

Gizmos: a library of enhanced controls

by Julian Smart and others

January 5th 2002

Contents

Copyright notice	ii
Introduction	1
What is the Gizmos library?	1
Alphabetical class reference	2
wxDynamicSashSplitEvent	2
wxDynamicSashUnifyEvent	2
wxDynamicSashWindow	3
wxEditableListBox	4
wxLEDNumberCtrl	5
wxMultiCellCanvas	6
wxMultiCellItemHandle	7
wxMultiCellSizer	8
wxRemotelyScrolledTreeCtrl	9
wxSplitterScrolledWindow	11
wxThinSplitterWindow	12
wxTreeCompanionWindow	13
Classes by category	14
Topic overviews	15
Notes on using the reference	15
References	16
Index	18

Copyright notice

The licence is the wxWindows Licence.

Introduction

What is the Gizmos library?

This manual describes a class library with a miscellany of useful user interface classes.

Alphabetical class reference

wxDynamicSashSplitEvent

wxDynamicSashSplitEvents are sent to your view by wxDynamicSashWindow whenever your view is being split by the user. It is your responsibility to handle this event by creating a new view window as a child of the wxDynamicSashWindow. wxDynamicSashWindow will automatically reparent it to the proper place in its window hierarchy.

Derived from

wxCommandEvent

Data structures

wxDynamicSashSplitEvent::wxDynamicSashSplitEvent

wxDynamicSashSplitEvent(const wxDynamicSashSplitEvent& event)

wxDynamicSashSplitEvent(wxObject* target)

wxDynamicSashSplitEvent()

wxDynamicSashSplitEvent::Clone

wxEvent* Clone() const

wxDynamicSashUnifyEvent

wxDynamicSashUnifyEvents are sent to your view by wxDynamicSashWindow whenever the sash which splits your view and its sibling is being reunified such that your view is expanding to replace its sibling. You needn't do anything with this event if you are allowing wxDynamicSashWindow to manage your view's scrollbars, but it is useful if you are managing the scrollbars yourself so that you can keep the scrollbars' event handlers connected to your view's event handler class.

Derived from

wxCommandEvent

Data structures

wxDynamicSashUnifyEvent::wxDynamicSashUnifyEvent

wxDynamicSashUnifyEvent(const wxDynamicSashUnifyEvent& event)

wxDynamicSashUnifyEvent(wxObject* target)

wxDynamicSashUnifyEvent()

wxDynamicSashUnifyEvent::Clone

wxEvent* Clone() const

wxDynamicSashWindow

wxDynamicSashWindow. See above.

Derived from

wxWindow

Data structures

wxDynamicSashWindow::wxDynamicSashWindow

wxDynamicSashWindow(wxWindow* parent, wxWindowID id, const wxPoint& pos = wxDefaultPosition, const wxSize& size = wxDefaultSize, long style = wxCLIP_CHILDREN | wxDS_MANAGE_SCROLLBARS | wxDS_DRAG_CORNER, const wxString& name = "dynamicSashWindow")

wxDynamicSashWindow()

wxDynamicSashWindow::~~wxDynamicSashWindow

~wxDynamicSashWindow()

wxDynamicSashWindow::AddChild

void AddChild(wxWindowBase* child)

This is overloaded from wxWindowBase. It's not here for you to call directly.

wxDynamicSashWindow::Create

bool Create(wxWindow* parent, wxWindowID id, const wxPoint& pos = wxDefaultPosition, const wxSize& size = wxDefaultSize, long style = wxCLIP_CHILDREN | wxDS_MANAGE_SCROLLBARS | wxDS_DRAG_CORNER, const wxString& name = "dynamicSashWindow")

wxDynamicSashWindow::GetHScrollBar

wxScrollBar* GetHScrollBar(const wxWindow* *child*) const

wxDynamicSashWindow::GetVScrollBar

wxScrollBar* GetVScrollBar(const wxWindow* *child*) const

wxEditableListBox

This class provides a composite control that lets the user easily enter list of strings

