

The Package Multis

Installation

TMultiPanel

TMultiButton

TMultiButtonStyleManager

TMultiplexSlider

TMultiSeperator

TMultiLayer

TMultiRadioGroup

TMultiCheckGroup

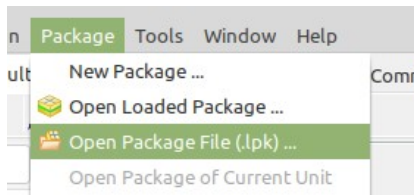
TMultiSwitch

This text was translated with DeepL and my poor school English.

See also <https://www.lazarusforum.de/viewtopic.php?f=29&t=14033>
or the demo projects in the package.

Installation

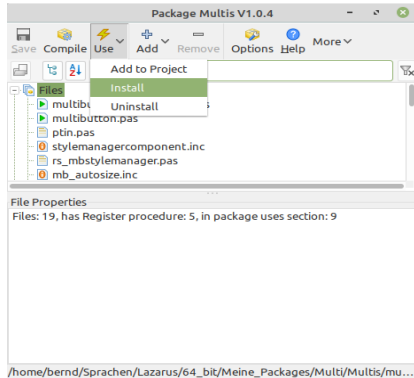
The package is located in the following Github account: <https://github.com/wennerer/Multis>
After the package has been cloned or downloaded, it can be installed in Lazarus. To do this, open Lazarus and click on Open Package File (.lpk)... under Package.



Now navigate to the Multis folder and select the file multis.lpk.

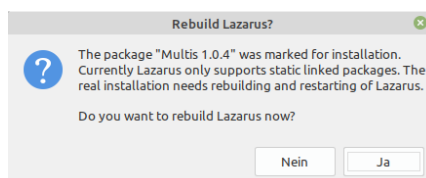


The following window will open:

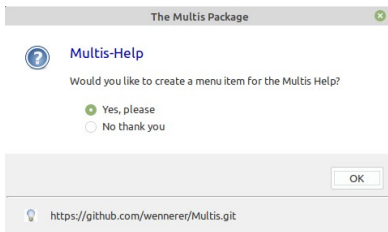


Click on Use and then Install.

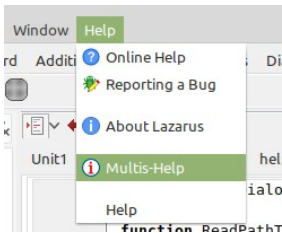
Confirm this dialogue with Yes:



In between, you will be asked if you want to create a menu entry for this help. If you want to do this, you should simply confirm with Yes.



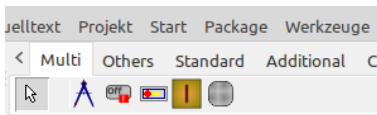
In the Help menu there is now a new entry Multis-Help.



If you want to call up the installation dialogue again for whatever reason, delete the file multis.xml in the Lazarus Config folder. The dialogue will then be displayed again.

Tip: If the help menu entry is available but the help file is not found, this may be because invalid package links have been entered. Please search for multis under Package, Package-Links and remove all invalid links.

After successful installation, there is a new Multi tab in the palette selection.



TMultiPanel

Properties

Action	: The Action associated with the control
Align	: Specifies the placement of the control inside its Parent.
Anchors	: The set of anchor definitions for this control
AnimationSpeed	: Speed for Appear bzw. Disappear (default 0,05) (only at runtime!)
Autosize	: Allows automatic adjustment of the size for the control, according to its content
Appear	: makes the panel appear (only at runtime!)
BidiMode	: Customization (of text controls) in bidirectional reading environments
BorderSettings	: The properties of the border
BorderSettings.Between	: The space between inner- and outerborder
BorderSettings.InnerColor	: The color of the innerborder
BorderSettings.InnerWidth	: The width of the innerborder
BorderSettings.OuterColor	: The color of the outerborder
BorderSettings.OuterWidth	: The width of the outerborder
BorderSpacing	: Determines the inner and outer border spacing for this control
Caption	: The text that the user writes in the panel
CaptionAlignment	: Alignment of the text in the caption (left, center, right)
CaptionHorMargin	: The horizontal distance of the text in the text rectangle (only effective with taLeftJustify)
CaptionLayout	: Alignment of the text in the caption (top, center, bottom)
CaptionVerMargin	: The vertical distance of the text in the text rectangle (only effective with tlTop)
CaptionWordbreak	: Allows a line break in the caption
ColorEnd	: The end color of the panel (for color gradient)
ColorGradient	: The direction of the gradient
ColorStart	: The start color of the panel (for color gradient)
Constraints	: The minimum and maximum Width and Height for the control
Cursor	: The shape of the mouse pointer, when the mouse is over this control
Disappear	: makes the panel disappear (only at runtime!)
DoubleBuffered	: Allows to reduce flicker in the painting of the control
DragCursor	: The cursor shape shown while the control is dragged
DragKind	: The operation when the control is dragged - Drag or Dock
DragMode	: Allows the user to drag the control
DrawACustomPanel	: Opens an editor where you can draw a panel
DropDownMenu	: The properties of the dropdownmenu
DropDownMenu.Active	: Activates the dropdown function
DropDownMenu.Compressed	: Properties of the compressed panel
DropDownMenu.Compressed.Active	: Makes the selection the starting value
DropDownMenu.Compressed.Height	: The height of the compressed panel
DropDownMenu.Compressed.Width	: The width of the compressed panel
DropDownMenu.Direction	: The fold-out direction
DropDownMenu.Hotspot	: Defines the area in which a click is effective, only active with DropDownMenu.Active and trPinned (only at runtime!)

