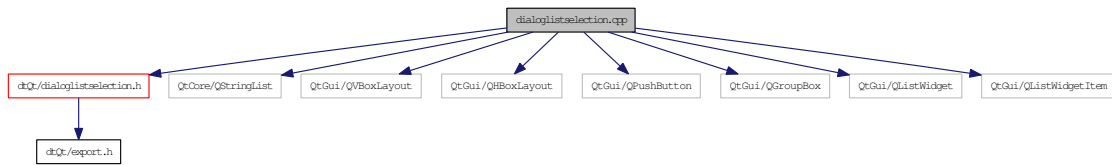
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.7 dialoglistselection.cpp File Reference

```
#include <dtQt/dialoglistselection.h>
#include <QtCore/QStringList>
#include <QtGui/QVBoxLayout>
#include <QtGui/QHBoxLayout>
#include <QtGui/QPushButton>
#include <QtGui/QGroupBox>
#include <QtGui/QListWidget>
#include <QtGui/QListWidgetItem>
```

Include dependency graph for dialoglistselection.cpp:



### Namespaces

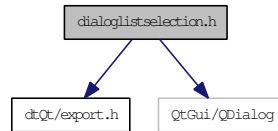
- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

## 5.8 dialoglistselection.h File Reference

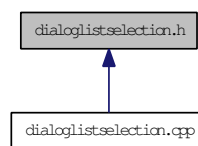
```
#include <dtQt/export.h>
```

```
#include <QtGui/QDialog>
```

Include dependency graph for dialoglistselection.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DialogListSelection**

*This is a generic dialog box for presenting a list of items for the user to choose from.*

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.9 dynamicabstractcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicabstractcontrol.h>
#include <dtQt/propertyeditortreeview.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/enginepropertytypes.h>
#include <QtGui/QColor>
#include <QtGui/QPalette>
#include <QtGui/QWidget>
#include <QtGui/QPainter>
#include <QtCore/QSize>
#include <QtGui/QStyleOptionViewItem>
#include <dtQt/dynamicabstractparentcontrol.h>
#include <dtQt/dynamicboolcontrol.h>
#include <dtQt/dynamiccolorrgbacontrol.h>
#include <dtQt/dynamicenumcontrol.h>
#include <dtQt/dynamicfloatcontrol.h>
#include <dtQt/dynamicdoublecontrol.h>
#include <dtQt/dynamicintcontrol.h>
#include <dtQt/dynamiclabelcontrol.h>
#include <dtQt/dynamiclongcontrol.h>
#include <dtQt/dynamicstringcontrol.h>
#include <dtQt/dynamicvecncontrol.h>
#include <dtQt/dynamicarraycontrol.h>
#include <dtQt/dynamiccontainercontrol.h>
#include <dtDAL/datatype.h>
```

### Namespaces

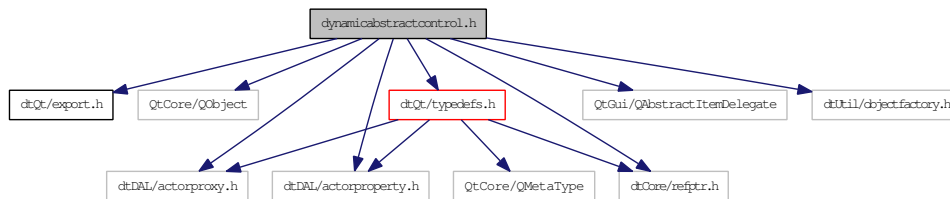
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.10 dynamicabstractcontrol.h File Reference

```
#include <dtQt/export.h>
#include <QtCore/QObject>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtQt/typedefs.h>
#include <QtGui/QAbstractItemDelegate>
#include <dtCore/refptr.h>
#include <dtUtil/objectfactory.h>
```

Include dependency graph for dynamicabstractcontrol.h:



### Classes

- class **DynamicAbstractControl**  
*This is the base class for all the dynamic controls that are used in the property editor.*
- class **DynamicControlFactory**

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

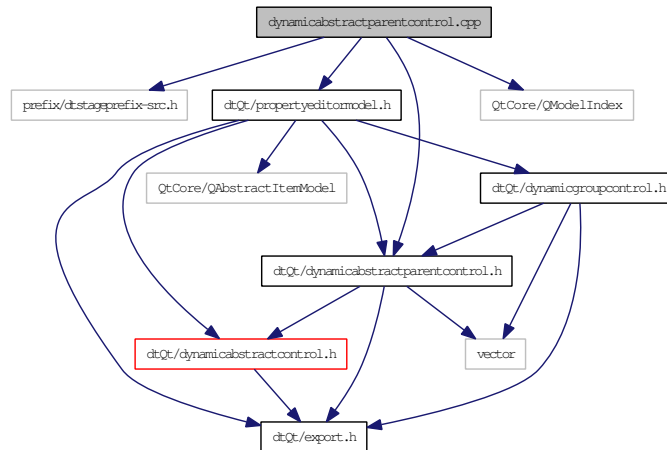
### Variables

- const unsigned int **NUM\_DECIMAL\_DIGITS\_DOUBLE** = 15
- const unsigned int **NUM\_DECIMAL\_DIGITS\_FLOAT** = 8

## 5.11 dynamicabstractparentcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicabstractparentcontrol.h>
#include <dtQt/propertyeditormodel.h>
#include <QtCore/QModelIndex>
```

Include dependency graph for dynamicabstractparentcontrol.cpp:



### Namespaces

- namespace **dtQt**  
Contains classes that help integrate Delta3D with Qt.

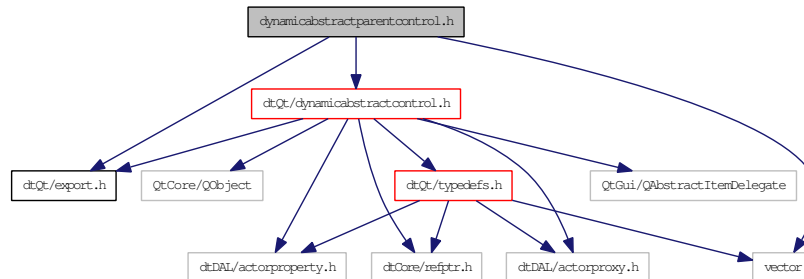
## 5.12 dynamicabstractparentcontrol.h File Reference

```
#include <dtQt/export.h>
```

```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <vector>
```

Include dependency graph for dynamicabstractparentcontrol.h:



### Classes

- class **DynamicAbstractParentControl**  
*This is a base class for any dynamic control that has children.*

### Namespaces

- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

## 5.13 dynamicarraycontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <QtGui/QWidget>
#include <QtGui/QGridLayout>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtGui/QDoubleValidator>
#include <dtQt/dynamicarraycontrol.h>
#include <dtQt/dynamicarrayelementcontrol.h>
#include <dtQt/propertyeditormodel.h>
#include <dtQt/propertyeditortreeview.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/arrayactorpropertybase.h>
#include <dtUtil/log.h>
#include <QtGui/QPushButton>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

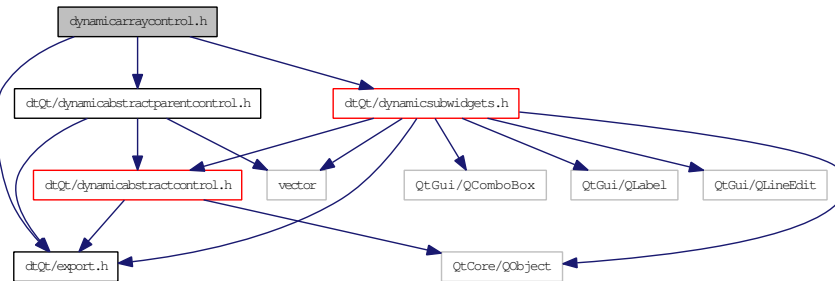
## 5.14 dynamicarraycontrol.h File Reference

```
#include <dtQt/export.h>
```

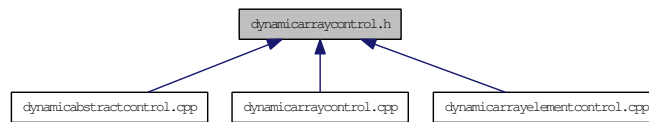
```
#include <dtQt/dynamicabstractparentcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicarraycontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicArrayControl**

*This is the dynamic control for the float data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.15 dynamicarrayelementcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <QtGui/QWidget>
#include <QtGui/QGridLayout>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtGui/QDoubleValidator>
#include <QtGui/QPushButton>
#include <dtQt/dynamicarrayelementcontrol.h>
#include <dtQt/dynamicarraycontrol.h>
#include <dtQt/basepropertyeditor.h>
#include <dtQt/propertyeditormodel.h>
#include <dtQt/propertyeditortreeview.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/arrayactorpropertybase.h>
#include <dtUtil/log.h>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

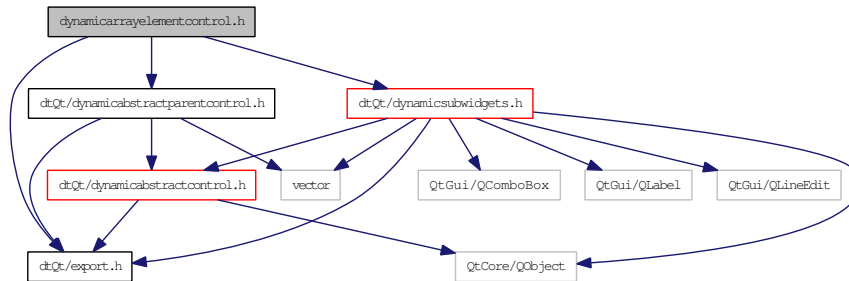
## 5.16 dynamicarrayelementcontrol.h File Reference

```
#include <dtQt/export.h>
```

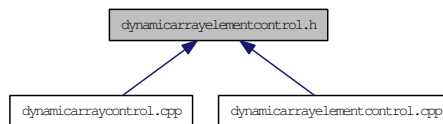
```
#include <dtQt/dynamicabstractparentcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicarrayelementcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicArrayElementControl**

*This is the dynamic control for the float data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.17 dynamicboolcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicboolcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtGui/QComboBox>
#include <QtGui/QGridLayout>
#include <QtGui/QIntValidator>
#include <QtGui/QLabel>
#include <QtGui/QWidget>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

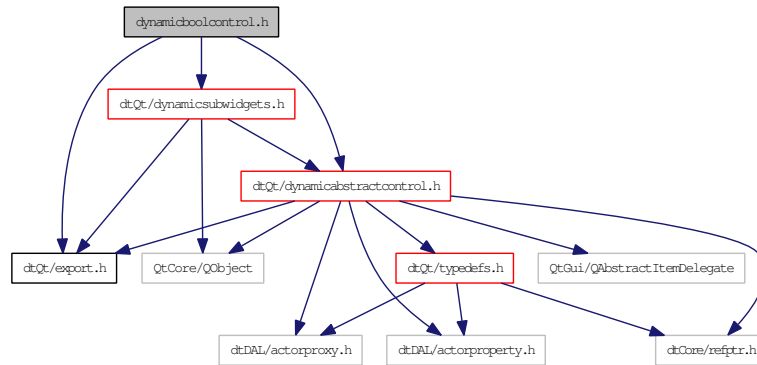
## 5.18 dynamicboolcontrol.h File Reference

```
#include <dtQt/export.h>
```

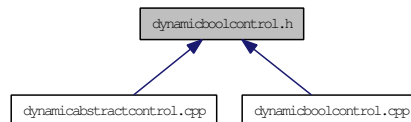
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicboolcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicBoolControl**

*This is the dynamic control for the bool data type - used in the property editor.*

### Namespaces

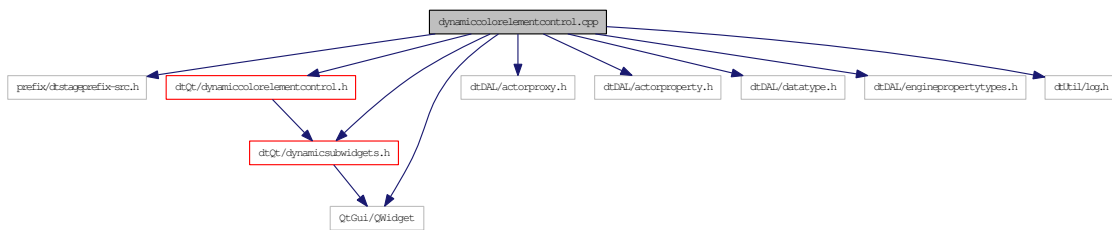
- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.19 dynamiccolorelementcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamiccolorelementcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtGui/QWidget>
```

Include dependency graph for dynamiccolorelementcontrol.cpp:



### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

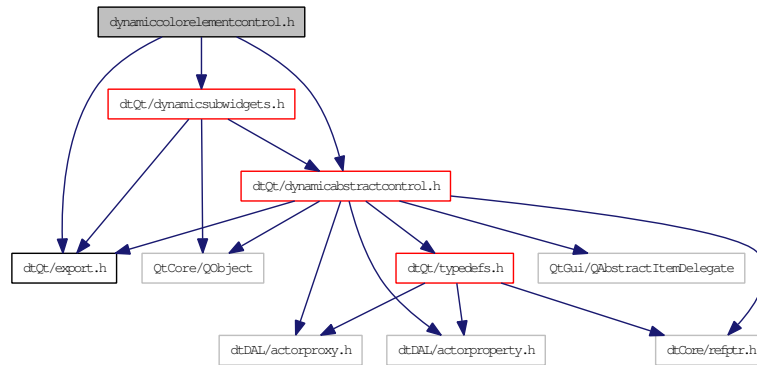
## 5.20 dynamiccolorelementcontrol.h File Reference