Derived from

wxPanel

Data structures

wxEditableListBox::wxEditableListBox

**wxEditableListBox(wxWindow* *parent*, wxWindowID *id*, const wxString& *label*,
const wxPoint& *pos* = wxDefaultPosition, const wxSize& *size* = wxDefaultSize, const
wxString& *name* = wxT("editableListBox"))**

wxEditableListBox::GetStrings

void GetStrings(wxArrayString& *strings*)

wxEditableListBox::OnDelItem

void OnDelItem(wxCommandEvent& *event*)

wxEditableListBox::OnDownItem

void OnDownItem(wxCommandEvent& *event*)

wxEditableListBox::OnEditItem

void OnEditItem(wxCommandEvent& *event*)

wxEditableListBox::OnEndLabelEdit

void OnEndLabelEdit(wxListEvent& *event*)

wxEditableListBox::OnItemSelected

void OnItemSelected(wxListEvent& *event*)

wxEitableListBox::OnNewItem**void OnNewItem(wxCommandEvent& event)****wxEitableListBox::OnUpItem****void OnUpItem(wxCommandEvent& event)****wxEitableListBox::SetStrings****void SetStrings(const wxArrayString& strings)****wxLEDNumberCtrl**

wxLEDNumberCtrl

Derived from*wxControl***Data structures****wxLEDNumberCtrl::wxLEDNumberCtrl****wxLEDNumberCtrl(wxWindow* parent, wxWindowID id = -1, const wxPoint& pos = wxDefaultPosition, const wxSize& size = wxDefaultSize, long style = wxLED_ALIGN_LEFT | wxLED_DRAW_FADED)****wxLEDNumberCtrl()**

Constructors.

wxLEDNumberCtrl::Create**bool Create(wxWindow* parent, wxWindowID id = -1, const wxPoint& pos = wxDefaultPosition, const wxSize& size = wxDefaultSize, long style = 0)**

Create functions.

wxLEDNumberCtrl::GetAlignment**wxLEDValueAlign GetAlignment() const****wxLEDNumberCtrl::GetDrawFaded****bool GetDrawFaded() const**

wxLEDNumberCtrl::GetValue

const wxString& GetValue() *const*

wxLEDNumberCtrl::SetAlignment

void SetAlignment(wxLEDValueAlign Alignment, bool Redraw = TRUE)

wxLEDNumberCtrl::SetDrawFaded

void SetDrawFaded(bool DrawFaded, bool Redraw = TRUE)

wxLEDNumberCtrl::SetValue

void SetValue(const wxString& Value, bool Redraw = TRUE)

wxMultiCellCanvas

wxCell is used internally, so we don't need to declare it here

wxMultiCellCanvas

Derived from

wxFlexGridSizer

Data structures**wxMultiCellCanvas::wxMultiCellCanvas**

wxMultiCellCanvas(wxWindow* parent, int numRows = 2, int numCols = 2)

wxMultiCellCanvas::Add

void Add(wxWindow* win, unsigned int row, unsigned int col)

wxMultiCellCanvas::CalculateConstraints

void CalculateConstraints()

wxMultiCellCanvas::MaxCols

int MaxCols()

wxMultiCellCanvas::MaxRows

int MaxRows()

wxMultiCellCanvas::Resize

void **Resize**(*int numRows, int numCols*)

wxMultiCellCanvas::SetMinCellSize

void **SetMinCellSize**(*const wxSize size*)

wxMultiCellItemHandle

classes

wxMultiCellItemHandle

Derived from

wxObject

Data structures

wxMultiCellItemHandle::wxMultiCellItemHandle

wxMultiCellItemHandle(*int row, int column, wxSize size, wxResizable style = wxNOT_RESIZABLE, wxSize weight = wxSize(1,1), int align = wxALIGN_NOT*)

wxMultiCellItemHandle(*int row, int column, wxResizable style, wxSize weight = wxSize(1,1), int align = wxALIGN_NOT*)

wxMultiCellItemHandle(*int row, int column, int align*)

wxMultiCellItemHandle(*int row, int column, int height = 1, int width = 1, wxSize size = wxDefaultSize, wxResizable style = wxNOT_RESIZABLE, wxSize weight = wxSize(1,1), int align = wxALIGN_NOT*)

wxMultiCellItemHandle::GetAlignment

int **GetAlignment**()