DropDownMenu.HotspotCursor	: The shape of the mouse pointer, when the mouse is over the hotspot (only at runtime!)
DropDownMenu.Speed	: The drawing speed (timer intervall)
DropDownMenu.Step	: The drawing steps (pixels)
DropDownMenu.Stretched	: Properties of the stretched Panel
DropDownMenu.Stretched.Active	: Makes the selection the starting value
DropDownMenu.Stretched.Height	: The height of the stretched panel
DropDownMenu.Stretched.Width	: The width of the stretched panel
DropDownMenu.Trigger	: Trigger
Font	: The font to be used for text display in this panel
Height	: The vertical size of the control
HelpContext	: The ID for context-sensitive Help on this control
HelpKeyword	: The keyword for context-sensitive Help on this control
HelpType	: Determines whether context-sensitive Help is selected by numeric ID or keyword
Hint	: The text to show in the Hint window for the control
ImageIndex	: The Index of a Image in a ImageList
ImageLeft	: The coordinate of the left edge of a Image
Images	: A list for including images
ImageTop	: The coordinate of the top edge of a Image
ImageWidth	: The unique width of all images in the list
Left	: The client coordinate of the left edge of the control
ParentAsBkgrd	: Background of the panel takes on the colour of the parent (only at runtime!)
RndRctRadius	: Corner diameter if the geometric shape is RoundRect
Style	: The geometric shape of the panel
Top	: The client coordinate of the top edge of the control
Visible	: Allows to show or hide the control, and all of its children
Width	: The horizontal extent of the control

Events

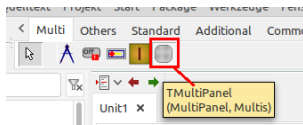
OnChangeBounds	: Event handler for a change of the Bounds of the control
OnClick	: Notification handler for mouse clicks
OnCompressed	: Handler when the panel is compressed, only active when DropDownMenu.Active
OnDragDrop	: This handler determines the action on an drop onto this control, in a drag-drop operation
OnDragOver	: Event handler for a control being dragged over this control
OnEndDrag	: Notification handler for the end of a dragging operation
OnEnter	: Handler for control receiving the focus
OnExit	: Handler for control loosing the focus; This is a good place for checking the finished user input
OnKeyDown	: Handler for keyboard key pressed
OnKeyPress	: Handler for a character entered by the user
OnKeyUp	: Handler for keyboard key released
OnMouseDown	: Event handler for mouse button going down
OnMouseEnter	: Event handler for mouse entering the area of the control
OnMouseLeave	: Event handler for mouse leaving the area of the control
OnMouseMove	: Event handler for mouse movement within the control
OnMouseUp	: Event handler for mouse button going up
OnStartDrag	: Event handler for the start of a dragging operation
OnStreched	: Handler wenn das Panel ausgeklappt ist, nur aktive wenn DropDownMenu.Active
OnVisible	: Is triggered when the panel becomes visible for the first time

Public procedures

```
procedure MouseMove({%H-}Shift: TShiftState; X, Y: Integer);override;
procedure MouseDown({%H-}Button: TMouseButton;{%H-}Shift: TShiftState; X, Y: Integer);override;
procedure MouseUp({%H-}Button: TMouseButton; {%H-}Shift: TShiftState; {%H-}X, {%H-}Y: Integer);override;
procedure LoadFromFile(aFileName: string);
procedure InvalidateBackground;
procedure ParentInputHandler({%H-}Sender: TObject; Msg: Cardinal);
procedure Notification(AComponent: TComponent;Operation: TOperation); override;
constructor Create(AOwner: TComponent); override;
destructor Destroy; override;
procedure MouseEnter; override;
procedure MouseLeave; override;
procedure Paint; override;
```

Description

You will find the MultiPanel in the Multis tab.

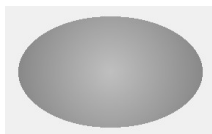


The shape of the MultiPanel can be influenced with the [Style](#) property.

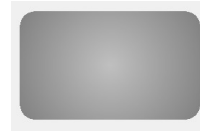
mpsRect:



mpsEllipse:



mpsRoundRect:

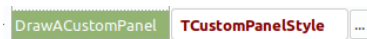


The corner radius can be set with [RndRctRadius](#). Default setting is 40

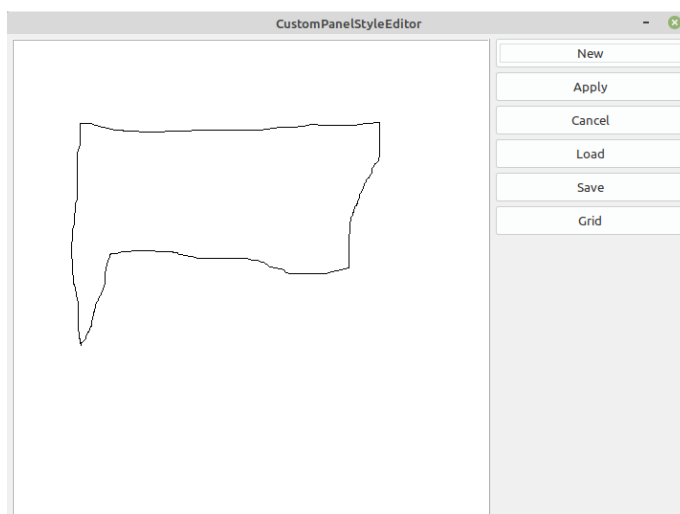
mpsCustom:



By default, mpsCustom has a triangle behind it. To draw a custom panel, click on the 3 dots behind [DrawACustomPanel](#).



A property editor will open:



If you now click on New, you can simply draw the shape of the MultiPanel with the mouse. If you click on Use, the MultiPanel shape is adopted. With Discard the MultiPanel shape is not accepted and the editor is closed. With Save you can save a drawn shape and with Load you can get it again. Grid displays an auxiliary grid that may help you when drawing.

Remember it must be set to mpsCustom!

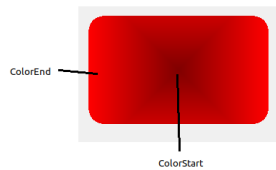
At runtime, MultiPanels saved in advance can also be loaded with [LoadFromFile](#).

To change the colour of the MultiPanel you need the properties [ColorStart](#), [ColorEnd](#) and [ColorGradient](#). To get a single-coloured MultiPanel, ColorStart and ColorEnd must be the same. Otherwise, the composition of the three properties determines the appearance.

gcSpread:



gcRadiant:

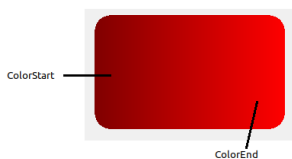


gcAlternate:



Alternately sets one pixel each to Start- and End-Color.

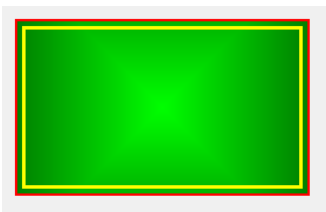
gcHorizontal:



gcVertical:



If you want to highlight the border, you can use the properties of the [BorderSettings](#).



BorderSettings		(TBorder)
Between	7	
InnerColor	Yellow	clYellow
InnerWidth	3	
OuterColor	Red	clRed
OuterWidth	3	

To create a border, simply select a colour. If you do not want a border, use clNone.