```
#include <dtQt/export.h>
```

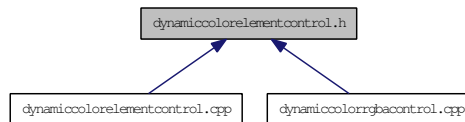
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamiccolorelementcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicColorElementControl**

*This is the a sub control used by the various color property classes.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.21 dynamiccolorrgbacontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamiccolorelementcontrol.h>
#include <dtQt/dynamiccolorrgbacontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtQt/propertyeditortreeview.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <dtUtil/mathdefines.h>
#include <QtGui/QColorDialog>
#include <QtGui/QColor>
#include <QtGui/QGridLayout>
#include <QtGui/QHBoxLayout>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtGui/QPainter>
#include <QtGui/QPaintDevice>
#include <QtGui/QPaintEngine>
#include <QtGui/QPushButton>
#include <QtCore/QSize>
#include <QtGui/QStyleOptionViewItem>
#include <QtGui/QWidget>
```

### Namespaces

- namespace **dtQt**

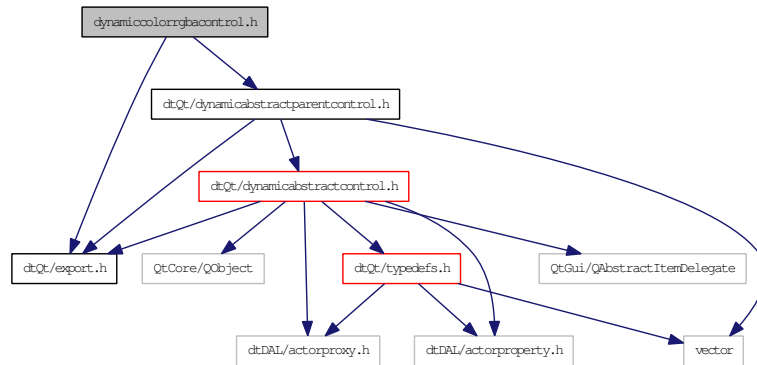
*Contains classes that help integrate Delta3D with Qt.*

## 5.22 dynamiccolorrgbacontrol.h File Reference

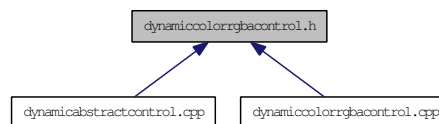
```
#include <dtQt/export.h>
```

```
#include <dtQt/dynamicabstractparentcontrol.h>
```

Include dependency graph for dynamiccolorrgbacontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicColorRGBAControl**

*This is the dynamic control for the an RGBA Color picker - used in the property editor It adds a group of child elements to the tree, since you can't edit 3 things in one control easily.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.23 dynamiccontainercontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <QtGui/QWidget>
#include <QtGui/QGridLayout>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtGui/QDoubleValidator>
#include <dtQt/dynamiccontainercontrol.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/containeractorproperty.h>
#include <dtUtil/log.h>
#include <dtQt/propertyeditormodel.h>
#include <dtQt/propertyeditortreeview.h>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

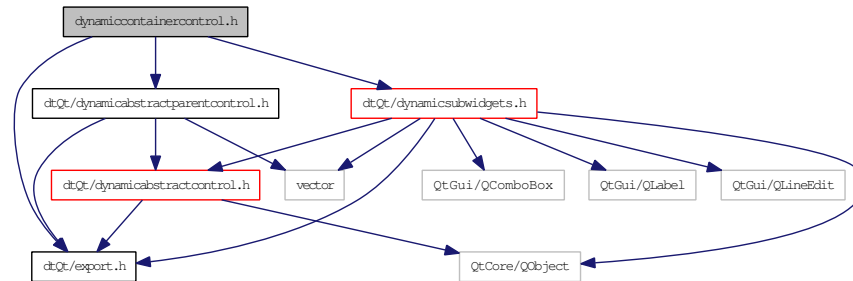
## 5.24 dynamiccontainercontrol.h File Reference

```
#include <dtQt/export.h>
```

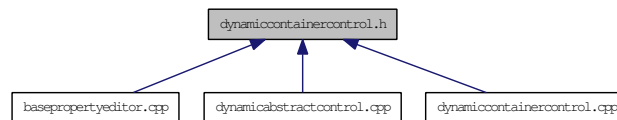
```
#include <dtQt/dynamicabstractparentcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamiccontainercontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicContainerControl**

*This is the dynamic control for the float data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.25 dynamicdoublecontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicdoublecontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtGui/QWidget>
#include <QtGui/QLineEdit>
#include <QtGui/QDoubleValidator>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

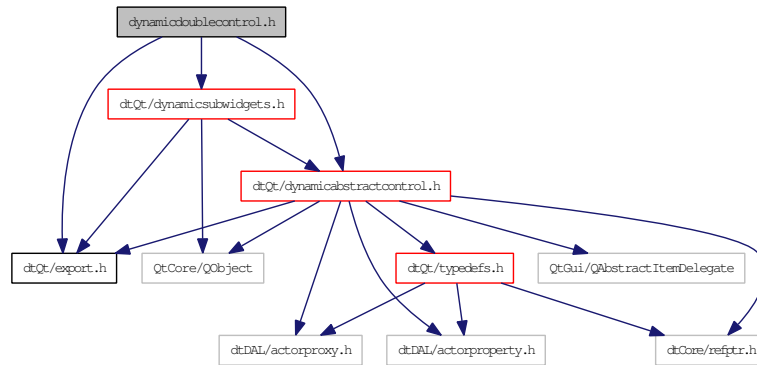
## 5.26 dynamicdoublecontrol.h File Reference

```
#include <dtQt/export.h>
```

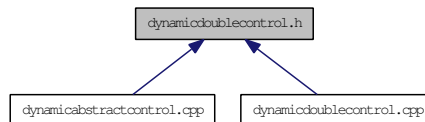
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicdoublecontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicDoubleControl**

*This is the dynamic control for the double data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.27 dynamicenumcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtUtil/enumeration.h>
#include <dtQt/dynamicenumcontrol.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtGui/QComboBox>
#include <QtGui/QGridLayout>
#include <QtGui/QIntValidator>
#include <QtGui/QLabel>
#include <QtGui/QWidget>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

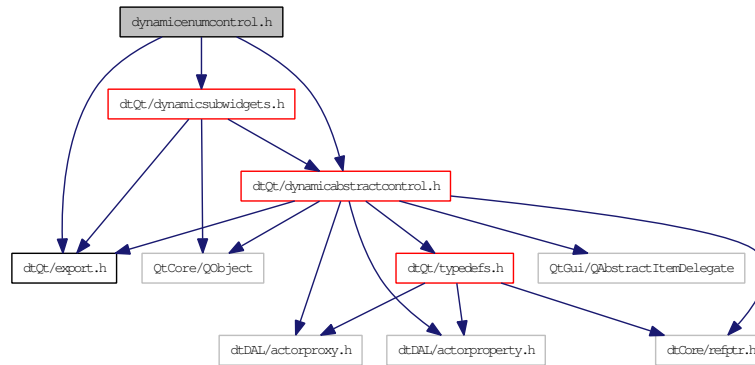
## 5.28 dynamicenumcontrol.h File Reference

```
#include <dtQt/export.h>
```

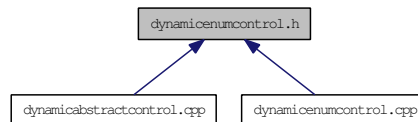
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicenumcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicEnumControl**

*This is the dynamic control for the enum data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.29 dynamicfloatcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicfloatcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtGui/QWidget>
#include <QtGui/QLineEdit>
#include <QtGui/QDoubleValidator>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

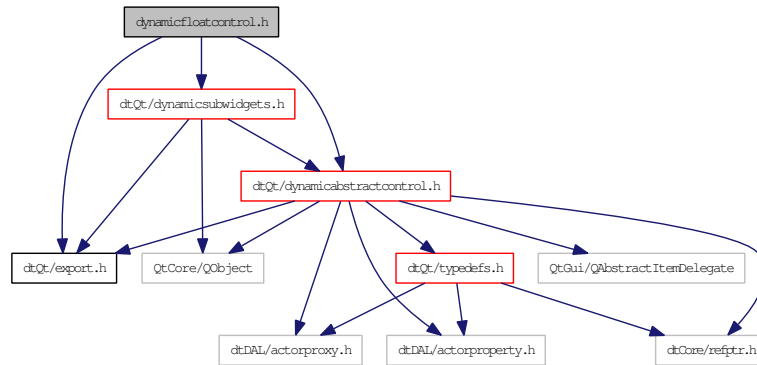
## 5.30 dynamicfloatcontrol.h File Reference

```
#include <dtQt/export.h>
```

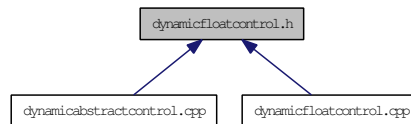
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicfloatcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicFloatControl**

*This is the dynamic control for the float data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.31 dynamicgroupcontrol.cpp File Reference

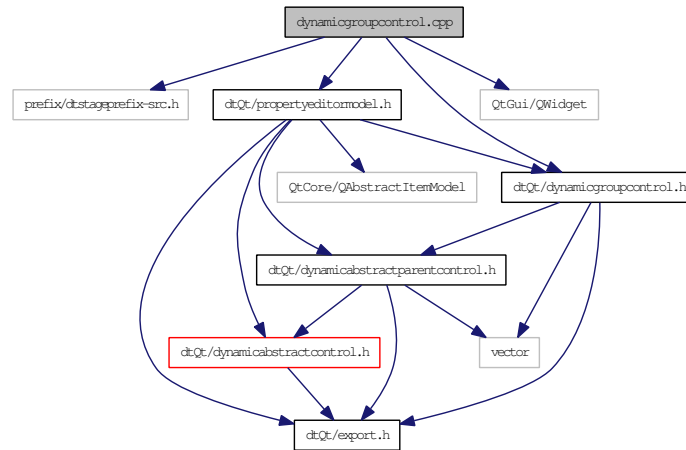
```
#include <prefix/dtstageprefix-src.h>
```

```
#include <dtQt/dynamicgroupcontrol.h>
```

```
#include <dtQt/propertyeditormodel.h>
```

```
#include <QtGui/QWidget>
```

Include dependency graph for dynamicgroupcontrol.cpp:



### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

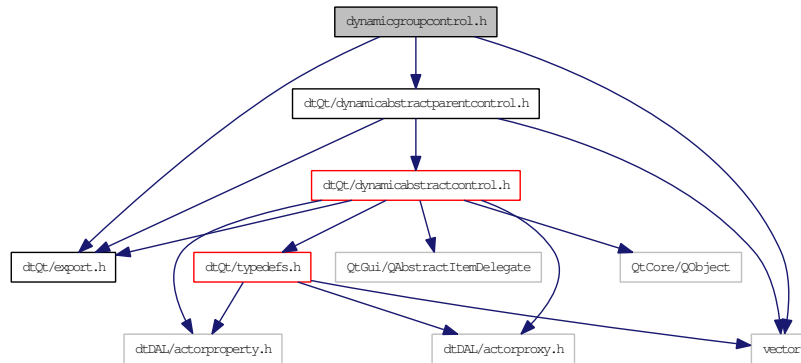
## 5.32 dynamicgroupcontrol.h File Reference

```
#include <dtQt/export.h>
```

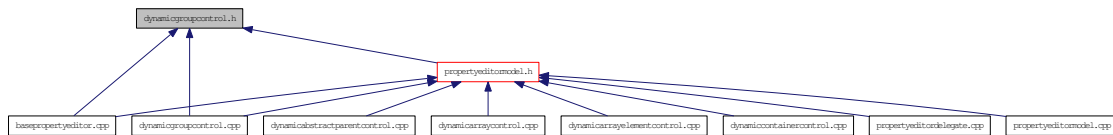
```
#include <dtQt/dynamicabstractparentcontrol.h>
```

```
#include <vector>
```

Include dependency graph for dynamicgroupcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicGroupControl**

*This is the dynamic control for the Group data type - used in the property editor The primary purpose of the group control is to provide a visual grouping of property types so that they aren't all laid out together.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.33 dynamicintcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicintcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtGui/QGridLayout>
#include <QtGui/QWidget>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtGui/QIntValidator>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

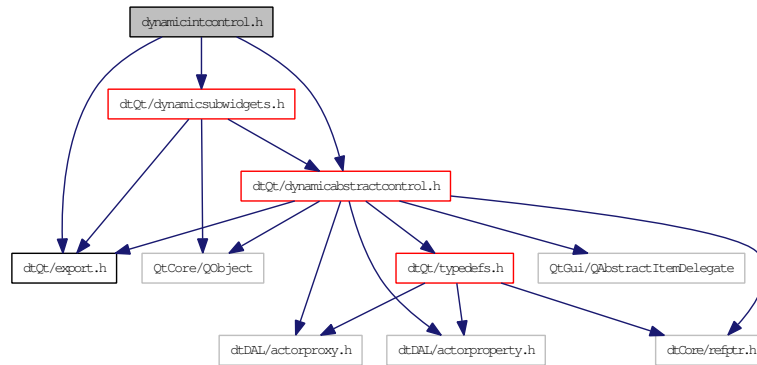
## 5.34 dynamicintcontrol.h File Reference

```
#include <dtQt/export.h>
```

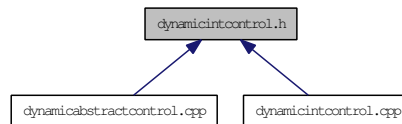
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicintcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicIntControl**

*This is the dynamic control for the int data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.35 dynamiclabelcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamiclabelcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/enginepropertytypes.h>
#include <QtGui/QGridLayout>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtCore/QString>
#include <QtGui/QWidget>
```

### Namespaces

- namespace **dtQt**

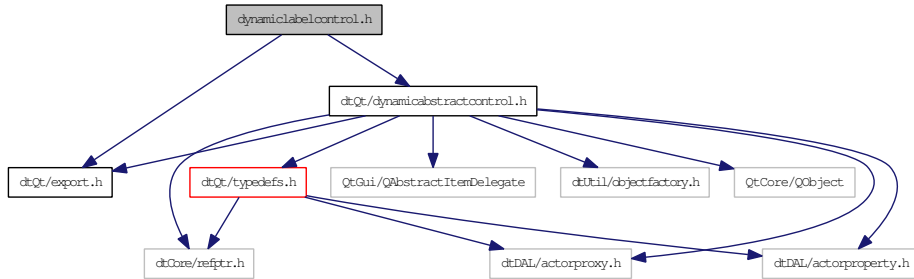
*Contains classes that help integrate Delta3D with Qt.*