wxMultiCellItemHandle::GetColumn

int **GetColumn**()

wxMultiCellItemHandle::GetHeight

int **GetHeight**()

wxMultiCellItemHandle::GetLocalSize

wxSize GetLocalSize()

wxMultiCellItemHandle::GetRow

int GetRow()

wxMultiCellItemHandle::GetStyle

wxResizable GetStyle()

wxMultiCellItemHandle::GetWeight

wxSize GetWeight()

wxMultiCellItemHandle::GetWidth

int GetWidth()

wxMultiCellSizer

wxMultiCellSizer

Derived from

wxSizer

Data structures

wxMultiCellSizer::wxMultiCellSizer

wxMultiCellSizer(int rows, int cols)

wxMultiCellSizer(wxSize & size)

wxMultiCellSizer::~~wxMultiCellSizer

~wxMultiCellSizer()

wxMultiCellSizer::CalcMin

wxSize CalcMin()

wxMultiCellSizer::EnableGridLines

bool EnableGridLines(wxWindow* win)

wxMultiCellSizer::OnPaint

void OnPaint(**wxDC&** *dc*)

wxMultiCellSizer::RecalcSizes

void RecalcSizes()

wxMultiCellSizer::SetColumnWidth

bool SetColumnWidth(**int** *column*, **int** *colSize* = 5, **bool** *expandable* = FALSE)

wxMultiCellSizer::SetDefaultCellSize

bool SetDefaultCellSize(**wxSize** *size*)

wxMultiCellSizer::SetGridPen

bool SetGridPen(**wxPen*** *pen*)

wxMultiCellSizer::SetRowHeight

bool SetRowHeight(**int** *row*, **int** *rowSize* = 5, **bool** *expandable* = FALSE)

wxRemotelyScrolledTreeCtrl

wxRemotelyScrolledTreeCtrl This tree control disables its vertical scrollbar and catches scroll events passed by a scrolled window higher in the hierarchy. It also updates the scrolled window vertical scrollbar as appropriate. **Derived from**

wxTreeCtrl

Data structures**wxRemotelyScrolledTreeCtrl::wxRemotelyScrolledTreeCtrl**

wxRemotelyScrolledTreeCtrl(**wxWindow*** *parent*, **wxWindowID** *id*, **const wxPoint&** *pt* = *wxDefaultPosition*, **const wxSize&** *sz* = *wxDefaultSize*, **long** *style* = *wxTR_HAS_BUTTONS*)

wxRemotelyScrolledTreeCtrl::~~wxRemotelyScrolledTreeCtrl

~wxRemotelyScrolledTreeCtrl()

wxRemotelyScrolledTreeCtrl::AdjustRemoteScrollbars

void AdjustRemoteScrollbars()

Adjust the containing wxScrolledWindow's scrollbars appropriately

wxRemotelyScrolledTreeCtrl::CalcTreeSize

void CalcTreeSize(const wxTreeItemId& id, wxRect& rect)

void CalcTreeSize(wxRect& rect)

Calculate the tree overall size so we can set the scrollbar correctly

wxRemotelyScrolledTreeCtrl::GetCompanionWindow

wxWindow* GetCompanionWindow() const

wxRemotelyScrolledTreeCtrl::GetScrollPos

int GetScrollPos(int orient) const

In case we're using the generic tree control.

wxRemotelyScrolledTreeCtrl::GetScrolledWindow

wxScrolledWindow* GetScrolledWindow() const

Find the scrolled window that contains this control

wxRemotelyScrolledTreeCtrl::GetViewStart

void GetViewStart(int* x, int* y) const

In case we're using the generic tree control. Get the view start

wxRemotelyScrolledTreeCtrl::HideVScrollbar

void HideVScrollbar()

Helpers

wxRemotelyScrolledTreeCtrl::OnExpand

void OnExpand(wxTreeEvent& event)

wxRemotelyScrolledTreeCtrl::OnPaint

void OnPaint(wxPaintEvent& event)

wxRemotelyScrolledTreeCtrl::OnScroll

void OnScroll(wxScrollWinEvent& event)

wxRemotelyScrolledTreeCtrl::OnSize

void OnSize(wxSizeEvent& event)