[BorderSettings.Between](#)

: The space between inner- and outerborder

[BorderSettings.InnerColor](#)

: The color of the innerborder

[BorderSettings.InnerWidth](#)

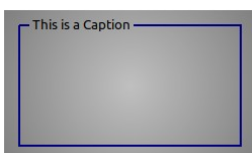
: The width of the innerborder

[BorderSettings.OuterColor](#)

: The color of the outerborder

[BorderSettings.OuterWidth](#)

: The width of the outerborder



If you want to imitate a GroupBox, make the inner border visible and move it to the desired position with [BorderSettings.Between](#). In order for the border to be interrupted in the caption area, [taLeftJustify](#) and [tlTop](#) must be selected for the caption!

The [Appear](#), [Disappear](#) and [AnimationSpeed](#) properties can only be set at runtime!

To make an invisible MultiPanel appear, use the property [Appear](#).

Example code:

```
procedure TForm1.MultiButton1Click(Sender: TObject);
begin
    MultiPanel1.Appear:= true;
end;
```

To make a visible MultiPanel disappear, use the property [Disappear](#).

Example code:

```
procedure TForm1.MultiButton2Click(Sender: TObject);
begin
    MultiPanel1.Disappear:= true;
end;
```

With the property [AnimationSpeed](#) the speed of appearance or disappearance can be influenced.

The default value is 0.05. The smaller the value, the slower the animation. With a value of 0.001 it is already very slow.

Example code:

```
MultiPanel1.AnimationSpeed:= 0.001;
```

If you select something other than `mpsRect` as the geometric shape (Style property), a part of the background of the `MultiPanel` becomes visible. These visible corners take on the colour set in the parent. If there are self-drawn lines in the parent, for example, these are also shown. This happens because the property `ParentAsBkgrd` is set to true by default.

`ParentAsBkgrd := true`



`ParentAsBkgrd := false`

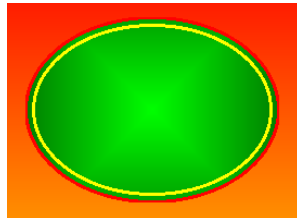


This setting makes sense especially when the parent changes its size. Because then, for example, the drawn line is not scaled correctly here.

If the parent has a colour gradient, it is possible to compensate for the scaling problem by calling the procedure `InvalidateBackground`.

Example code:

```
procedure TForm1.FormChangeBounds(Sender: TObject);
begin
    MultiPanel1.InvalidateBackground;
end;
```



To create a [DropDown](#) menu (hamburger menu), first set the property [DropDownMenu.Active](#) to true.

DrawACustomPanel	TCustomPanelStyle
DropDownMenu	(TDDMenu)
Active	<input checked="" type="checkbox"/> (True)
Compressed	(TComp)
Direction	LeftTop_RightBottom
Speed	3
Step	2
Stretched	(TStre)
Trigger	trHover

The MultiPanel now shows the compressed state. The positioning can be done with the mouse or the properties [Left](#) or [Top](#) (of course anchors can also be set). The size can simply be dragged with the mouse or assigned with the properties [DropDownMenu.Compressed.Height](#) or [DropDownMenu.Compressed.Width](#). If the size fits, switch to the expanded state with the properties [DropDownMenu.Stretched.Active](#) or [DropDownMenu.Compressed.Active](#). Now the desired size can also be set by dragging with the mouse or with the properties [DropDownMenu.Stretched.Height](#) or [DropDownMenu.Stretched.Width](#). It is recommended to place the desired child controls (buttons etc.) in this state. The direction in which the MultiPanel unfolds is determined by the property [DropDow-](#)

[nMenu.Direction](#). The following options are available:

[TDirection = \(LeftTop_RightBottom, RightTop_LeftBottom, LeftBottom_RightTop, RightBottom_LeftTop\)](#)

The speed of the unfolding can be influenced by the properties [DropDownMenu.Speed](#) and [DropDownMenu.Step](#). The timer interval with which the unfolding is called is hidden behind Speed. To slow down, increase this value to the desired speed. With Step you can set the number of additional pixels that are drawn per interval. If you want to unfold faster, increase the value for Step.

With [DropDownMenu.Trigger](#) you determine the trigger for the unfolding.

The following possibilities are available:

[TTrigger = \(trClick, trHover, trPinned\)](#)

With trClick you have to click in the panel, with trHover it is enough to move the mouse over it. With trPinned, the mouse must be clicked in a definable hotspot ([DropDownMenu.Hotspot](#)). If the mouse is over this hotspot, the mouse cursor can be adjusted with [DropDownMenu.HotspotCursor](#). By default, it changes to cr-HandPoint.

TMultiButton

Properties

Action	: The Action associated with the control
Align	: Specifies the placement of the control inside its Parent
AllowsUp	: Allows a pressed button to be set to not pressed
Anchors	: The set of anchor definitions for this control
AutoSize	: Allows automatic adjustment of the size for the control, according to its content
BidiMode	: Customization (of text controls) in bidirectional reading environments
BorderColor	: The color of the border
BorderSpacing	: Determines the inner and outer border spacing for this control
BorderWidth	: The whidth of the border
Caption	: The text that the user writes in the button
CaptionAlignment	: Alignment of the text in the caption (left, center, right)
CaptionHorMargin	: The horizontal distance of the text in the text rectangle (only effective with taLeftJustify)
CaptionLayout	: Alignment of the text in the caption (top, center, bottom)
CaptionVerMargin	: The vertical distance of the text in the text rectangle (only effective with tlTop)
CaptionWordbreak	: Allows a line break in the caption
ColorEnd	: The end color of the button (for color gradient)
ColorGradient	: The direction of the gradient
ColorStart	: The start color of the button (for color gradient)
Constraints	: The minimum and maximum Width and Height for the control
DisabledAlphaBValue	: The blendvalue at Enable:=false, only at runtime!
DisabledColor	: The colour at Enable:=false, only at runtime!
Down	: The Button has been set in the Down state
DragCursor	: The cursor shape shown while the control is dragged
DragKind	: The operation when the control is dragged - Drag or Dock
DragMode	: Allows the user to drag the control
Enable	: Determines whether the control reacts on mouse or keyboard input
FocusAlphaBValue	: How translucent the focusframe is (0=transparent, 255=opaque)
FocusColor	: The color of the Fokusframe/Foregroundfocus when the Control has the focus
FocusFrameOn	: Switches the focus frame on and off
FocusFrameWidth	: The whidth of the focus-frame
Font	: The font to be used for text display in this button
ForegroundFocusOn	: Indicates when the button has focus, switches on off
GroupIndex	: The Index within the group of MultiButtons
Height	: The vertical size of the control.The height of the MultiButton is minus HoverFrameWidth