## 5.36 dynamiclabelcontrol.h File Reference

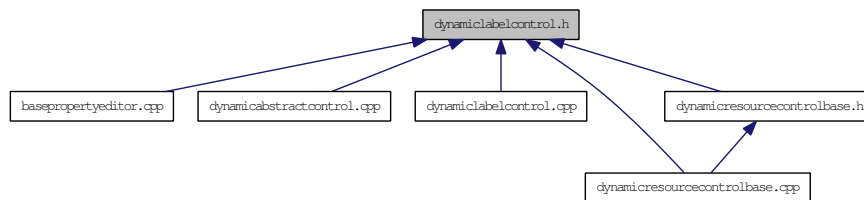
```
#include <dtQt/export.h>
```

```
#include <dtQt/dynamicabstractcontrol.h>
```

Include dependency graph for dynamiclabelcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicLabelControl**  
*This is the dynamic control that is a bit odd.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

## 5.37 dynamiclongcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamiclongcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtGui/QGridLayout>
#include <QtGui/QWidget>
#include <QtGui/QIntValidator>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

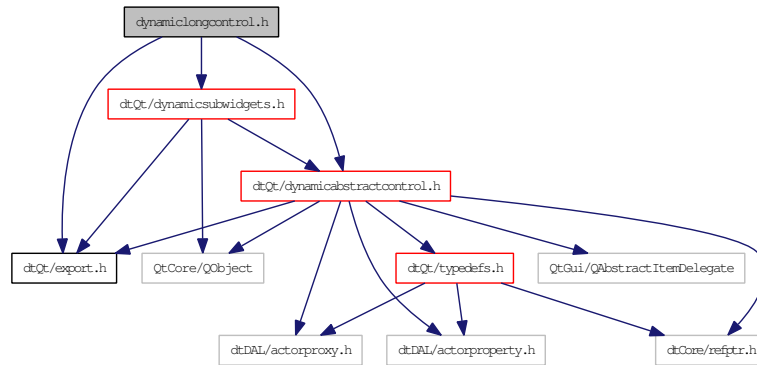
## 5.38 dynamiclongcontrol.h File Reference

```
#include <dtQt/export.h>
```

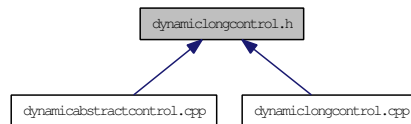
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamiclongcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicLongControl**

*This is the dynamic control for the long data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.39 dynamicresourcecontrolbase.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicresourcecontrolbase.h>
#include <dtQt/dynamiclabelcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtQt/propertyeditortreeview.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtDAL/resourcedescriptor.h>
#include <dtUtil/log.h>
#include <QtCore/QSize>
#include <QtCore/QRect>
#include <QtGui/QHBoxLayout>
#include <QtGui/QGridLayout>
#include <QtGui/QLabel>
#include <QtGui/QPushButton>
#include <QtGui/QWidget>
#include <QtGui/QColor>
#include <QtGui/QPalette>
#include <QtGui/QFocusFrame>
```

### Namespaces

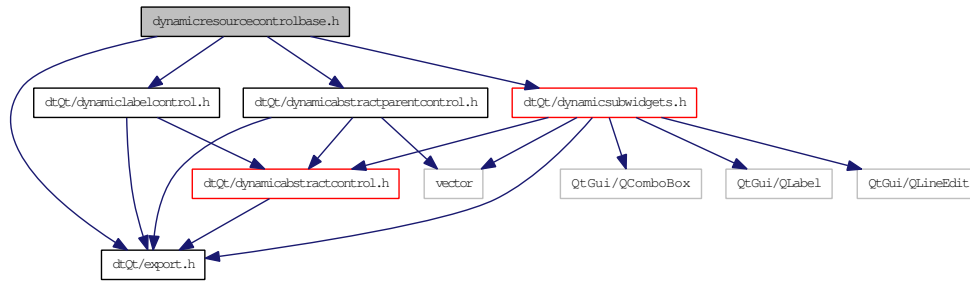
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

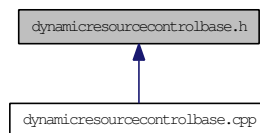
## 5.40 dynamicresourcecontrolbase.h File Reference

```
#include <dtQt/export.h>
#include <dtQt/dynamicabstractparentcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtQt/dynamiclabelcontrol.h>
```

Include dependency graph for dynamicresourcecontrolbase.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicResourceControlBase**  
*This is the resource actor property.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

## 5.41 dynamicstringcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicstringcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtGui/QGridLayout>
#include <QtGui/QWidget>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

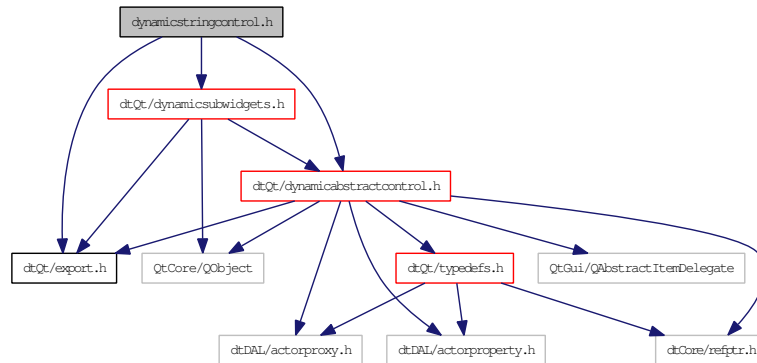
## 5.42 dynamicstringcontrol.h File Reference

```
#include <dtQt/export.h>
```

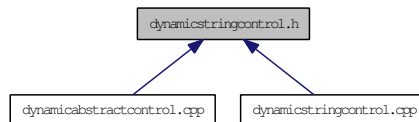
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicstringcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicStringControl**

*This is the dynamic control for the String data type - used in the property editor.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.43 dynamicsubwidgets.h File Reference

```
#include <dtQt/export.h>
#include <dtQt/dynamicabstractcontrol.h>
#include <QtGui/QComboBox>
#include <QtGui/QLabel>
#include <QtGui/QLineEdit>
#include <QtCore/QObject>
#include <QtGui/QPalette>
#include <QtGui/QPushButton>
#include <QtGui/QSpinBox>
#include <QtGui/QWidget>
#include <vector>
```

### Classes

- class **SubQComboBox**  
*Subclass of QComboBox.*
- class **SubQLabel**  
*Subclass of QLabel.*
- class **SubQLineEdit**  
*This is collection of subclasses of QWidgets that updates a dynamic abstract control on destruction.*
- class **SubQPushButton**  
*Subclass of QPushButton.*
- class **SubQSpinBox**  
*Subclass of QSpinBox.*
- class **SubQWidget**  
*Subclass of QWidget.*

### Namespaces

- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*



## 5.45 dynamicvectorelementcontrol.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicvectorelementcontrol.h>
#include <dtQt/dynamicsubwidgets.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <dtDAL/datatype.h>
#include <dtDAL/enginepropertytypes.h>
#include <dtUtil/log.h>
#include <QtCore/QLocale>
#include <QtGui/QWidget>
#include <QtGui/QDoubleValidator>
#include <sstream>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

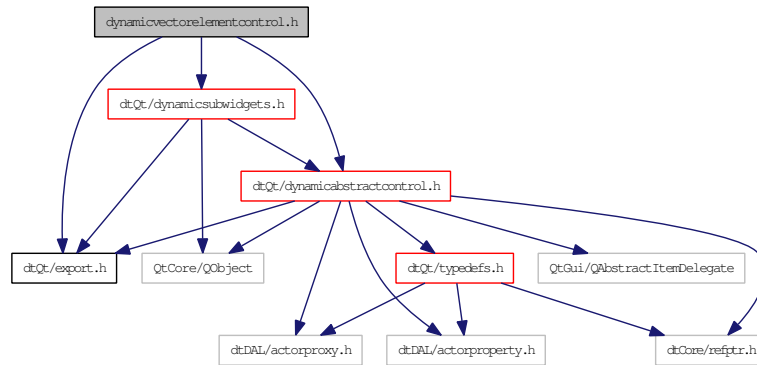
## 5.46 dynamicvectorelementcontrol.h File Reference

```
#include <dtQt/export.h>
```

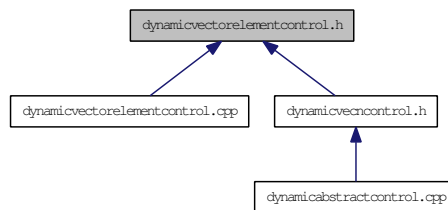
```
#include <dtQt/dynamicabstractcontrol.h>
```

```
#include <dtQt/dynamicsubwidgets.h>
```

Include dependency graph for dynamicvectorelementcontrol.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **DynamicVectorElementControl**

*This is the a sub control used by the various vector property classes.*

### Namespaces

- namespace **dtDAL**
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.47 export.h File Reference

### Defines

- #define DT\_QT\_EXPORT

#### 5.47.1 Define Documentation

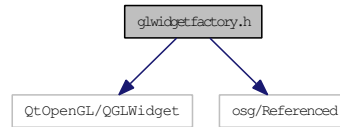
##### 5.47.1.1 #define DT\_QT\_EXPORT

## 5.48 glwidgetfactory.h File Reference

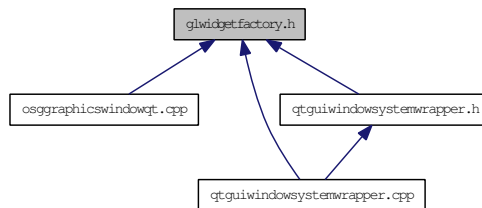
```
#include <QtOpenGL/QGLWidget>
```

```
#include <osg/Referenced>
```

Include dependency graph for glwidgetfactory.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **GLWidgetFactory**

*Abstract factory interface used to create specialized **OSGAdapterWidget** (p. 94) for custom application requirements.*

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.49 librarypathseditor.cpp File Reference

```
#include <QtGui/QVBoxLayout>
#include <QtGui/QHBoxLayout>
#include <QtGui/QGridLayout>
#include <QtGui/QPushButton>
#include <QtGui/QMessageBox>
#include <QtGui/QFileDialog>
#include <QtGui/QListWidgetItem>
#include <QtGui/QListWidget>
#include <QtGui/QGroupBox>
#include <dtQt/librarypathseditor.h>
#include <dtUtil/librarysharingmanager.h>
#include <cassert>
```

### Namespaces

- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

### Enumerations

- enum { **ERROR\_LIB\_NOT\_LOADED** = 0, **ERROR\_ACTORS\_IN\_LIB**, **ERROR\_INVALID\_LIB**, **ERROR\_UNKNOWN** }

#### 5.49.1 Enumeration Type Documentation

##### 5.49.1.1 anonymous enum

Enumerator:

```
ERROR_LIB_NOT_LOADED
ERROR_ACTORS_IN_LIB
ERROR_INVALID_LIB
ERROR_UNKNOWN
```

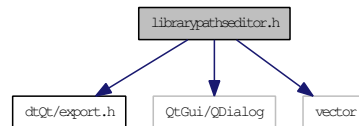
## 5.50 librarypathseditor.h File Reference

```
#include <dtQt/export.h>
```

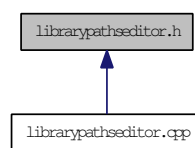
```
#include <QtGui/QDialog>
```

```
#include <vector>
```

Include dependency graph for librarypathseditor.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **LibraryPathsEditor**

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.51 mainpage.h File Reference

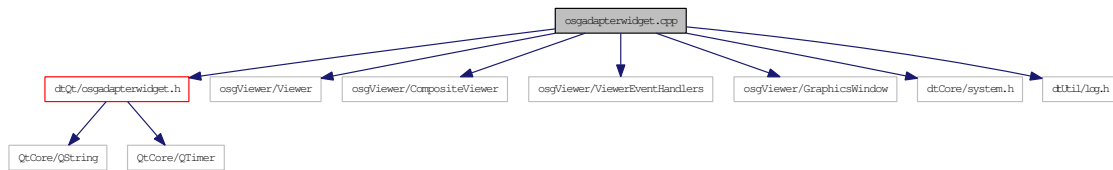
### 5.51.1 Detailed Description

This file contains Doxygen special commands and text for the **Main Page** (p. ??) and some other minor aspects of this documentation. It is not part of Delta3D.

## 5.52 osgadapterwidget.cpp File Reference

```
#include <dtQt/osgadapterwidget.h>
#include <osgViewer/Viewer>
#include <osgViewer/CompositeViewer>
#include <osgViewer/ViewerEventHandlers>
#include <osgViewer/GraphicsWindow>
#include <dtCore/system.h>
#include <dtUtil/log.h>
```

Include dependency graph for osgadapterwidget.cpp:



### Classes

- class **QtKeyboardMap**

### Namespaces

- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

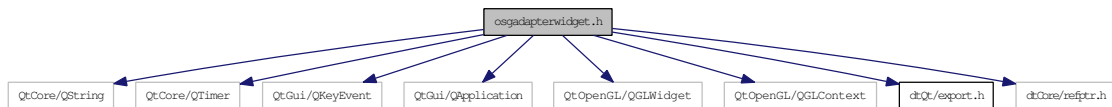
### Variables

- static **QtKeyboardMap** **STATIC\_KEY\_MAP**

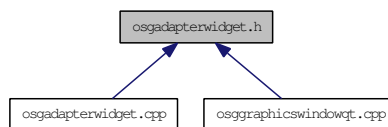
## 5.53 osgadapterwidget.h File Reference