Events

wxRemotelyScrolledTreeCtrl::PrepareDC

void PrepareDC(wxDC& dc)

In case we're using the generic tree control.

wxRemotelyScrolledTreeCtrl::ScrollToLine

void ScrollToLine(int posHoriz, int posVert)

Scroll to the given line (in scroll units where each unit is the height of an item)

wxRemotelyScrolledTreeCtrl::SetCompanionWindow

void SetCompanionWindow(wxWindow* companion)

Accessors The companion window is one which will get notified when certain events happen such as node expansion

wxRemotelyScrolledTreeCtrl::SetScrollbars

void SetScrollbars(int pixelsPerUnitX, int pixelsPerUnitY, int noUnitsX, int noUnitsY, int xPos = 0, int yPos = 0, bool noRefresh = FALSE)

Overrides Override this in case we're using the generic tree control. Calls to this should disable the vertical scrollbar. Number of pixels per user unit (0 or -1 for no scrollbar)
Length of virtual canvas in user units Length of page in user units

wxSplitterScrolledWindow

wxSplitterScrolledWindow This scrolled window is aware of the fact that one of its children is a splitter window. It passes on its scroll events (after some processing) to both splitter children for them to scroll appropriately. **Derived from**

wxScrolledWindow

Data structures

wxSplitterScrolledWindow::wxSplitterScrolledWindow

wxSplitterScrolledWindow(wxWindow* *parent*, wxWindowID *id* = -1, const wxPoint& *pos* = wxDefaultPosition, const wxSize& *sz* = wxDefaultSize, long *style* = 0)

wxSplitterScrolledWindow::OnScroll

void OnScroll(wxScrollWinEvent& *event*)

Overrides Events

wxSplitterScrolledWindow::OnSize

void OnSize(wxSizeEvent& *event*)

wxThinSplitterWindow

wxThinSplitterWindow Implements a splitter with a less obvious sash than the usual one. **Derived from**

wxSplitterWindow

Data structures

wxThinSplitterWindow::wxThinSplitterWindow

wxThinSplitterWindow(wxWindow* *parent*, wxWindowID *id* = -1, const wxPoint& *pos* = wxDefaultPosition, const wxSize& *sz* = wxDefaultSize, long *style* = wxSP_3D | wxCLIP_CHILDREN)

wxThinSplitterWindow::DrawSash

void DrawSash(wxDC& *dc*)

wxThinSplitterWindow::OnSize

void OnSize(wxSizeEvent& *event*)

Events

wxThinSplitterWindow::SashHitTest

bool SashHitTest(int *x*, int *y*, int *tolerance* = 2)

Tests for *x*, *y* over sash. Overriding this allows us to increase the tolerance.

wxThinSplitterWindow::SizeWindows

void SizeWindows()

Overrides

wxTreeCompanionWindow

wxTreeCompanionWindow A window displaying values associated with tree control items. **Derived from**

wxWindow

Data structures

wxTreeCompanionWindow::wxTreeCompanionWindow

wxTreeCompanionWindow(*wxWindow** parent, *wxWindowID* id = -1, **const wxPoint&** pos = *wxDefaultPosition*, **const wxSize&** sz = *wxDefaultSize*, **long** style = 0)

wxTreeCompanionWindow::DrawItem

void DrawItem(*wxDC&* dc, *wxTreeItemId* id, **const wxRect&** rect)

Overrides

wxTreeCompanionWindow::GetTreeCtrl

wxRemotelyScrolledTreeCtrl* GetTreeCtrl() **const**

Operations Accessors

wxTreeCompanionWindow::OnExpand

void OnExpand(*wxTreeEvent&* event)

wxTreeCompanionWindow::OnPaint

void OnPaint(*wxPaintEvent&* event)

Events

wxTreeCompanionWindow::OnScroll

void OnScroll(*wxScrollWinEvent&* event)

wxTreeCompanionWindow::SetTreeCtrl

void SetTreeCtrl(*wxRemotelyScrolledTreeCtrl** treeCtrl)

Classes by category

A classification of Gizmos classes by category.