HelpContext	: The ID for context-sensitive Help on this control
HelpKeyword	: The keyword for context-sensitive Help on this control
HelpType	: Determines whether context-sensitive Help is selected by numeric ID or keyword
Hint	: The text to show in the Hint window for the control
HoverEndColor	: The endcolor of a hoverevent
HoverFontColor	: The color of the Caption during one hoverevent
HoverImageIndex	: The Index of a Image in a ImageList when during one hoverevent
HoverOn	: Allows to show or hide a hoverevent
HoverStartColor	: The startcolor of a hoverevent
ImageIndex	: The Index of a Image in a ImageList
ImageLeft	: The coordinate of the left edge of a Image
Images	: A list for including images
ImageTop	: The coordinate of the top edge of a Image
ImageWidth	: The unique width of all images in the list
Left	: The client coordinate of the left edge of the control
MessageButton	: A message button to display information or to provide a second integrated button
MessageButton.Alignment	: The position of the messagebutton
MessageButton.BorderColor	: The color of the border
MessageButton.BorderWidth	: The whidth of the border
MessageButton.CalculateAlthoughInvisible	: Is required if the MessageButton is only visible at runtime
MessageButton.Caption	: The text that the user writes in the messagebutton
MessageButton.CaptionAlignment	: Alignment of the text in the caption (left, center, right)
MessageButton.CaptionHorMargin	: The horizontal distance of the text in the text rectangle (only effective with taLeftJustify)
MessageButton.CaptionLayout	: Alignment of the text in the caption (top, center, bottom)
MessageButton.CaptionVerMargin	: The vertical distance of the text in the text rectangle (only effective with tlTop)
MessageButton.ColorEnd	: The end color of the messagebutton (for color gradient)
MessageButton.ColorGradient	: The direction of the gradient
MessageButton.ColorStart	: The start color of the messagebutton (for color gradient)
MessageButton.Font	: The font to be used for text display in this button
MessageButton.Height	: The vertical size of the control
MessageButton.HoverOn	: The color of a hoverevent
MessageButton.HoverOn	: Allows to show or hide a hoverevent
MessageButton.ImageIndex	: The Index of a Image in a ImageList
MessageButton.ImageLeft	: The coordinate of the left edge of a Image
MessageButton.Images	: A list for including images
MessageButton.ImageTop	: The coordinate of the top edge of a Image

MessageButton.ImageWidth	: The unique width of all images in the list
MessageButton.PositionFactor	: Position factor, only active if alSE,alSW,alNW,alNE,alW,alE,alN,alS,alRightIn,alLeftIn,alTopIn,alBottomIn
MessageButton.PresdColBlendVal	: How translucent the pressedcolor is (0=transparent, 255=opaque)
MessageButton.Pres sedColor	: The color of the messagebutton when it is pressed
MessageButton.ShowBorder	: Allows to show or hide a border
MessageButton.ShowPressed	: Allows to show or hide the pressedoption
MessageButton.Style	: The geometric shape of the messagebutton
MessageButton.Visible	: Allows to show or hide the control, and all of its children
MessageButton.Width	: The horizontal extent of the control
MultiButton_StyleManager	: Simplifies the design of the MultiButton
PopupMenu	: A context-sensitive menu that pops up when the right mouse button is clicked over this control
PressedEndColor	: The end color of the button when it is pressed (for color gradient)
PressedFontColor	: The color of the text of the caption when the button is pressed
PressedImageIndex	: The Index of a Image in a ImageList when the Button is pressed
PressedStartColor	: The starting color of the button when it is pressed (for color gradient)
RndRctRadius	: Corner diameter if the geometric shape is RoundRect
ShowBorder	: Allows to show or hide a border
ShowHint	: When True, the Hint text is shown when the mouse hovers over the control
ShowMsgButtonInGroup	: Shows the message button on the MultiButton in a group
ShowTurnedOn	: Makes a visible MessageButton coloured when the button is Down
Style	: The geometric shape of the button
TabOrder	: Determines the sequence of control navigation when the user presses the Tab key
TabStop	: Allows the user to navigate to this control, by pressing the Tab key
Top	: The client coordinate of the top edge of the control
Visible	: Allows to show or hide the control, and all of its children
Width	: The horizontal size of the control.The width of the MultiButton is minus HoverFrameWidth

Public Procedures

procedure **SetStyleManager**(AValue: TmultiButtonStyleManager);

Public Variables

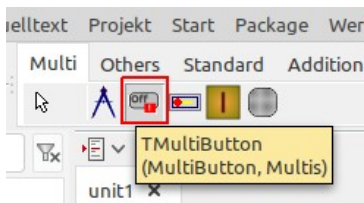
MouseButton : can be used in OnClick to determine with which mouse button the MultiButton was clicked.

Events

OnClick	: Notification handler for mouse clicks
OnDragDrop	: This handler determines the action on an drop onto this control, in a drag-drop operation
OnDragOver	: Event handler for a control being dragged over this control
OnEndDrag	: Notification handler for the end of a dragging operation
OnEnter	: Handler for control receiving the focus
OnExit	: Handler for control loosing the focus; This is a good place for checking the finished user input
OnKeyDown	: Handler for keyboard key pressed
OnKeyPress	: Handler for a character entered by the user
OnKeyUp	: Handler for keyboard key released
OnMouseDown	: Event handler for mouse button going down
OnMouseEnter	: Event handler for mouse entering the area of the control
OnMouseLeave	: Event handler for mouse leaving the area of the control
OnMouseMove	: Event handler for mouse movement within the control
OnMouseUp	: Event handler for mouse button going up
OnStartDrag	: Event handler for the start of a dragging operation
MessageButton.OnClick	: Notification handler for mouse clicks
MessageButton.OnMouseMove	: Event handler for mouse movement within the control

Description

You can find the MultiButton in the Multi tab:



It is important to know that the MultiButton is surrounded by a focus frame. As you can see here, the focused MultiButton has an olive green frame. This means that the actual button is smaller around the frame.



The [FocusColor](#) property can be used to set the colour of the focus frame. With [FocusAlphaBValue](#) the transparency of the focus frame can be controlled. The value 0 means transparent and 255 opaque. [FocusFrameWidth](#) determines the thickness of the frame.

Wert 50:



Wert 200:



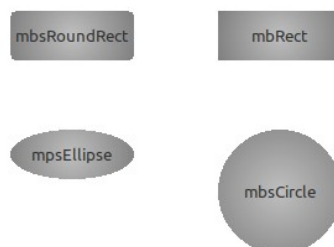
If [FocusFrameOn](#) is set to false, the border is retained but the focus is not shown in colour.