```
#include <QtCore/QString>
#include <QtCore/Qtimer>
#include <QtGui/QKeyEvent>
#include <QtGui/QApplication>
#include <QtOpenGL/QGLWidget>
#include <QtOpenGL/QGLContext>
#include <dtQt/export.h>
#include <dtCore/refptr.h>
```

Include dependency graph for osgadapterwidget.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **OSGAdapterWidget**

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.54 osggraphicswindowqt.cpp File Reference

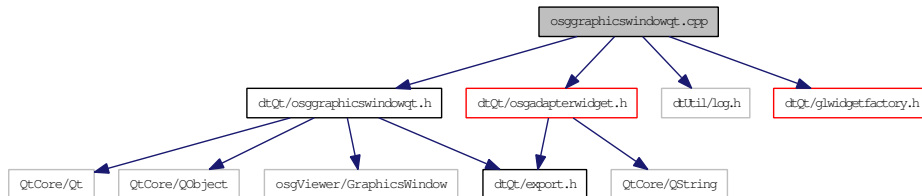
```
#include <dtQt/osggraphicswindowqt.h>
```

```
#include <dtQt/osgadapterwidget.h>
```

```
#include <dtUtil/log.h>
```

```
#include <dtQt/glwidgetfactory.h>
```

Include dependency graph for osggraphicswindowqt.cpp:



### Namespaces

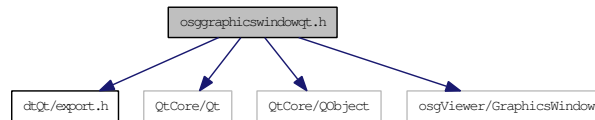
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

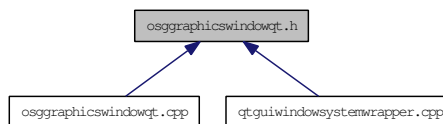
## 5.55 osggraphicswindowqt.h File Reference

```
#include <dtQt/export.h>
#include <QtCore/Qt>
#include <QtCore/QObject>
#include <osgViewer/GraphicsWindow>
```

Include dependency graph for osggraphicswindowqt.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **OSGGraphicsWindowQt**

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.56 projectcontextdialog.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/projectcontextdialog.h>
#include <QtGui/QVBoxLayout>
#include <QtGui/QHBoxLayout>
#include <QtGui/QPushButton>
#include <QtGui/QGroupBox>
#include <QtGui/QGridLayout>
#include <QtGui/QLineEdit>
#include <QtGui/QFileDialog>
#include <QtGui/QLabel>
#include <dtDAL/project.h>
```

### Namespaces

- namespace **dtQt**

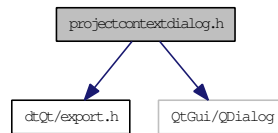
*Contains classes that help integrate Delta3D with Qt.*

## 5.57 projectcontextdialog.h File Reference

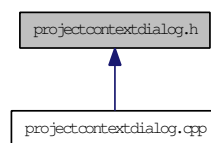
```
#include <dtQt/export.h>
```

```
#include <QtGui/QDialog>
```

Include dependency graph for projectcontextdialog.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **ProjectContextDialog**

### Namespaces

- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

## 5.58 propertyeditordelegate.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/propertyeditormodel.h>
#include <dtQt/propertyeditordelegate.h>
#include <dtQt/dynamicabstractcontrol.h>
#include <QtGui/QPainter>
#include <QtGui/QFrame>
#include <QtGui/QKeyEvent>
#include <QtGui/QApplication>
#include <QtGui/QSpinBox>
#include <QtGui/QToolButton>
#include <QtGui/QHBoxLayout>
#include <QtGui/QMessageBox>
#include <QtGui/QLabel>
#include <QtGui/QComboBox>
#include <QtGui/QPushButton>
#include <QtGui/qdrawutil.h>
#include <QtCore/qdebug.h>
#include <limits.h>
```

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

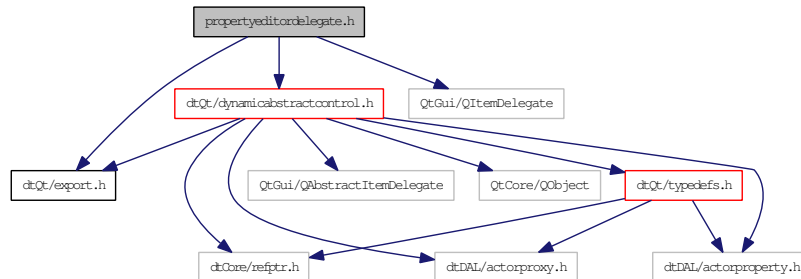
## 5.59 propertyeditordelegate.h File Reference

```
#include <dtQt/export.h>
```

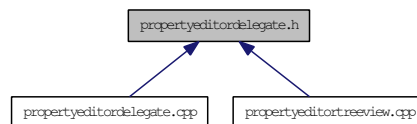
```
#include <QtGui/QItemDelegate>
```

```
#include <dtQt/dynamicabstractcontrol.h>
```

Include dependency graph for propertyeditordelegate.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **PropertyEditorDelegate**

*This class is a delegate for editing commands between the model and the dynamic control.*

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

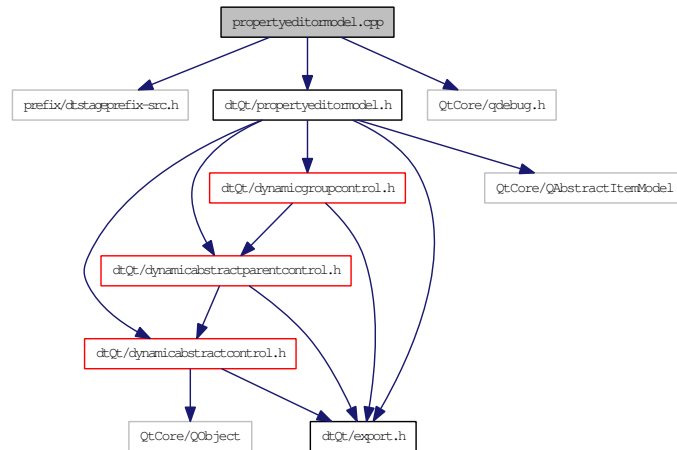
## 5.60 propertyeditormodel.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
```

```
#include <dtQt/propertyeditormodel.h>
```

```
#include <QtCore/qdebug.h>
```

Include dependency graph for propertyeditormodel.cpp:



### Namespaces

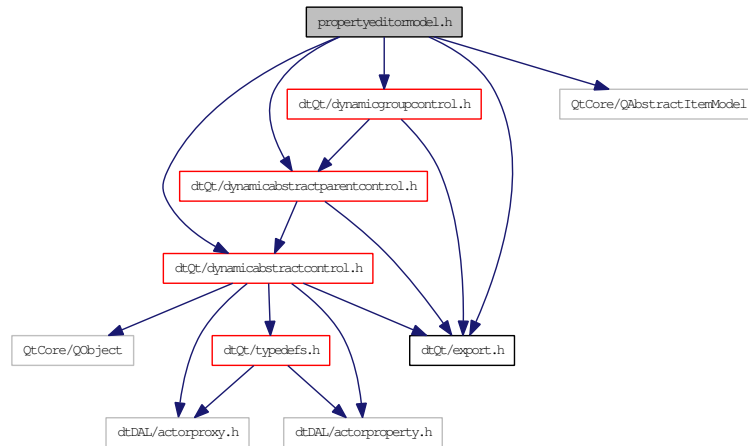
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.61 propertyeditormodel.h File Reference

```
#include <dtQt/export.h>
#include <dtQt/dynamicabstractcontrol.h>
#include <dtQt/dynamicabstractparentcontrol.h>
#include <dtQt/dynamicgroupcontrol.h>
#include <QtCore/QAbstractItemModel>
```

Include dependency graph for propertyeditormodel.h:



### Classes

- class **PropertyEditorModel**

*This class is the model used to represent the properties for a selected object.*

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.62 propertyeditortreeview.cpp File Reference

```
#include <prefix/dtstageprefix-src.h>
#include <dtQt/dynamicabstractcontrol.h>
#include <dtQt/propertyeditormodel.h>
#include <dtQt/propertyeditortreeview.h>
#include <dtQt/propertyeditordelegate.h>
#include <QtCore/qdebug.h>
#include <QtGui/QItemDelegate>
#include <QtGui/qheaderview.h>
#include <QtGui/qpainter.h>
#include <QtGui/qscrollbar.h>
#include <QtCore/QAbstractItemModel>
```

### Namespaces

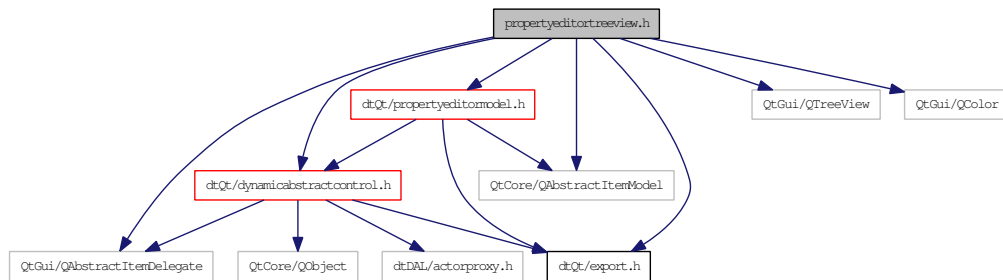
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

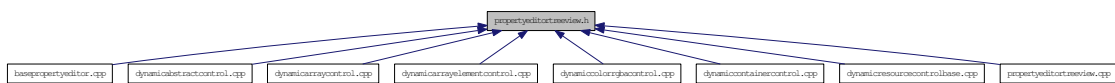
## 5.63 propertyeditortreeview.h File Reference

```
#include <dtQt/export.h>
#include <dtQt/dynamicabstractcontrol.h>
#include <dtQt/propertyeditormodel.h>
#include <QtCore/QAbstractItemModel>
#include <QtGui/QAbstractItemDelegate>
#include <QtGui/QTreeView>
#include <QtGui/QColor>
```

Include dependency graph for propertyeditortreeview.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **PropertyEditorTreeView**

*This class is the tree control for the properties of a proxy.*

### Namespaces

- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

## 5.64 qtguiwindowssystemwrapper.cpp File Reference

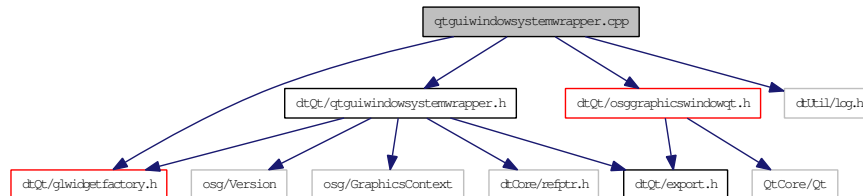
```
#include <dtQt/qtguiwindowssystemwrapper.h>
```

```
#include <dtQt/osggraphicswindowqt.h>
```

```
#include <dtQt/glwidgetfactory.h>
```

```
#include <dtUtil/log.h>
```

Include dependency graph for qtguiwindowssystemwrapper.cpp:



### Namespaces

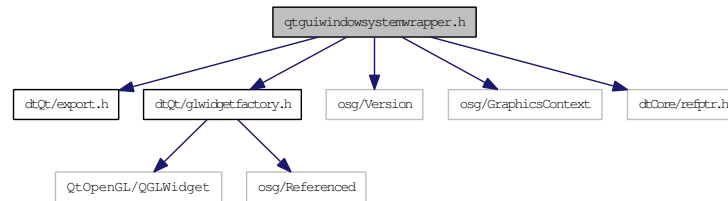
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

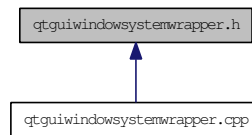
## 5.65 QtGuiWindowSystemWrapper.h File Reference

```
#include <dtQt/export.h>
#include <dtQt/glwidgetfactory.h>
#include <osg/Version>
#include <osg/GraphicsContext>
#include <dtCore/refptr.h>
```

Include dependency graph for QtGuiWindowSystemWrapper.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **QtGuiWindowSystemWrapper**

### Namespaces

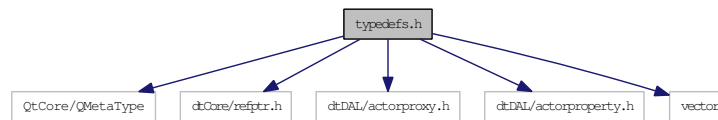
- namespace **dtQt**

*Contains classes that help integrate Delta3D with Qt.*

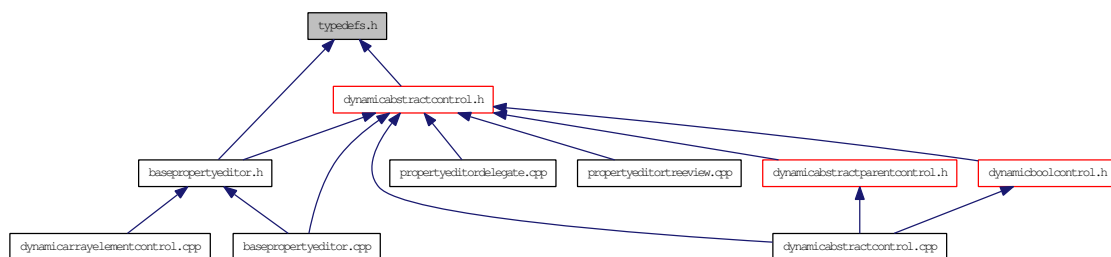
## 5.66 typedefs.h File Reference

```
#include <QtCore/QMetaType>
#include <dtCore/refptr.h>
#include <dtDAL/actorproxy.h>
#include <dtDAL/actorproperty.h>
#include <vector>
```

Include dependency graph for typedefs.h:



This graph shows which files directly or indirectly include this file:



### Typedefs

- typedef dtCore::RefPtr< dtDAL::ActorProperty > **ActorPropertyRefPtr**
- typedef dtCore::RefPtr< dtDAL::ActorProxy > **ActorProxyRefPtr**
- typedef std::vector< **ActorProxyRefPtr** > **ActorProxyRefPtrVector**