Topic overviews

This chapter contains a selection of topic overviews, first things first:

Notes on using the reference

In the descriptions of the wxWindows classes and their member functions, note that descriptions of inherited member functions are not duplicated in derived classes unless their behaviour is different. So in using a class such as wxScrolledWindow, be aware that wxWindow functions may be relevant.

Note also that arguments with default values may be omitted from a function call, for brevity. Size and position arguments may usually be given a value of -1 (the default), in which case wxWindows will choose a suitable value.

Most strings are returned as wxString objects. However, for remaining char * return values, the strings are allocated and deallocated by wxWindows. Therefore, return values should always be copied for long-term use, especially since the same buffer is often used by wxWindows.

The member functions are given in alphabetical order except for constructors and destructors which appear first.

References

Index

—~—
~wxDynamicSashWindow, 3
~wxMultiCellSizer, 8
~wxRemotelyScrolledTreeCtrl, 9

—A—
Add, 6
AddChild, 3
AdjustRemoteScrollbars, 10

—C—
CalcMin, 8
CalcTreeSize, 10
CalculateConstraints, 6
Clone, 2, 3
Create, 3, 5

—D—
DrawItem, 13
DrawSash, 12

—E—
EnableGridLines, 8

—G—
GetAlignment, 5, 7
GetColumn, 7
GetCompanionWindow, 10
GetDrawFaded, 5
GetHeight, 7
GetHScrollBar, 4
GetLocalSize, 8
GetRow, 8
GetScrolledWindow, 10
GetScrollPos, 10
GetStrings, 4
GetStyle, 8
GetTreeCtrl, 13
GetValue, 6
GetViewStart, 10
GetVScrollBar, 4
GetWeight, 8
GetWidth, 8

—H—
HideVScrollbar, 10

—M—
MaxCols, 6
MaxRows, 6

—O—
OnDeleteItem, 4
OnDownItem, 4
OnEditItem, 4
OnEndLabelEdit, 4
OnExpand, 10, 13
OnItemSelected, 4
OnNewItem, 5
OnPaint, 9, 10, 13
OnScroll, 11, 12, 13
OnSize, 11, 12
OnUpItem, 5

—P—
PrepareDC, 11

—R—
RecalcSizes, 9
Resize, 7

—S—
SashHitTest, 12
ScrollToLine, 11
SetAlignment, 6
SetColumnWidth, 9
SetCompanionWindow, 11
SetDefaultCellSize, 9
SetDrawFaded, 6
SetGridPen, 9
SetMinCellSize, 7
SetRowHeight, 9
SetScrollbars, 11
SetStrings, 5
SetTreeCtrl, 13
SetValue, 6
SizeWindows, 12

—W—
wxDynamicSashSplitEvent, 2
wxDynamicSashSplitEvent::Clone, 2
wxDynamicSashSplitEvent::wxDynamicSashSplitEvent, 2
wxDynamicSashUnifyEvent, 3
wxDynamicSashUnifyEvent::Clone, 3