With [ForegroundFocusOn](#), the focused MultiButton has a dotted rectangle. The colour of the rectangle can be influenced with [FocusColor](#). It can be useful here to set [FocusFrameWidth](#) to 0 and [FocusFrameOn](#) to false so that the corners are not visible!



The [Style](#) property is used to set the desired geometric shape of the MultiButton.

If [mbsRoundRect](#) is set, the [RndRctRadius](#) property can be used to set the diameter of the corner rounding.



If you want to add a coloured border to the MultiButton, set [ShowBorder](#) to true. The colour of the border is set with [BorderColor](#) and the width with [BorderWidth](#).



To change the colour of the MultiButton you need the properties [ColorStart](#), [ColorEnd](#) and [ColorGradient](#). To get a single-coloured MultiButton, ColorStart and ColorEnd must be the same. Otherwise, the composition of the three properties determines the appearance.



Alternate alternately sets one pixel each to Start- and EndColor.



Here is StartColor clGreen and EndColor clYellow.

By default, [HoverOn](#) is set to true. This means that when a hover event occurs (the mouse moves over the MultiButton) the appearance can be changed as desired with [HoverStartColor](#), [HoverEndColor](#), [HoverFontColor](#) and [HoverImageIndex](#). If you do not want this, set HoverOn to false.

HoverEndColor	: The endcolor of a hoverevent
HoverFontColor	: The color of the Caption during one hoverevent
HoverImageIndex	: The Index of a Image in a ImageList when during one hoverevent
HoverOn	: Allows to show or hide a hoverevent
HoverStartColor	: The startcolor of a hoverevent

When the MultiButton is pressed, the properties [PressedStartColor](#), [PressedEndColor](#), [PressedFontColor](#) and [PressedImageIndex](#) influence the appearance. If you do not want any changes when the button is pressed, the only thing left to do is to set the same settings such as ColorStart etc..

PressedEndColor	: The end color of the button when it is pressed (for color gradient)
PressedFontColor	: The color of the text of the caption when the button is pressed
PressedImageIndex	: The Index of a Image in a ImageList when the Button is pressed
PressedStartColor	: The starting color of the button when it is pressed (for color gradient)

The [Enable](#) property determines whether the control reacts to mouse or keyboard input. The appearance when not enabled can be influenced at runtime with [DisabledAlphaBValue](#) and [DisabledColor](#).



If you do not want to see a frame, set [DisabledColor](#) to the same colour as the parent.

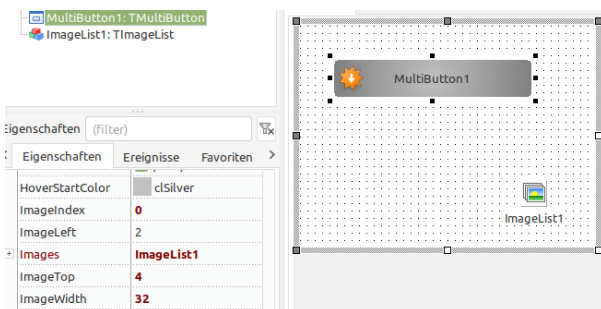


If necessary, use `MultiButton2.DisabledColor:=GetColorResolvingParent` instead of `clDefault`.

If you want to insert an image, you must first drag an ImageList onto the form. You then assign the desired images in the desired sizes to this ImageList. The operation of the ImageListEditor is described very well here:
<https://www.lazarusforum.de/viewtopic.php?f=18&t=13170>



With **Images** you enter the image list on the form. With **ImageIndex** you can select the desired image from the ImageList, where -1 means no image. With **ImageLeft** and **ImageTop** you determine the position of the image. With **ImageWidth** you can scale the size of the image. It is recommended to scale only smaller.



Tip:

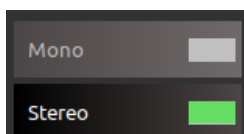
If you use HighDPI under Windows, the images and the MultiButton are scaled. In order for it to work at runtime, I had to select Vista-8:an,8.1+:pro Monitor(True/PM) in the project settings for DPI adjustment.

The **AllowsUp** property turns the button into a kind of switch. This means that when the button is pressed, it remains pressed until it is pressed again. If you want the MultiButton to appear pressed at the beginning of the programme, you do this with the property **Down**.

If the button is pressed, it is displayed with the properties Pressed....!

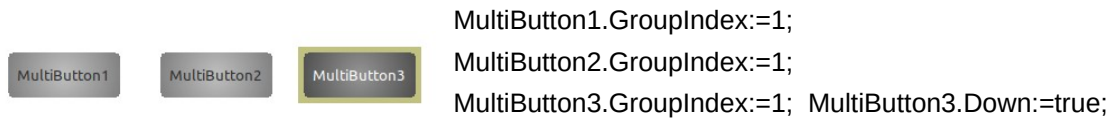


If you want to have an additional feature that the button is in the down position, you can make a MessageButton visible and set the property **ShowTurnedOn** to true. The MessageButton then gets the colour in down position that is defined under MessageButton.PressedColor.



See also **GroupIndex**

If needed, the MultiButton can belong to a group. This is achieved with the property [GroupIndex](#). If a MultiButton has a value other than 0, it belongs to the group with the same value. Only one MultiButton can be pressed in a group. If a button in the group should already be pressed when the programme is started, this can be achieved with the property [Down](#).



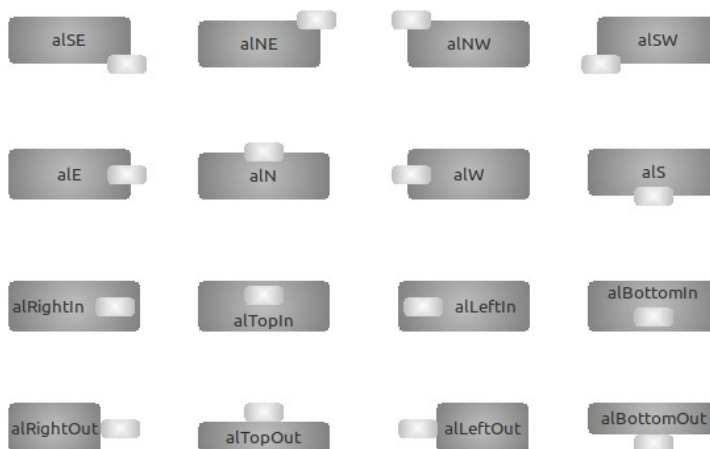
If you want to make the pressed button visually clearer, you can still use the property [ShowMsgButtonInGroup](#). The last pressed button gets a [MessageButton](#).