### Functions

- **Q\_DECLARE\_METATYPE** (dtCore::RefPtr< dtDAL::ActorProperty >)
- **Q\_DECLARE\_METATYPE** (dtCore::RefPtr< dtDAL::ActorProxy >)

#### 5.66.1 Typedef Documentation

5.66.1.1 typedef dtCore::RefPtr<dtDAL::ActorProperty> **ActorPropertyRefPtr**

5.66.1.2 typedef dtCore::RefPtr<dtDAL::ActorProxy> **ActorProxyRefPtr**

5.66.1.3 typedef std::vector<ActorProxyRefPtr> **ActorProxyRefPtrVector**

#### 5.66.2 Function Documentation

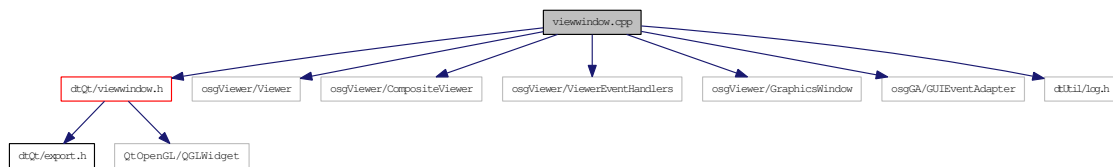
5.66.2.1 **Q\_DECLARE\_METATYPE** (dtCore::RefPtr< dtDAL::ActorProperty >)

5.66.2.2 **Q\_DECLARE\_METATYPE** (dtCore::RefPtr< dtDAL::ActorProxy >)

## 5.67 viewwindow.cpp File Reference

```
#include <dtQt/viewwindow.h>
#include <osgViewer/Viewer>
#include <osgViewer/CompositeViewer>
#include <osgViewer/ViewerEventHandlers>
#include <osgViewer/GraphicsWindow>
#include <osgGA/GUIEventAdapter>
#include <dtUtil/log.h>
```

Include dependency graph for viewwindow.cpp:



### Classes

- class **QtKeyboardMap**

### Variables

- static **QtKeyboardMap** **STATIC\_KEY\_MAP**

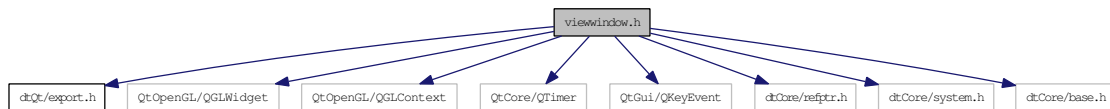
#### 5.67.1 Variable Documentation

##### 5.67.1.1 QtKeyboardMap STATIC\_KEY\_MAP [static]

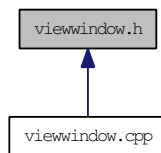
## 5.68 viewwindow.h File Reference

```
#include <dtQt/export.h>
#include <QtOpenGL/QGLWidget>
#include <QtOpenGL/QGLContext>
#include <QtCore/QTimer>
#include <QtGui/QKeyEvent>
#include <dtCore/refptr.h>
#include <dtCore/system.h>
#include <dtCore/base.h>
```

Include dependency graph for viewwindow.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class **ViewWindow**  
*Little class used to hold the Delta3D rendering surface.*

### Namespaces

- namespace **dtQt**  
*Contains classes that help integrate Delta3D with Qt.*

# Index

---

## - Symbols -

- ~BaseLibraryListEditor
  - dtQt::BaseLibraryListEditor, 14
- ~BasePropertyEditor
  - dtQt::BasePropertyEditor, 17
- ~DeltaStepper
  - dtQt::DeltaStepper, 18
- ~DialogListSelection
  - dtQt::DialogListSelection, 19
- ~DynamicAbstractControl
  - dtQt::DynamicAbstractControl, 23
- ~DynamicAbstractParentControl
  - dtQt::DynamicAbstractParentControl, 31
- ~DynamicArrayControl
  - dtQt::DynamicArrayControl, 34
- ~DynamicArrayElementControl
  - dtQt::DynamicArrayElementControl, 37
- ~DynamicBoolControl
  - dtQt::DynamicBoolControl, 41
- ~DynamicColorElementControl
  - dtQt::DynamicColorElementControl, 44
- ~DynamicColorRGBAControl
  - dtQt::DynamicColorRGBAControl, 48
- ~DynamicContainerControl
  - dtQt::DynamicContainerControl, 52
- ~DynamicDoubleControl
  - dtQt::DynamicDoubleControl, 56
- ~DynamicEnumControl
  - dtQt::DynamicEnumControl, 59
- ~DynamicFloatControl
  - dtQt::DynamicFloatControl, 62
- ~DynamicGroupControl
  - dtQt::DynamicGroupControl, 65
- ~DynamicIntControl
  - dtQt::DynamicIntControl, 67
- ~DynamicLabelControl
  - dtQt::DynamicLabelControl, 70
- ~DynamicLongControl
  - dtQt::DynamicLongControl, 73
- ~DynamicResourceControlBase
  - dtQt::DynamicResourceControlBase, 78
- ~DynamicStringControl
  - dtQt::DynamicStringControl, 82
- ~DynamicVecNControl
  - dtQt::DynamicVecNControl, 85
- ~DynamicVectorElementControl
  - dtQt::DynamicVectorElementControl, 88
- ~GLWidgetFactory
  - dtQt::GLWidgetFactory, 91
- ~LibraryPathsEditor
  - dtQt::LibraryPathsEditor, 92
- ~OSGAdapterWidget
  - dtQt::OSGAdapterWidget, 95
- ~OSGGraphicsWindowQt
  - dtQt::OSGGraphicsWindowQt, 97
- ~ProjectContextDialog
  - dtQt::ProjectContextDialog, 99
- ~PropertyEditorDelegate
  - dtQt::PropertyEditorDelegate, 100
- ~PropertyEditorModel

- dtQt::PropertyEditorModel, 103
- ~PropertyEditorTreeView
  - dtQt::PropertyEditorTreeView, 106
- ~QtGuiWindowSystemWrapper
  - dtQt::QtGuiWindowSystemWrapper, 107
- ~QtKeyboardMap
  - dtQt::QtKeyboardMap, 109
  - QtKeyboardMap, 108
- ~SubQComboBox
  - dtQt::SubQComboBox, 110
- ~SubQLabel
  - dtQt::SubQLabel, 112
- ~SubQLineEdit
  - dtQt::SubQLineEdit, 115
- ~SubQPushButton
  - dtQt::SubQPushButton, 116
- ~SubQSpinBox
  - dtQt::SubQSpinBox, 118
- ~SubQWidget
  - dtQt::SubQWidget, 120
- ~ViewWindow
  - dtQt::ViewWindow, 123

## - A -

- ActorPropertyChanged
  - dtQt::BasePropertyEditor, 17
- actorPropertyChanged
  - dtQt::DynamicAbstractControl, 24
  - dtQt::DynamicBoolControl, 41
  - dtQt::DynamicColorElementControl, 44
  - dtQt::DynamicColorRGBAControl, 48
  - dtQt::DynamicDoubleControl, 56
  - dtQt::DynamicEnumControl, 59
  - dtQt::DynamicFloatControl, 62
  - dtQt::DynamicIntControl, 67
  - dtQt::DynamicLongControl, 73
  - dtQt::DynamicResourceControlBase, 79
  - dtQt::DynamicStringControl, 82
  - dtQt::DynamicVectorElementControl, 88
- ActorPropertyRefPtr
  - typedefs.h, 190
- ActorProxyRefPtr
  - typedefs.h, 190
- ActorProxyRefPtrVector
  - typedefs.h, 190
- addChildControl
  - dtQt::DynamicGroupControl, 65
- addManagedChild
  - dtQt::SubQWidget, 121
- addSelfToParentWidget
  - dtQt::DynamicColorRGBAControl, 48
  - dtQt::DynamicGroupControl, 65
- AnyItemsSelected
  - dtQt::LibraryPathsEditor, 93

## - B -

- BaseClass
  - dtQt::OSGGraphicsWindowQt, 97
- BaseLibraryListEditor
  - dtQt::BaseLibraryListEditor, 14

- baselibrarylisteditor.cpp, 125
- baselibrarylisteditor.h, 126
- BasePropertyEditor
  - dtQt::BasePropertyEditor, 17
- basepropertyeditor.cpp, 127
- basepropertyeditor.h, 128
- buildDynamicControls
  - dtQt::BasePropertyEditor, 17
- C -**
- checkEvents
  - dtQt::OSGGraphicsWindowQt, 97
- className
  - dtQt::OSGGraphicsWindowQt, 97
- clearPressed
  - dtQt::DynamicResourceControlBase, 79
- closeEditor
  - dtQt::PropertyEditorTreeView, 106
- closeImplementation
  - dtQt::OSGGraphicsWindowQt, 97
- colorPickerPressed
  - dtQt::DynamicColorRGBAControl, 48
- columnCount
  - dtQt::PropertyEditorModel, 103
- CommitCurrentEdits
  - dtQt::BasePropertyEditor, 17
- convertColorFloatToInt
  - dtQt::DynamicColorElementControl, 44
- convertColorIntToFloat
  - dtQt::DynamicColorElementControl, 45
- CreateDynamicControl
  - dtQt::DynamicControlFactory, 54
- createEditor
  - dtQt::DynamicAbstractControl, 24
  - dtQt::DynamicArrayControl, 34
  - dtQt::DynamicArrayElementControl, 38
  - dtQt::DynamicBoolControl, 41
  - dtQt::DynamicColorElementControl, 45
  - dtQt::DynamicColorRGBAControl, 48
  - dtQt::DynamicDoubleControl, 56
  - dtQt::DynamicEnumControl, 59
  - dtQt::DynamicFloatControl, 62
  - dtQt::DynamicIntControl, 67
  - dtQt::DynamicLongControl, 73
  - dtQt::DynamicResourceControlBase, 79
  - dtQt::DynamicStringControl, 82
  - dtQt::DynamicVectorElementControl, 89
  - dtQt::PropertyEditorDelegate, 101
- CreateElementControl
  - dtQt::DynamicColorRGBAControl, 49
  - dtQt::DynamicVecNControl, 85
- createGraphicsContext
  - dtQt::QtGuiWindowSystemWrapper, 107
- CreateWidget
  - dtQt::GLWidgetFactory, 91
- currentChanged
  - dtQt::PropertyEditorTreeView, 106
- D -**
- data
  - dtQt::PropertyEditorModel, 103
- DeltaStepper
  - dtQt::DeltaStepper, 18
- deltastepper.cpp, 129
- deltastepper.h, 130
- DialogListSelection
  - dtQt::DialogListSelection, 19
- dialoglistselection.cpp, 131
- dialoglistselection.h, 132
- drawDecoration
  - dtQt::PropertyEditorDelegate, 101
- DT\_QT\_EXPORT
  - export.h, 171
- dtDAL, 9
- dtQt, 10
  - NUM\_DECIMAL\_DIGITS\_DOUBLE, 12
  - NUM\_DECIMAL\_DIGITS\_FLOAT, 12
  - STATIC\_KEY\_MAP, 12
- dtQt::BaseLibraryListEditor, 13
  - ~BaseLibraryListEditor, 14
  - BaseLibraryListEditor, 14
  - EnableButtons, 14
  - ERROR\_INVALID\_LIB, 14
  - ERROR\_LIB\_NOT\_LOADED, 14
  - ERROR\_OBJECTS\_IN\_LIB\_EXIST, 14
  - ERROR\_UNKNOWN, 14
  - Errors, 14
  - GetLibraryListWidget, 14
  - GetLibraryNames, 14
  - HandleFailure, 14
  - RefreshLibraries, 14
  - SelectLibraryToOpen, 14
  - ShiftLibraryDown, 14
  - ShiftLibraryUp, 14
  - showEvent, 14
  - SpawnDeleteConfirmation, 14
  - SpawnFileBrowser, 14
- dtQt::BasePropertyEditor, 16
  - ~BasePropertyEditor, 17
  - ActorPropertyChanged, 17
  - BasePropertyEditor, 17
  - buildDynamicControls, 17
  - CommitCurrentEdits, 17
  - GetDynamicControlFactory, 17
  - GetGroupBoxLabelText, 17
  - GetPropertyEditorModel, 17
  - GetRootControl, 17
  - GetSelectedPropertyContainers, 17
  - HandlePropertyContainersSelected, 17
  - PropertyAboutToChangeFromControl, 17
  - PropertyChangedFromControl, 17
  - PropertyContainerRefPtrVector, 17
  - ProxyNameChanged, 17
- dtQt::DeltaStepper, 18
  - ~DeltaStepper, 18
  - DeltaStepper, 18
  - Start, 18
  - Stop, 18
  - Tick, 18
- dtQt::DialogListSelection, 19
  - ~DialogListSelection, 19
  - DialogListSelection, 19
  - GetSelectedItem, 19
  - onCurrentRowChanged, 19
  - onItemClicked, 20
  - onItemDoubleClicked, 20
  - onSelectionChanged, 20
  - SetListItems, 20
- dtQt::DynamicAbstractControl, 21
  - ~DynamicAbstractControl, 23
  - actorPropertyChanged, 24