wxDynamicSashUnifyEvent::wxDynamicSashUnifyEvent, 2
wxDynamicSashWindow, 3
wxDynamicSashWindow::~wxDynamicSashWindow, 3
wxDynamicSashWindow::AddChild, 3
wxDynamicSashWindow::Create, 3
wxDynamicSashWindow::GetHScrollbar, 3
wxDynamicSashWindow::GetVScrollbar, 4
wxDynamicSashWindow::wxDynamicSashWindow, 3
wxEditableListBox, 4
wxEditableListBox::GetStrings, 4
wxEditableListBox::OnDeleteItem, 4
wxEditableListBox::OnDownItem, 4
wxEditableListBox::OnEditItem, 4
wxEditableListBox::OnEndLabelEdit, 4
wxEditableListBox::OnItemSelected, 4
wxEditableListBox::OnNewItem, 5
wxEditableListBox::OnUpItem, 5
wxEditableListBox::SetStrings, 5
wxEditableListBox::SetDrawFaded, 4
wxLEDNumberCtrl, 5
wxLEDNumberCtrl::Create, 5
wxLEDNumberCtrl::GetAlignment, 5
wxLEDNumberCtrl::GetDrawFaded, 5
wxLEDNumberCtrl::GetValue, 6
wxLEDNumberCtrl::SetAlignment, 6
wxLEDNumberCtrl::SetDrawFaded, 6
wxLEDNumberCtrl::SetValue, 6
wxLEDNumberCtrl::wxLEDNumberCtrl, 5
wxMultiCellCanvas, 6
wxMultiCellCanvas::Add, 6
wxMultiCellCanvas::CalculateConstraints, 6
wxMultiCellCanvas::MaxCols, 6
wxMultiCellCanvas::MaxRows, 6
wxMultiCellCanvas::Resize, 7
wxMultiCellCanvas::SetMinCellSize, 7
wxMultiCellCanvas::wxMultiCellCanvas, 6
wxMultiCellItemHandle, 7
wxMultiCellItemHandle::GetAlignment, 7
wxMultiCellItemHandle::GetColumn, 7
wxMultiCellItemHandle::GetHeight, 7
wxMultiCellItemHandle::GetLocalSize, 7
wxMultiCellItemHandle::GetRow, 8
wxMultiCellItemHandle::GetStyle, 8
wxMultiCellItemHandle::GetWeight, 8
wxMultiCellItemHandle::GetWidth, 8
wxMultiCellItemHandle::wxMultiCellItemHandle, 7
wxMultiCellSizer, 8
wxMultiCellSizer::~wxMultiCellSizer, 8
wxMultiCellSizer::CalcMin, 8
wxMultiCellSizer::EnableGridLines, 8
wxMultiCellSizer::OnPaint, 9
wxMultiCellSizer::RecalcSizes, 9
wxMultiCellSizer::SetColumnWidth, 9
wxMultiCellSizer::SetDefaultCellSize, 9
wxMultiCellSizer::SetGridPen, 9
wxMultiCellSizer::SetRowHeight, 9
wxMultiCellSizer::wxMultiCellSizer, 8
wxRemotelyScrolledTreeCtrl, 9
wxRemotelyScrolledTreeCtrl::~wxRemotelyScrolledTreeCtrl, 9
wxRemotelyScrolledTreeCtrl::AdjustRemoteScrollbars, 9
wxRemotelyScrolledTreeCtrl::CalcTreeSize, 10
wxRemotelyScrolledTreeCtrl::GetCompanionWindow, 10
wxRemotelyScrolledTreeCtrl::GetScrolledWindow, 10
wxRemotelyScrolledTreeCtrl::GetScrollPos, 10
wxRemotelyScrolledTreeCtrl::GetViewStart, 10
wxRemotelyScrolledTreeCtrl::HideVScrollbar, 10
wxRemotelyScrolledTreeCtrl::OnExpand, 10
wxRemotelyScrolledTreeCtrl::OnPaint, 10
wxRemotelyScrolledTreeCtrl::OnScroll, 10
wxRemotelyScrolledTreeCtrl::OnSize, 11
wxRemotelyScrolledTreeCtrl::PrepareDC, 11
wxRemotelyScrolledTreeCtrl::ScrollToLine, 11
wxRemotelyScrolledTreeCtrl::SetCompanionWindow, 11
wxRemotelyScrolledTreeCtrl::SetScrollbars, 11
wxRemotelyScrolledTreeCtrl::wxRemotelyScrolledTreeCtrl, 9
wxSplitterScrolledWindow, 12
wxSplitterScrolledWindow::OnScroll, 12
wxSplitterScrolledWindow::OnSize, 12
wxSplitterScrolledWindow::wxSplitterScrolledWindow, 11
wxThinSplitterWindow, 12
wxThinSplitterWindow::DrawSash, 12
wxThinSplitterWindow::OnSize, 12
wxThinSplitterWindow::SashHitTest, 12
wxThinSplitterWindow::SizeWindows, 12
wxThinSplitterWindow::wxThinSplitterWindow, 12
wxTreeCompanionWindow, 13
wxTreeCompanionWindow::DrawItem, 13
wxTreeCompanionWindow::GetTreeCtrl, 13
wxTreeCompanionWindow::OnExpand, 13
wxTreeCompanionWindow::OnPaint, 13
wxTreeCompanionWindow::OnScroll, 13
wxTreeCompanionWindow::SetTreeCtrl, 13
wxTreeCompanionWindow::wxTreeCompanionWindow, 13