The [MessageButton](#)



To use the integrated [MessageButton](#), you must first set [MessageButton.Visible](#) to true. Then you can set the position of the [MessageButton](#) with [MessageButton.Alignment](#).



The [MessageButton.PositionFactor](#) property can be used to influence the position of the [MessageButton](#) somewhat. However, only with aSE, aSW, aNW, aNE, aW, aE, aN, aS, alRightIn, alLeftIn, alTopIn, alBottomIn



If you want to change the shape of the MessageButton, you can do this with the property [MessageButton.Style](#).



If you place several MultiButtons in a row and the MessageButton is not visible for all of them, the MultiButtons have different sizes. Here no. 2 appears larger:



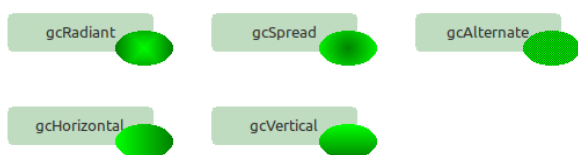
To get around this there is the property [MessageButton.CalculateAlthoughInvisible](#). If you set this property to true for No. 2, it looks like this:



If you want to provide the MessageButton with a coloured border, set [MessageButton.ShowBorder](#) to true. The colour of the border is set with [MessageButton.BorderColor](#) and the width with [MessageButton.BorderWidth](#).



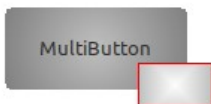
To change the colour of the MessageButton you need the properties [MessageButton.ColorStart](#), [MessageButton.ColorEnd](#) and [MessageButton.ColorGradient](#). To get a single-coloured MessageButton, ColorStart and ColorEnd must be the same. Otherwise, the composition of the three properties determines the appearance.



Alternate alternately sets one pixel each to Start- and EndColor.

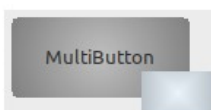
Here StartColor is clLime and EndColor is clGreen.

By default, [MessageButton.HoverOn](#) is set to true. This means that when a hover event occurs (the mouse moves over the MessageButton) a border is drawn around the MessageButton. The colour of the border can be set with [MessageButton.HoverColor](#). If you do not want this, set HoverOn to false.



[MessageButton.BorderWidth](#) influences the thickness of the hover border!

If the MessageButton is pressed and [MessageButton.ShowPressed](#) is true, then the colour set in [MessageButton.PressedColor](#) with the value stored in [MessageButton.PressedColBlendVal](#) is faded over the MessageButton. Where 0 means transparent and 255 means opaque.



If you want to determine in the OnClick with which mouse button the event was triggered, you can do this with the public variable [MouseButton](#). The value in MouseButton is retained until it is overwritten again in the MouseDown procedure of the component.

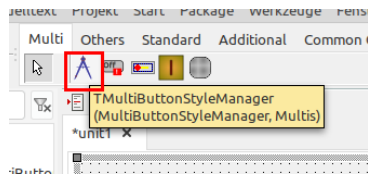
Example:

```
procedure TForm1.MultiButton1Click(Sender: TObject);
begin
  if MultiButton1.MouseButton = mbLeft then showmessage('Left');
  if MultiButton1.MouseButton = mbRight then showmessage('Right');
  if MultiButton1.MouseButton = mbMiddle then showmessage('Middle');
end;
```

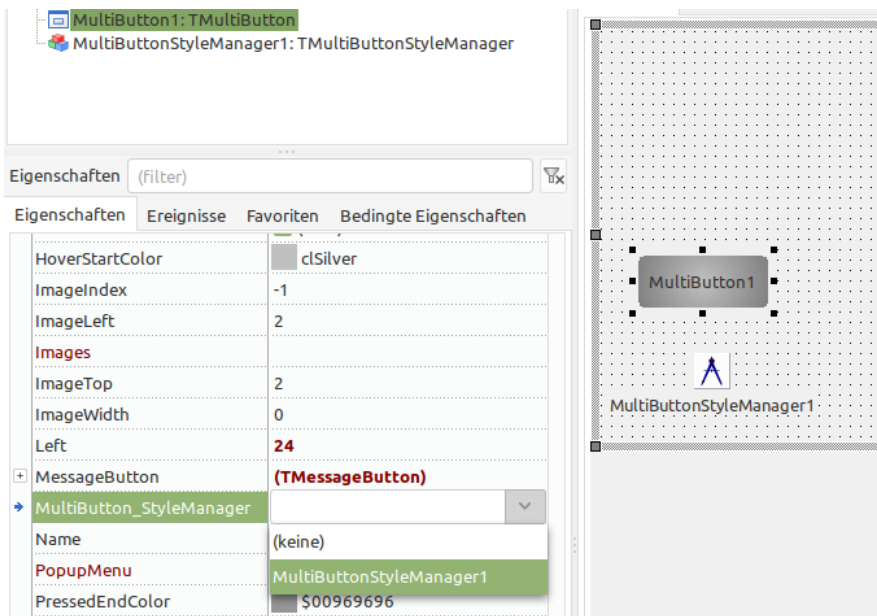
TMultiButtonStyleManager

If you are building a form with many MultiButtons that should look similar, we recommend using the MultiButtonStyleManager.

You can find it here:



Like any component, you simply drag it onto the form. To connect it to a MultiButton, you must now select the StyleManager in the [MultiButton_StyleManager](#) property (of the button).



The properties displayed in the Object Inspector under MultiButtonStyleManager now affect all connected MultiButtons simultaneously.



Here all six buttons are connected. By changing the style to mbsEllipse (in the OI under MBStyleManager) all MultiButtons change at once!

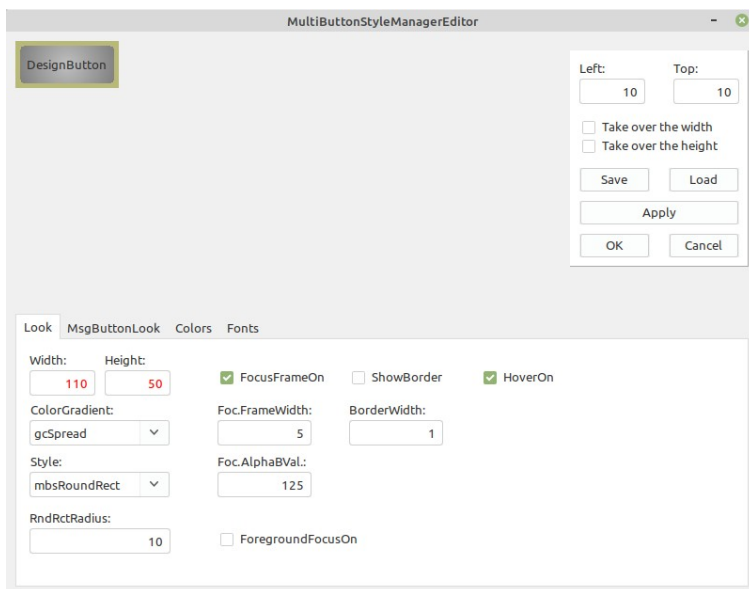
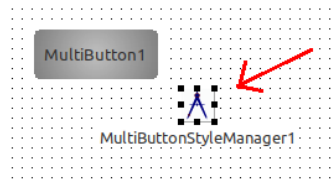
Attention! If you try to change a property of a connected MultiButton in the OI tab of the button (not the manager), this will fail!