- createEditor, 24
- DynamicAbstractControl, 23
- getChild, 24
- getChildCount, 24
- getChildIndex, 24
- getDescription, 24
- getDisplayName, 24
- GetDynamicControlFactory, 25
- GetModel, 25
- getParent, 25
- getValueAsString, 25
- handleSubEditDestroy, 25
- InitializeData, 25
- InstallEventFilterOnControl, 25
- isControlDoesCustomPainting, 26
- isEditable, 26
- mInitialized, 28
- mModel, 28
- mParent, 28
- mPropContainer, 28
- mPropertyTree, 28
- NeedsPersistentEditor, 26
- NotifyParentOfPreUpdate, 26
- OnChildPreUpdate, 26
- paintColumn, 26
- PropertyAboutToChange, 26
- PropertyAboutToChangePassThrough, 26
- PropertyChanged, 26
- PropertyChangedPassThrough, 27
- SetBackgroundColor, 27
- SetDynamicControlFactory, 27
- SetTreeView, 27
- sizeHint, 27
- updateData, 27
- updateEditorFromModel, 27
- updateModelFromEditor, 27
- dtQt::DynamicAbstractParentControl, 29
  - ~DynamicAbstractParentControl, 31
  - DynamicAbstractParentControl, 31
  - getChild, 31
  - getChildCount, 31
  - getChildIndex, 31
  - mChildren, 31
  - removeAllChildren, 31
- dtQt::DynamicArrayControl, 32
  - ~DynamicArrayControl, 34
  - createEditor, 34
  - DynamicArrayControl, 34
  - getDescription, 34
  - getDisplayName, 34
  - getValueAsString, 34
  - handleSubEditDestroy, 34
  - InitializeData, 34
  - isEditable, 34
  - onAddClicked, 35
  - onClearClicked, 35
  - resizeChildren, 35
  - updateData, 35
- dtQt::DynamicArrayElementControl, 36
  - ~DynamicArrayElementControl, 37
  - createEditor, 38
  - DynamicArrayElementControl, 37
  - getDescription, 38
  - getDisplayName, 38
  - GetIndex, 38
  - getValueAsString, 38
  - handleSubEditDestroy, 38
  - InitializeData, 38
  - isEditable, 38
  - OnChildPreUpdate, 38
  - onCopyClicked, 38
  - onDeleteClicked, 38
  - onShiftDownClicked, 38
  - onShiftUpClicked, 38
  - SetActive, 38
  - SetIndex, 38
  - updateData, 39
- dtQt::DynamicBoolControl, 40
  - ~DynamicBoolControl, 41
  - actorPropertyChanged, 41
  - createEditor, 41
  - DynamicBoolControl, 41
  - FALSE\_LABEL, 42
  - getDescription, 41
  - getDisplayName, 41
  - getValueAsString, 42
  - handleSubEditDestroy, 42
  - InitializeData, 42
  - isEditable, 42
  - itemSelected, 42
  - TRUE\_LABEL, 42
  - updateData, 42
  - updateEditorFromModel, 42
  - updateModelFromEditor, 42
- dtQt::DynamicColorElementControl, 43
  - ~DynamicColorElementControl, 44
  - actorPropertyChanged, 44
  - convertColorFloatToInt, 44
  - convertColorIntToFloat, 45
  - createEditor, 45
  - DynamicColorElementControl, 44
  - getDescription, 45
  - getDisplayName, 45
  - getValue, 45
  - getValueAsString, 45
  - handleSubEditDestroy, 45
  - InitializeData, 45
  - isEditable, 45
  - setValue, 45
  - updateData, 45
  - updateEditorFromModel, 45
  - updateModelFromEditor, 45
- dtQt::DynamicColorRGBAControl, 47
  - ~DynamicColorRGBAControl, 48
  - actorPropertyChanged, 48
  - addSelfToParentWidget, 48
  - colorPickerPressed, 48
  - createEditor, 48
  - CreateElementControl, 49
  - DynamicColorRGBAControl, 48
  - getDescription, 49
  - getDisplayName, 49
  - getValueAsString, 49
  - handleSubEditDestroy, 49
  - InitializeData, 49
  - installEventFilterOnControl, 49
  - isEditable, 49
  - NeedsPersistentEditor, 49
  - updateData, 49
  - updateEditorFromModel, 49
  - updateModelFromEditor, 49
- dtQt::DynamicContainerControl, 51

- ~DynamicContainerControl, 52
- DynamicContainerControl, 52
- getDescription, 52
- getDisplayName, 52
- getValueAsString, 52
- initializeData, 52
- isEditable, 52
- updateData, 52
- dtQt::DynamicControlFactory, 54
  - CreateDynamicControl, 54
  - DynamicControlFactory, 54
  - RegisterControlForDataType, 54
- dtQt::DynamicDoubleControl, 55
  - ~DynamicDoubleControl, 56
  - actorPropertyChanged, 56
  - createEditor, 56
  - DynamicDoubleControl, 56
  - getDescription, 56
  - getDisplayName, 56
  - getValueAsString, 56
  - handleSubEditDestroy, 56
  - initializeData, 57
  - isEditable, 57
  - updateData, 57
  - updateEditorFromModel, 57
  - updateModelFromEditor, 57
- dtQt::DynamicEnumControl, 58
  - ~DynamicEnumControl, 59
  - actorPropertyChanged, 59
  - createEditor, 59
  - DynamicEnumControl, 59
  - getDescription, 59
  - getDisplayName, 59
  - getValueAsString, 59
  - handleSubEditDestroy, 60
  - initializeData, 60
  - isEditable, 60
  - itemSelected, 60
  - updateData, 60
  - updateEditorFromModel, 60
  - updateModelFromEditor, 60
- dtQt::DynamicFloatControl, 61
  - ~DynamicFloatControl, 62
  - actorPropertyChanged, 62
  - createEditor, 62
  - DynamicFloatControl, 62
  - getDescription, 62
  - getDisplayName, 62
  - getValueAsString, 62
  - handleSubEditDestroy, 62
  - initializeData, 63
  - isEditable, 63
  - updateData, 63
  - updateEditorFromModel, 63
  - updateModelFromEditor, 63
- dtQt::DynamicGroupControl, 64
  - ~DynamicGroupControl, 65
  - addChildControl, 65
  - addSelfToParentWidget, 65
  - DynamicGroupControl, 65
  - getChildGroupControl, 65
  - getDisplayName, 65
  - updateData, 65
- dtQt::DynamicIntControl, 66
  - ~DynamicIntControl, 67
  - actorPropertyChanged, 67
  - createEditor, 67
  - DynamicIntControl, 67
  - getDescription, 67
  - getDisplayName, 67
  - getValueAsString, 67
  - handleSubEditDestroy, 67
  - initializeData, 68
  - isEditable, 68
  - updateData, 68
  - updateEditorFromModel, 68
  - updateModelFromEditor, 68
- dtQt::DynamicLabelControl, 69
  - ~DynamicLabelControl, 70
  - DynamicLabelControl, 70
  - getDescription, 70
  - getDisplayName, 70
  - getValueAsString, 70
  - initializeData, 70
  - setDisplayValues, 70
  - updateData, 70
- dtQt::DynamicLongControl, 72
  - ~DynamicLongControl, 73
  - actorPropertyChanged, 73
  - createEditor, 73
  - DynamicLongControl, 73
  - getDescription, 73
  - getDisplayName, 73
  - getValueAsString, 73
  - handleSubEditDestroy, 73
  - initializeData, 74
  - isEditable, 74
  - updateData, 74
  - updateEditorFromModel, 74
  - updateModelFromEditor, 74
- dtQt::DynamicNoUpdateParentControl, 75
  - updateData, 76
- dtQt::DynamicResourceControlBase, 77
  - ~DynamicResourceControlBase, 78
  - actorPropertyChanged, 79
  - clearPressed, 79
  - createEditor, 79
  - DynamicResourceControlBase, 78
  - getCurrentResource, 79
  - getDescription, 79
  - getDisplayName, 79
  - GetProperty, 79
  - getValueAsString, 79
  - handleSubEditDestroy, 79
  - initializeData, 79
  - installEventFilterOnControl, 79
  - isEditable, 79
  - updateData, 79
  - updateEditorFromModel, 80
  - updateModelFromEditor, 80
  - useCurrentPressed, 80
- dtQt::DynamicStringControl, 81
  - ~DynamicStringControl, 82
  - actorPropertyChanged, 82
  - createEditor, 82
  - DynamicStringControl, 82
  - getDescription, 82
  - getDisplayName, 82
  - getValueAsString, 82
  - handleSubEditDestroy, 82
  - initializeData, 83
  - isEditable, 83

- updateData, 83
- updateEditorFromModel, 83
- updateModelFromEditor, 83
- dtQt::DynamicVecNControl, 84
  - ~DynamicVecNControl, 85
  - CreateElementControl, 85
  - DynamicVecNControl, 85
  - getDescription, 85
  - getDisplayName, 85
  - getValueAsString, 85
  - initializeData, 85
  - isEditable, 85
  - PropertyGetValueType, 85
- dtQt::DynamicVectorElementControl, 86
  - ~DynamicVectorElementControl, 88
  - actorPropertyChanged, 88
  - createEditor, 89
  - DynamicVectorElementControl, 88
  - getDescription, 89
  - getDisplayName, 89
  - getValue, 89
  - getValueAsString, 89
  - handleSubEditDestroy, 89
  - initializeData, 89
  - isEditable, 89
  - setValue, 89
  - updateData, 89
  - updateEditorFromModel, 89
  - updateModelFromEditor, 89
- dtQt::GLWidgetFactory, 91
  - ~GLWidgetFactory, 91
  - CreateWidget, 91
  - GLWidgetFactory, 91
- dtQt::LibraryPathsEditor, 92
  - ~LibraryPathsEditor, 92
  - AnyItemsSelected, 93
  - LibraryPathsEditor, 92
  - RefreshButtons, 93
  - ShiftPathDown, 93
  - ShiftPathUp, 93
  - SpawnDeleteConfirmation, 93
  - SpawnFileBrowser, 93
- dtQt::OSGAdapterWidget, 94
  - ~OSGAdapterWidget, 95
  - GetGraphicsWindow, 95
  - initializeGL, 95
  - keyPressEvent, 95
  - keyReleaseEvent, 95
  - mDoResize, 95
  - mDrawOnSeparateThread, 95
  - mGraphicsWindow, 95
  - mouseMoveEvent, 95
  - mousePressEvent, 95
  - mouseReleaseEvent, 95
  - mThreadGLContext, 95
  - mTimer, 95
  - OSGAdapterWidget, 95
  - paintGL, 95
  - paintGLImpl, 95
  - resizeGL, 95
  - resizeGLImpl, 95
  - SetGraphicsWindow, 95
  - ThreadedInitializeGL, 95
  - ThreadedMakeCurrent, 95
  - ThreadedUpdateGL, 95
  - wheelEvent, 95
- dtQt::OSGGraphicsWindowQt, 96
  - ~OSGGraphicsWindowQt, 97
  - BaseClass, 97
  - checkEvents, 97
  - className, 97
  - closeImplementation, 97
  - GetQGLWidget, 97
  - getWindowRectangle, 97
  - grabFocus, 97
  - grabFocusIfPointerInWindow, 97
  - isRealizedImplementation, 97
  - isSameKindAs, 97
  - libraryName, 97
  - makeCurrentImplementation, 97
  - OSGGraphicsWindowQt, 97
  - realizeImplementation, 97
  - releaseContextImplementation, 97
  - requestClose, 97
  - resizedImplementation, 97
  - setCursor, 97
  - SetQGLWidget, 97
  - setWindowDecorationImplementation, 97
  - setWindowName, 97
  - setWindowRectangleImplementation, 98
  - swapBuffersImplementation, 98
  - useCursor, 98
  - valid, 98
- dtQt::ProjectContextDialog, 99
  - ~ProjectContextDialog, 99
  - getProjectPath, 99
  - ProjectContextDialog, 99
  - spawnFileBrowser, 99
- dtQt::PropertyEditorDelegate, 100
  - ~PropertyEditorDelegate, 100
  - createEditor, 101
  - drawDecoration, 101
  - eventFilter, 101
  - paint, 101
  - PropertyEditorDelegate, 100
  - setEditorData, 101
  - setModelData, 101
  - sizeHint, 101
- dtQt::PropertyEditorModel, 102
  - ~PropertyEditorModel, 103
  - columnCount, 103
  - data, 103
  - DynamicAbstractParentControl, 104
  - DynamicGroupControl, 104
  - flags, 103
  - GetAbstractControlFromIndex, 103
  - GetRootControl, 103
  - hasChildren, 103
  - headerData, 104
  - index, 104
  - IndexOf, 104
  - insertRows, 104
  - isEditable, 104
  - parent, 104
  - PropertyEditor, 104
  - PropertyEditorModel, 103
  - removeRows, 104
  - rowCount, 104
  - setData, 104
  - setDescription, 104
  - SetRootControl, 104
- dtQt::PropertyEditorTreeView, 105