For me, the procedure that has turned out to be good is that I first set all the desired properties of the MultiButtons with the Stylmanager and then remove the connection again. But that is certainly a matter of taste.

A special situation arises with the properties Width and Height. These two properties can only be changed via the style manager if the properties [OffsetHeight](#) and [OffsetWidth](#) of the style manager are deliberately set to true (default = false).

MultiButtonStylemanagerEditor

Furthermore, the MultiButtonStylemanager offers the possibility to make all settings in a MultiButtonStylemanagerEditor. This can be opened by double-clicking on the component symbol on the form.

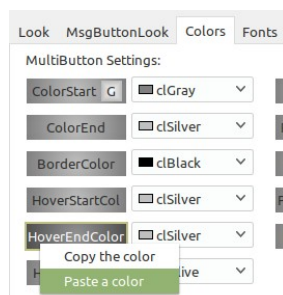
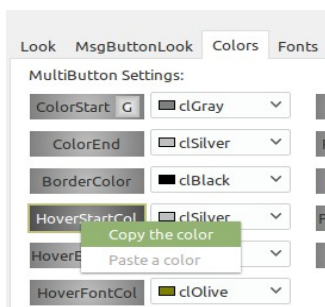


If you move the mouse over the various setting options, their functions are displayed as hints.

The button in the upper left corner serves as a pattern. It can be moved in the editor with *Left* and *Top*.

If you have created a design that you may want to use in another project, you can save it in a file by clicking *Save*. With *Load* you can retrieve it from there.

If you want to copy a colour from one selection to another, click with the right mouse button on the selection button, a pop-up opens and you can select copy or paste.



If you have connected several MultiButtons to a style manager and want to change the buttons via the style manager at runtime, you can use the public procedure [SetStyleManager](#).

This works like this, for example:

```
procedure TForm1.AdjustTheMultiButtons(Sender : TObject);
var lv : integer;
begin
  MultiButtonStyleManager1.ColorStart := clLime;
  MultiButtonStyleManager1.ColorEnd  := clRed;
  for lv := 0 to pred(ComponentCount) do
    if (Components[lv] is TMultiButton) then
      if TMultiButton(Components[lv]).MultiButton_StyleManager = MultiButtonStyleManager1 then
        TMultiButton(Components[lv]).SetStyleManager(MultiButtonStylemanager1);
  end;
```



TMultiplexSlider

Properties

Action	: The Action associated with the control
Align	: Specifies the placement of the control inside its Parent
Anchors	: The set of anchor definitions for this control
AutoRangeNegative	: If Min is reached, Min increases by AutoRangeValue
AutoRangePositive	: If Max is reached, Max increases by AutoRangeValue
AutoRangeValue	: Only active with AutoRangePositive and AutoRangeNegative
AutoSize	: Only active in conjunction with textlabel
BorderColor	: The color of the border (clNone makes invisible)
BorderSpacing	: Determines the inner and outer border spacing for this control
BorderWidth	: The width of the border
ColorEnd	: The end color of the slider (for color gradient)
ColorGradient	: The direction of the gradient
ColorStart	: The start color of the slider (for color gradient)
Constraints	: The minimum and maximum Width and Height for the control
Cursor	: The shape of the mouse pointer, when the mouse is over this control
DragCursor	: The cursor shape shown while the control is dragged
DragKind	: The operation when the control is dragged - Drag or Dock
DragMode	: Allows the user to drag the control
Enabled	: Determines whether the control reacts on mouse or keyboard input
FocusAlphaBValue	: How translucent the focusframe is (0=transparent, 255=opaque)
FocusColor	: The color of the Fokusframe/Foregroundfocus when the Control has the focus
FocusFrameOn	: Switches the focus frame on and off
FocusFrameWidth	: The whidth of the focus-frame
ForegroundFocusOn	: Indicates when the slider has focus, switches on off
Height	: The vertical size of the control
HelpContext	: The ID for context-sensitive Help on this control
HelpKeyword	: The keyword for context-sensitive Help on this control
HelpType	: Determines whether context-sensitive Help is selected by numeric ID or keyword
Hint	: The text to show in the Hint window for the control
JumpToPosition	: If active, the button jumps to position when clicked in the track
Knob1Settings	: The properties of the Knobs
Knob1Settings.ColorGradient	: The direction of the gradient
Knob1Settings.Design	: The appearance of the knob
Knob1Settings.DesignColor	: The color of the border or characters in the knob