- ~PropertyEditorTreeView, 106
- closeEditor, 106
- currentChanged, 106
- isReadOnly, 106
- PropertyEditorTreeView, 106
- reset, 106
- ROW\_COLOR\_EVEN, 106
- ROW\_COLOR\_ODD, 106
- selectionChanged, 106
- setRoot, 106
- dtQt::QtGuiWindowSystemWrapper, 107
  - ~QtGuiWindowSystemWrapper, 107
  - createGraphicsContext, 107
  - EnableQtGUIWrapper, 107
  - getNumScreens, 107
  - getScreenResolution, 107
  - QtGuiWindowSystemWrapper, 107
  - SetGLWidgetFactory, 107
  - setScreenRefreshRate, 107
  - setScreenResolution, 107
- dtQt::QtKeyboardMap, 109
  - ~QtKeyboardMap, 109
  - QtKeyboardMap, 109
  - remapKey, 109
- dtQt::SubQComboBox, 110
  - ~SubQComboBox, 110
  - SubQComboBox, 110
- dtQt::SubQLabel, 112
  - ~SubQLabel, 112
  - SubQLabel, 112
- dtQt::SubQLineEdit, 114
  - ~SubQLineEdit, 115
  - SubQLineEdit, 115
- dtQt::SubQPushButton, 116
  - ~SubQPushButton, 116
  - SubQPushButton, 116
- dtQt::SubQSpinBox, 118
  - ~SubQSpinBox, 118
  - SubQSpinBox, 118
- dtQt::SubQWidget, 120
  - ~SubQWidget, 120
  - addManagedChild, 121
  - SubQWidget, 120
- dtQt::ViewWindow, 122
  - ~ViewWindow, 123
  - GetGraphicsWindow, 123
  - glDraw, 123
  - initializeGL, 123
  - keyPressEvent, 123
  - keyReleaseEvent, 123
  - mDoResize, 123
  - mDrawOnSeparateThread, 123
  - mGraphicsWindow, 123
  - mouseMoveEvent, 123
  - mousePressEvent, 123
  - mouseReleaseEvent, 123
  - mThreadGLContext, 123
  - mTimer, 123
  - OnMessage, 123
  - paintGL, 123
  - paintGLImpl, 123
  - resizeGL, 123
  - resizeGLImpl, 123
  - SetGraphicsWindow, 123
  - ThreadedInitializeGL, 123
  - ThreadedMakeCurrent, 123
  - ThreadedUpdateGL, 123
  - ViewWindow, 123
  - wheelEvent, 123
- DynamicAbstractControl
  - dtQt::DynamicAbstractControl, 23
- dynamicabstractcontrol.cpp, 133
- dynamicabstractcontrol.h, 134
- DynamicAbstractParentControl
  - dtQt::DynamicAbstractParentControl, 31
  - dtQt::PropertyEditorModel, 104
- dynamicabstractparentcontrol.cpp, 135
- dynamicabstractparentcontrol.h, 136
- DynamicArrayControl
  - dtQt::DynamicArrayControl, 34
- dynamicarraycontrol.cpp, 137
- dynamicarraycontrol.h, 138
- DynamicArrayElementControl
  - dtQt::DynamicArrayElementControl, 37
- dynamicarrayelementcontrol.cpp, 139
- dynamicarrayelementcontrol.h, 140
- DynamicBoolControl
  - dtQt::DynamicBoolControl, 41
- dynamicboolcontrol.cpp, 141
- dynamicboolcontrol.h, 142
- DynamicColorElementControl
  - dtQt::DynamicColorElementControl, 44
- dynamiccolorelementcontrol.cpp, 143
- dynamiccolorelementcontrol.h, 144
- DynamicColorRGBAControl
  - dtQt::DynamicColorRGBAControl, 48
- dynamiccolorrrgbacontrol.cpp, 145
- dynamiccolorrrgbacontrol.h, 146
- DynamicContainerControl
  - dtQt::DynamicContainerControl, 52
- dynamiccontainercontrol.cpp, 147
- dynamiccontainercontrol.h, 148
- DynamicControlFactory
  - dtQt::DynamicControlFactory, 54
- DynamicDoubleControl
  - dtQt::DynamicDoubleControl, 56
- dynamicdoublecontrol.cpp, 149
- dynamicdoublecontrol.h, 150
- DynamicEnumControl
  - dtQt::DynamicEnumControl, 59
- dynamicenumcontrol.cpp, 151
- dynamicenumcontrol.h, 152
- DynamicFloatControl
  - dtQt::DynamicFloatControl, 62
- dynamicfloatcontrol.cpp, 153
- dynamicfloatcontrol.h, 154
- DynamicGroupControl
  - dtQt::DynamicGroupControl, 65
  - dtQt::PropertyEditorModel, 104
- dynamicgroupcontrol.cpp, 155
- dynamicgroupcontrol.h, 156
- DynamicIntControl
  - dtQt::DynamicIntControl, 67
- dynamicintcontrol.cpp, 157
- dynamicintcontrol.h, 158
- DynamicLabelControl
  - dtQt::DynamicLabelControl, 70
- dynamiclabelcontrol.cpp, 159
- dynamiclabelcontrol.h, 160
- DynamicLongControl
  - dtQt::DynamicLongControl, 73
- dynamiclongcontrol.cpp, 161

- dynamiclongcontrol.h, 162
- DynamicResourceControlBase
  - dtQt::DynamicResourceControlBase, 78
- dynamicresourcecontrolbase.cpp, 163
- dynamicresourcecontrolbase.h, 164
- DynamicStringControl
  - dtQt::DynamicStringControl, 82
- dynamicstringcontrol.cpp, 165
- dynamicstringcontrol.h, 166
- dynamicsubwidgets.h, 167
- DynamicVecNControl
  - dtQt::DynamicVecNControl, 85
- dynamicvecncontrol.h, 168
- DynamicVectorElementControl
  - dtQt::DynamicVectorElementControl, 88
- dynamicvectorelementcontrol.cpp, 169
- dynamicvectorelementcontrol.h, 170
- E -**
- EnableButtons
  - dtQt::BaseLibraryListEditor, 14
- EnableQtGUIWrapper
  - dtQt::QtGuiWindowSystemWrapper, 107
- ERROR\_ACTORS\_IN\_LIB
  - librarypathseditor.cpp, 173
- ERROR\_INVALID\_LIB
  - dtQt::BaseLibraryListEditor, 14
  - librarypathseditor.cpp, 173
- ERROR\_LIB\_NOT\_LOADED
  - dtQt::BaseLibraryListEditor, 14
  - librarypathseditor.cpp, 173
- ERROR\_OBJECTS\_IN\_LIB\_EXIST
  - dtQt::BaseLibraryListEditor, 14
- ERROR\_UNKNOWN
  - dtQt::BaseLibraryListEditor, 14
  - librarypathseditor.cpp, 173
- Errors
  - dtQt::BaseLibraryListEditor, 14
- eventFilter
  - dtQt::PropertyEditorDelegate, 101
- export.h, 171
  - DT\_QT\_EXPORT, 171
- F -**
- FALSE\_LABEL
  - dtQt::DynamicBoolControl, 42
- flags
  - dtQt::PropertyEditorModel, 103
- G -**
- GetAbstractControlFromIndex
  - dtQt::PropertyEditorModel, 103
- getChild
  - dtQt::DynamicAbstractControl, 24
  - dtQt::DynamicAbstractParentControl, 31
- getChildCount
  - dtQt::DynamicAbstractControl, 24
  - dtQt::DynamicAbstractParentControl, 31
- getChildGroupControl
  - dtQt::DynamicGroupControl, 65
- getChildIndex
  - dtQt::DynamicAbstractControl, 24
  - dtQt::DynamicAbstractParentControl, 31
- getCurrentResource
  - dtQt::DynamicResourceControlBase, 79
- getDescription
  - dtQt::DynamicAbstractControl, 24
  - dtQt::DynamicArrayControl, 34
  - dtQt::DynamicArrayElementControl, 38
  - dtQt::DynamicBoolControl, 41
  - dtQt::DynamicColorElementControl, 45
  - dtQt::DynamicColorRGBAControl, 49
  - dtQt::DynamicContainerControl, 52
  - dtQt::DynamicDoubleControl, 56
  - dtQt::DynamicEnumControl, 59
  - dtQt::DynamicFloatControl, 62
  - dtQt::DynamicIntControl, 67
  - dtQt::DynamicLabelControl, 70
  - dtQt::DynamicLongControl, 73
  - dtQt::DynamicResourceControlBase, 79
  - dtQt::DynamicStringControl, 82
  - dtQt::DynamicVecNControl, 85
  - dtQt::DynamicVectorElementControl, 89
- getDisplayName
  - dtQt::DynamicAbstractControl, 24
  - dtQt::DynamicArrayControl, 34
  - dtQt::DynamicArrayElementControl, 38
  - dtQt::DynamicBoolControl, 41
  - dtQt::DynamicColorElementControl, 45
  - dtQt::DynamicColorRGBAControl, 49
  - dtQt::DynamicContainerControl, 52
  - dtQt::DynamicDoubleControl, 56
  - dtQt::DynamicEnumControl, 59
  - dtQt::DynamicFloatControl, 62
  - dtQt::DynamicGroupControl, 65
  - dtQt::DynamicIntControl, 67
  - dtQt::DynamicLabelControl, 70
  - dtQt::DynamicLongControl, 73
  - dtQt::DynamicResourceControlBase, 79
  - dtQt::DynamicStringControl, 82
  - dtQt::DynamicVecNControl, 85
  - dtQt::DynamicVectorElementControl, 89
- GetDynamicControlFactory
  - dtQt::BasePropertyEditor, 17
  - dtQt::DynamicAbstractControl, 25
- GetGraphicsWindow
  - dtQt::OSGAdapterWidget, 95
  - dtQt::ViewWindow, 123
- GetGroupBoxLabelText
  - dtQt::BasePropertyEditor, 17
- GetIndex
  - dtQt::DynamicArrayElementControl, 38
- GetLibraryListWidget
  - dtQt::BaseLibraryListEditor, 14
- GetLibraryNames
  - dtQt::BaseLibraryListEditor, 14
- GetModel
  - dtQt::DynamicAbstractControl, 25
- getNumScreens
  - dtQt::QtGuiWindowSystemWrapper, 107
- getParent
  - dtQt::DynamicAbstractControl, 25
- getProjectPath
  - dtQt::ProjectContextDialog, 99
- GetProperty
  - dtQt::DynamicResourceControlBase, 79
- GetPropertyEditorModel
  - dtQt::BasePropertyEditor, 17
- GetQGLWidget
  - dtQt::OSGGraphicsWindowQt, 97
- GetRootControl

- dtQt::BasePropertyEditor, 17
- dtQt::PropertyEditorModel, 103
- getScreenResolution
  - dtQt::QtGuiWindowSystemWrapper, 107
- GetSelectedItem
  - dtQt::DialogListSelection, 19
- GetSelectedPropertyContainers
  - dtQt::BasePropertyEditor, 17
- getValue
  - dtQt::DynamicColorElementControl, 45
  - dtQt::DynamicVectorElementControl, 89
- getValueAsString
  - dtQt::DynamicAbstractControl, 25
  - dtQt::DynamicArrayControl, 34
  - dtQt::DynamicArrayElementControl, 38
  - dtQt::DynamicBoolControl, 42
  - dtQt::DynamicColorElementControl, 45
  - dtQt::DynamicColorRGBAControl, 49
  - dtQt::DynamicContainerControl, 52
  - dtQt::DynamicDoubleControl, 56
  - dtQt::DynamicEnumControl, 59
  - dtQt::DynamicFloatControl, 62
  - dtQt::DynamicIntControl, 67
  - dtQt::DynamicLabelControl, 70
  - dtQt::DynamicLongControl, 73
  - dtQt::DynamicResourceControlBase, 79
  - dtQt::DynamicStringControl, 82
  - dtQt::DynamicVecNControl, 85
  - dtQt::DynamicVectorElementControl, 89
- getWindowRectangle
  - dtQt::OSGGraphicsWindowQt, 97
- glDraw
  - dtQt::ViewWindow, 123
- GLWidgetFactory
  - dtQt::GLWidgetFactory, 91
- glwidgetfactory.h, 172
- grabFocus
  - dtQt::OSGGraphicsWindowQt, 97
- grabFocusIfPointerInWindow
  - dtQt::OSGGraphicsWindowQt, 97

**- H -**

- HandleFailure
  - dtQt::BaseLibraryListEditor, 14
- HandlePropertyContainersSelected
  - dtQt::BasePropertyEditor, 17
- handleSubEditDestroy
  - dtQt::DynamicAbstractControl, 25
  - dtQt::DynamicArrayControl, 34
  - dtQt::DynamicArrayElementControl, 38
  - dtQt::DynamicBoolControl, 42
  - dtQt::DynamicColorElementControl, 45
  - dtQt::DynamicColorRGBAControl, 49
  - dtQt::DynamicDoubleControl, 56
  - dtQt::DynamicEnumControl, 60
  - dtQt::DynamicFloatControl, 62
  - dtQt::DynamicIntControl, 67
  - dtQt::DynamicLongControl, 73
  - dtQt::DynamicResourceControlBase, 79
  - dtQt::DynamicStringControl, 82
  - dtQt::DynamicVectorElementControl, 89
- hasChildren
  - dtQt::PropertyEditorModel, 103
- headerData

- dtQt::PropertyEditorModel, 104

**- I -**

- inc/ Directory Reference, 6
- inc/dtQt/ Directory Reference, 3
- index
  - dtQt::PropertyEditorModel, 104
- IndexOf
  - dtQt::PropertyEditorModel, 104
- InitializeData
  - dtQt::DynamicAbstractControl, 25
  - dtQt::DynamicArrayControl, 34
  - dtQt::DynamicArrayElementControl, 38
  - dtQt::DynamicBoolControl, 42
  - dtQt::DynamicColorElementControl, 45
  - dtQt::DynamicColorRGBAControl, 49
  - dtQt::DynamicContainerControl, 52
  - dtQt::DynamicDoubleControl, 57
  - dtQt::DynamicEnumControl, 60
  - dtQt::DynamicFloatControl, 63
  - dtQt::DynamicIntControl, 68
  - dtQt::DynamicLabelControl, 70
  - dtQt::DynamicLongControl, 74
  - dtQt::DynamicResourceControlBase, 79
  - dtQt::DynamicStringControl, 83
  - dtQt::DynamicVecNControl, 85
  - dtQt::DynamicVectorElementControl, 89
- initializeGL
  - dtQt::OSGAdapterWidget, 95
  - dtQt::ViewWindow, 123
- insertRows
  - dtQt::PropertyEditorModel, 104
- InstallEventFilterOnControl
  - dtQt::DynamicAbstractControl, 25
- installEventFilterOnControl
  - dtQt::DynamicColorRGBAControl, 49
  - dtQt::DynamicResourceControlBase, 79
- isControlDoesCustomPainting
  - dtQt::DynamicAbstractControl, 26
- isEditable
  - dtQt::DynamicAbstractControl, 26
  - dtQt::DynamicArrayControl, 34
  - dtQt::DynamicArrayElementControl, 38
  - dtQt::DynamicBoolControl, 42
  - dtQt::DynamicColorElementControl, 45
  - dtQt::DynamicColorRGBAControl, 49
  - dtQt::DynamicContainerControl, 52
  - dtQt::DynamicDoubleControl, 57
  - dtQt::DynamicEnumControl, 60
  - dtQt::DynamicFloatControl, 63
  - dtQt::DynamicIntControl, 68
  - dtQt::DynamicLongControl, 74
  - dtQt::DynamicResourceControlBase, 79
  - dtQt::DynamicStringControl, 83
  - dtQt::DynamicVecNControl, 85
  - dtQt::DynamicVectorElementControl, 89
  - dtQt::PropertyEditorModel, 104
- isReadOnly
  - dtQt::PropertyEditorTreeView, 106
- isRealizedImplementation
  - dtQt::OSGGraphicsWindowQt, 97
- isSameKindAs
  - dtQt::OSGGraphicsWindowQt, 97
- itemSelected
  - dtQt::DynamicBoolControl, 42

dtQt::DynamicEnumControl, 60

## - K -

keyPressEvent  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 keyReleaseEvent  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123

## - L -

libraryName  
 dtQt::OSGGraphicsWindowQt, 97  
 LibraryPathsEditor  
 dtQt::LibraryPathsEditor, 92  
 librarypathseditor.cpp, 173  
 ERROR\_ACTORS\_IN\_LIB, 173  
 ERROR\_INVALID\_LIB, 173  
 ERROR\_LIB\_NOT\_LOADED, 173  
 ERROR\_UNKNOWN, 173  
 librarypathseditor.h, 174

## - M -

mainpage.h, 175  
 makeCurrentImplementation  
 dtQt::OSGGraphicsWindowQt, 97  
 mChildren  
 dtQt::DynamicAbstractParentControl, 31  
 mDoResize  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 mDrawOnSeparateThread  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 mGraphicsWindow  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 mInitialized  
 dtQt::DynamicAbstractControl, 28  
 mModel  
 dtQt::DynamicAbstractControl, 28  
 mouseMoveEvent  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 mousePressEvent  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 mouseReleaseEvent  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 mParent  
 dtQt::DynamicAbstractControl, 28  
 mPropContainer  
 dtQt::DynamicAbstractControl, 28  
 mPropertyTree  
 dtQt::DynamicAbstractControl, 28  
 mThreadGLContext  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 mTimer  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123

## - N -

NeedsPersistentEditor

dtQt::DynamicAbstractControl, 26  
 dtQt::DynamicColorRGBAControl, 49  
 NotifyParentOfPreUpdate  
 dtQt::DynamicAbstractControl, 26  
 NUM\_DECIMAL\_DIGITS\_DOUBLE  
 dtQt, 12  
 NUM\_DECIMAL\_DIGITS\_FLOAT  
 dtQt, 12

## - O -

onAddClicked  
 dtQt::DynamicArrayControl, 35  
 OnChildPreUpdate  
 dtQt::DynamicAbstractControl, 26  
 dtQt::DynamicArrayElementControl, 38  
 onClearClicked  
 dtQt::DynamicArrayControl, 35  
 onCopyClicked  
 dtQt::DynamicArrayElementControl, 38  
 onCurrentRowChanged  
 dtQt::DialogListSelection, 19  
 onDeleteClicked  
 dtQt::DynamicArrayElementControl, 38  
 onItemClicked  
 dtQt::DialogListSelection, 20  
 onItemDoubleClicked  
 dtQt::DialogListSelection, 20  
 OnMessage  
 dtQt::ViewWindow, 123  
 onSelectionChanged  
 dtQt::DialogListSelection, 20  
 onShiftDownClicked  
 dtQt::DynamicArrayElementControl, 38  
 onShiftUpClicked  
 dtQt::DynamicArrayElementControl, 38  
 OSGAdapterWidget  
 dtQt::OSGAdapterWidget, 95  
 osgadapterwidget.cpp, 176  
 osgadapterwidget.h, 177  
 OSGGraphicsWindowQt  
 dtQt::OSGGraphicsWindowQt, 97  
 osggraphicswindowqt.cpp, 178  
 osggraphicswindowqt.h, 179

## - P -

paint  
 dtQt::PropertyEditorDelegate, 101  
 paintColumn  
 dtQt::DynamicAbstractControl, 26  
 paintGL  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 paintGLImpl  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 parent  
 dtQt::PropertyEditorModel, 104  
 ProjectContextDialog  
 dtQt::ProjectContextDialog, 99  
 projectcontextdialog.cpp, 180  
 projectcontextdialog.h, 181  
 PropertyAboutToChange  
 dtQt::DynamicAbstractControl, 26  
 PropertyAboutToChangeFromControl  
 dtQt::BasePropertyEditor, 17

- PropertyAboutToChangePassThrough
    - dtQt::DynamicAbstractControl, 26
  - PropertyChanged
    - dtQt::DynamicAbstractControl, 26
  - PropertyChangedFromControl
    - dtQt::BasePropertyEditor, 17
  - PropertyChangedPassThrough
    - dtQt::DynamicAbstractControl, 27
  - PropertyContainerRefPtrVector
    - dtQt::BasePropertyEditor, 17
  - PropertyEditor
    - dtQt::PropertyEditorModel, 104
  - PropertyEditorDelegate
    - dtQt::PropertyEditorDelegate, 100
  - propertyeditordelegate.cpp, 182
  - propertyeditordelegate.h, 183
  - PropertyEditorModel
    - dtQt::PropertyEditorModel, 103
  - propertyeditormodel.cpp, 184
  - propertyeditormodel.h, 185
  - PropertyEditorTreeView
    - dtQt::PropertyEditorTreeView, 106
  - propertyeditortreeview.cpp, 186
  - propertyeditortreeview.h, 187
  - PropertyGetValueType
    - dtQt::DynamicVecNControl, 85
  - ProxyNameChanged
    - dtQt::BasePropertyEditor, 17
- Q -**
- Q\_DECLARE\_METATYPE
    - typedefs.h, 190
  - QtGuiWindowSystemWrapper
    - dtQt::QtGuiWindowSystemWrapper, 107
  - qtguiwindowssystemwrapper.cpp, 188
  - qtguiwindowssystemwrapper.h, 189
  - QtKeyboardMap, 108
    - ~QtKeyboardMap, 108
    - dtQt::QtKeyboardMap, 109
    - QtKeyboardMap, 108
    - remapKey, 108
- R -**
- realizeImplementation
    - dtQt::OSGGraphicsWindowQt, 97
  - RefreshButtons
    - dtQt::LibraryPathsEditor, 93
  - RefreshLibraries
    - dtQt::BaseLibraryListEditor, 14
  - RegisterControlForDataType
    - dtQt::DynamicControlFactory, 54
  - releaseContextImplementation
    - dtQt::OSGGraphicsWindowQt, 97
  - remapKey
    - dtQt::QtKeyboardMap, 109
    - QtKeyboardMap, 108
  - removeAllChildren
    - dtQt::DynamicAbstractParentControl, 31
  - removeRows
    - dtQt::PropertyEditorModel, 104
  - requestClose
    - dtQt::OSGGraphicsWindowQt, 97
  - reset
    - dtQt::PropertyEditorTreeView, 106
  - resizeChildren
    - dtQt::DynamicArrayControl, 35
  - resizedImplementation
    - dtQt::OSGGraphicsWindowQt, 97
  - resizeGL
    - dtQt::OSGAdapterWidget, 95
    - dtQt::ViewWindow, 123
  - resizeGLImpl
    - dtQt::OSGAdapterWidget, 95
    - dtQt::ViewWindow, 123
  - ROW\_COLOR\_EVEN
    - dtQt::PropertyEditorTreeView, 106
  - ROW\_COLOR\_ODD
    - dtQt::PropertyEditorTreeView, 106
  - rowCount
    - dtQt::PropertyEditorModel, 104
- S -**
- selectionChanged
    - dtQt::PropertyEditorTreeView, 106
  - SelectLibraryToOpen
    - dtQt::BaseLibraryListEditor, 14
  - SetActive
    - dtQt::DynamicArrayElementControl, 38
  - SetBackgroundColor
    - dtQt::DynamicAbstractControl, 27
  - setCursor
    - dtQt::OSGGraphicsWindowQt, 97
  - setData
    - dtQt::PropertyEditorModel, 104
  - setDescription
    - dtQt::PropertyEditorModel, 104
  - setDisplayValues
    - dtQt::DynamicLabelControl, 70
  - SetDynamicControlFactory
    - dtQt::DynamicAbstractControl, 27
  - setEditorData
    - dtQt::PropertyEditorDelegate, 101
  - SetGLWidgetFactory
    - dtQt::QtGuiWindowSystemWrapper, 107
  - SetGraphicsWindow
    - dtQt::OSGAdapterWidget, 95
    - dtQt::ViewWindow, 123
  - SetIndex
    - dtQt::DynamicArrayElementControl, 38
  - SetListItems
    - dtQt::DialogListSelection, 20
  - setModelData
    - dtQt::PropertyEditorDelegate, 101
  - SetQGLWidget
    - dtQt::OSGGraphicsWindowQt, 97
  - setRoot
    - dtQt::PropertyEditorTreeView, 106
  - SetRootControl
    - dtQt::PropertyEditorModel, 104
  - setScreenRefreshRate
    - dtQt::QtGuiWindowSystemWrapper, 107
  - setScreenResolution
    - dtQt::QtGuiWindowSystemWrapper, 107
  - SetTreeView
    - dtQt::DynamicAbstractControl, 27
  - setValue
    - dtQt::DynamicColorElementControl, 45
    - dtQt::DynamicVectorElementControl, 89
  - setWindowDecorationImplementation
    - dtQt::OSGGraphicsWindowQt, 97
  - setWindowName

dtQt::OSGGraphicsWindowQt, 97  
 setWindowRectangleImplementation  
 dtQt::OSGGraphicsWindowQt, 98  
 ShiftLibraryDown  
 dtQt::BaseLibraryListEditor, 14  
 ShiftLibraryUp  
 dtQt::BaseLibraryListEditor, 14  
 ShiftPathDown  
 dtQt::LibraryPathsEditor, 93  
 ShiftPathUp  
 dtQt::LibraryPathsEditor, 93  
 showEvent  
 dtQt::BaseLibraryListEditor, 14  
 sizeHint  
 dtQt::DynamicAbstractControl, 27  
 dtQt::PropertyEditorDelegate, 101  
 SpawnDeleteConfirmation  
 dtQt::BaseLibraryListEditor, 14  
 dtQt::LibraryPathsEditor, 93  
 SpawnFileBrowser  
 dtQt::BaseLibraryListEditor, 14  
 dtQt::LibraryPathsEditor, 93  
 spawnFileBrowser  
 dtQt::ProjectContextDialog, 99  
 src/ Directory Reference, 7  
 src/dtQt/ Directory Reference, 5  
 Start  
 dtQt::DeltaStepper, 18  
 STATIC\_KEY\_MAP  
 dtQt, 12  
 viewwindow.cpp, 191  
 Stop  
 dtQt::DeltaStepper, 18  
 SubQComboBox  
 dtQt::SubQComboBox, 110  
 SubQLabel  
 dtQt::SubQLabel, 112  
 SubQLineEdit  
 dtQt::SubQLineEdit, 115  
 SubQPushButton  
 dtQt::SubQPushButton, 116  
 SubQSpinBox  
 dtQt::SubQSpinBox, 118  
 SubQWidget  
 dtQt::SubQWidget, 120  
 swapBuffersImplementation  
 dtQt::OSGGraphicsWindowQt, 98

**- T -**

ThreadedInitializeGL  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 ThreadedMakeCurrent  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 ThreadedUpdateGL  
 dtQt::OSGAdapterWidget, 95  
 dtQt::ViewWindow, 123  
 Tick  
 dtQt::DeltaStepper, 18  
 TRUE\_LABEL  
 dtQt::DynamicBoolControl, 42  
 typedefs.h, 190  
 ActorPropertyRefPtr, 190  
 ActorProxyRefPtr, 190  
 ActorProxyRefPtrVector, 190

Q\_DECLARE\_METATYPE, 190

**- U -**

updateData  
 dtQt::DynamicAbstractControl, 27  
 dtQt::DynamicArrayControl, 35  
 dtQt::DynamicArrayElementControl, 39  
 dtQt::DynamicBoolControl, 42  
 dtQt::DynamicColorElementControl, 45  
 dtQt::DynamicColorRGBAControl, 49  
 dtQt::DynamicContainerControl, 52  
 dtQt::DynamicDoubleControl, 57  
 dtQt::DynamicEnumControl, 60  
 dtQt::DynamicFloatControl, 63  
 dtQt::DynamicGroupControl, 65  
 dtQt::DynamicIntControl, 68  
 dtQt::DynamicLabelControl, 70  
 dtQt::DynamicLongControl, 74  
 dtQt::DynamicNoUpdateParentControl, 76  
 dtQt::DynamicResourceControlBase, 79  
 dtQt::DynamicStringControl, 83  
 dtQt::DynamicVectorElementControl, 89  
 updateEditorFromModel  
 dtQt::DynamicAbstractControl, 27  
 dtQt::DynamicBoolControl, 42  
 dtQt::DynamicColorElementControl, 45  
 dtQt::DynamicColorRGBAControl, 49  
 dtQt::DynamicDoubleControl, 57  
 dtQt::DynamicEnumControl, 60  
 dtQt::DynamicFloatControl, 63  
 dtQt::DynamicIntControl, 68  
 dtQt::DynamicLongControl, 74  
 dtQt::DynamicResourceControlBase, 80  
 dtQt::DynamicStringControl, 83  
 dtQt::DynamicVectorElementControl, 89  
 updateModelFromEditor  
 dtQt::DynamicAbstractControl, 27  
 dtQt::DynamicBoolControl, 42  
 dtQt::DynamicColorElementControl, 45  
 dtQt::DynamicColorRGBAControl, 49  
 dtQt::DynamicDoubleControl, 57  
 dtQt::DynamicEnumControl, 60  
 dtQt::DynamicFloatControl, 63  
 dtQt::DynamicIntControl, 68  
 dtQt::DynamicLongControl, 74  
 dtQt::DynamicResourceControlBase, 80  
 dtQt::DynamicStringControl, 83  
 dtQt::DynamicVectorElementControl, 89  
 useCurrentPressed  
 dtQt::DynamicResourceControlBase, 80  
 useCursor  
 dtQt::OSGGraphicsWindowQt, 98

**- V -**

valid  
 dtQt::OSGGraphicsWindowQt, 98  
 ViewWindow  
 dtQt::ViewWindow, 123  
 viewwindow.cpp, 191  
 STATIC\_KEY\_MAP, 191  
 viewwindow.h, 192

**- W -**

wheelEvent  
 dtQt::OSGAdapterWidget, 95

dtQt::ViewWindow, 123