<code>Knob1Settings.HoverEndColor</code>	: The endcolor of a hoverevent
<code>Knob1Settings.HoverOn</code>	: Allows to show or hide a hoverevent
<code>Knob1Settings.HoverStartColor</code>	: The startcolor of a hoverevent
<code>Knob1Settings.KnobColorEnd</code>	: The end color of the knob (for color gradient)
<code>Knob1Settings.KnobColorStart</code>	: The start color of the knob (for color gradient)
<code>Knob1Settings.KnobPosition</code>	: The Position of the Knob in the Slider
<code>Knob1Settings.KnobStyle</code>	: The shape of the knob
<code>Knob1Settings.Visible</code>	: Shows the Knob
<code>Knob2Settings</code>	: The properties of the Knobs
<code>Knob2Settings.ColorGradient</code>	: The direction of the gradient
<code>Knob2Settings.Design</code>	: The appearance of the knob
<code>Knob2Settings.DesignColor</code>	: The color of the border or characters in the knob
<code>Knob2Settings.HoverEndColor</code>	: The endcolor of a hoverevent
<code>Knob2Settings.HoverOn</code>	: Allows to show or hide a hoverevent
<code>Knob2Settings.HoverStartColor</code>	: The startcolor of a hoverevent
<code>Knob2Settings.KnobColorEnd</code>	: The end color of the knob (for color gradient)
<code>Knob2Settings.KnobColorStart</code>	: The start color of the knob (for color gradient)
<code>Knob2Settings.KnobPosition</code>	: The Position of the Knob in the Slider
<code>Knob2Settings.KnobStyle</code>	: The shape of the knob
<code>Knob2Settings.Visible</code>	: Shows the Knob
<code>Knob3Settings</code>	: The properties of the Knobs
<code>Knob3Settings.ColorGradient</code>	: The direction of the gradient
<code>Knob3Settings.Design</code>	: The appearance of the knob
<code>Knob3Settings.DesignColor</code>	: The color of the border or characters in the knob
<code>Knob3Settings.HoverEndColor</code>	: The endcolor of a hoverevent
<code>Knob3Settings.HoverOn</code>	: Allows to show or hide a hoverevent
<code>Knob3Settings.HoverStartColor</code>	: The startcolor of a hoverevent
<code>Knob3Settings.KnobColorEnd</code>	: The end color of the knob (for color gradient)
<code>Knob3Settings.KnobColorStart</code>	: The start color of the knob (for color gradient)
<code>Knob3Settings.KnobPosition</code>	: The Position of the Knob in the Slider
<code>Knob3Settings.KnobStyle</code>	: The shape of the knob
<code>Knob3Settings.Visible</code>	: Shows the Knob
<code>Left</code>	: The client coordinate of the left edge of the control
<code>Max</code>	: The highest value in range
<code>Min</code>	: The lowest value in range
<code>Orientation</code>	: The orientation of the Slider
<code>PopupMenu</code>	: A context-sensitive menu that pops up when the right mouse button is clicked

Reversed	: Max and min are swapped
RndRctRadius	: Corner diameter if the geometric shape is RoundRect
Scale1Settings	: The properties of the first scale
Scale1Settings.BigMarkColor	: The color of the big marks
Scale1Settings.BigMarkInterval	: The distance of the big marks
Scale1Settings.BigMarksVisible	: Shows big marks
Scale1Settings.LineColor	: The color of the lines in the scale
Scale1Settings.LineWidth	: The whidth of the scalelines
Scale1Settings.ScaleStyle	: The appearance of the markings (ssNone makes invisible)
Scale1Settings.SmallMarkColor	: The color of the Marks in the scale
Scale1Settings.SmallMarkInterval	: The distance of marks in the scale
Scale2Settings	: The properties of the second scale
Scale2Settings.BigMarkColor	: The color of the big marks
Scale2Settings.BigMarkInterval	: The distance of the big marks
Scale2Settings.BigMarksVisible	: Shows big marks
Scale2Settings.LineColor	: The color of the lines in the scale
Scale2Settings.LineWidth	: The whidth of the scalelines
Scale2Settings.ScaleStyle	: The appearance of the markings (ssNone makes invisible)
Scale2Settings.SmallMarkColor	: The color of the Marks in the scale
Scale2Settings.SmallMarkInterval	: The distance of marks in the scale
ShowHint	: Enables the Hint display
Style	: The geometric shape of the slider
TabOrder	: Determines the sequence of control navigation when the user presses the Tab key
TabStop	: Allows the user to navigate to this control, by pressing the Tab key
TextSettings	: The properties of the textlabel
TextSettings.AdInPercent	: Shows the value of the slider in the textLabel in percent
TextSettings.AutoAd	: Shows the value of the slider in the TextLabel
TextSettings.BackgrdColor	: The backgroundcolor of the textlabel (clNone for no color)
TextSettings.BorderColor	: The color of the border (clNone for invisible)
TextSettings.BorderWidth	: The width of the border
TextSettings.CaptionAlignment	: Alignment of the text in the caption (left, center, right)
TextSettings.CaptionBelow	: Write the letters one below the other (only active poLeft and poRight)
TextSettings.CaptionHorMargin	: The horizontal distance of the text in the text rectangle (only effective with taLeftJustify)
TextSettings.CaptionLayout	: Alignment of the text in the caption (top, center, bottom)
TextSettings.CaptionVerMargin	: The vertical distance of the text in the text rectangle (only effective with tlTop)
TextSettings.Font	: The font to be used for textlabel

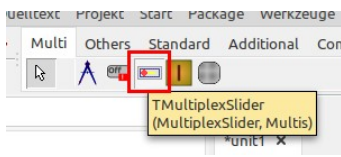
<code>TextSettings.Height</code>	: The height of the TextLabel (only effective with poTop/poBottom)
<code>TextSettings.Position</code>	: The position of the textlabel (poNone makes invisible)
<code>TextSettings.PostCaption</code>	: The text behind the value in the textlabel
<code>TextSettings.PreCaption</code>	: The text in front of the value in the textlabel
<code>TextSettings.Style</code>	: The geometric shape of the textlabel
<code>TextSettings.Width</code>	: The width of the TextLabel (only effective with poLeft/poRight)
<code>Top</code>	: The client coordinate of the top edge of the control
<code>TrackSettings</code>	: The properties of the track
<code>TrackSettings.ExtraColor</code>	: The color of the additional color (clNone for invisible)
<code>TrackSettings.ExtraRangeMax</code>	: The max Value of the additional color
<code>TrackSettings.ExtraRangeMin</code>	: The min Value of the additional color
<code>TrackSettings.SelRangeColor</code>	: The color of the selected area (clNone for invisible)
<code>TrackSettings.TrackColor</code>	: The color of the track
<code>ValueDisplaySettings</code>	: The properties of the ValueDisplay
<code>ValueDisplaySettings.BorderColor</code>	: The color of the border (clNone for invisible)
<code>ValueDisplaySettings.BorderWidth</code>	: The width of the border
<code>ValueDisplaySettings.ColorEnd</code>	: The end color of the display (for color gradient)
<code>ValueDisplaySettings.ColorGradient</code>	: The direction of the gradient
<code>ValueDisplaySettings.ColorStart</code>	: The start color of the display (for color gradient)
<code>ValueDisplaySettings.Font</code>	: The font to be used for display
<code>ValueDisplaySettings.InPercent</code>	: Shows the value in percent
<code>ValueDisplaySettings.Position</code>	: The position of the display in the slider, vdsNone makes invisible
<code>ValueDisplaySettings.Style</code>	: The geometric shape of the display, vdsNone makes no shape
<code>ValueDisplaySettings.X</code>	: affects the position, only to be used with vdpXY,vdpAboveRight,vdpBelowLeft
<code>ValueDisplaySettings.Y</code>	: affects the position, only to be used with vdpXY,vdpAboveRight,vdpBelowLeft
<code>Visible</code>	: Allows the control, and all of its children, to be displayed or hidden
<code>Width</code>	: The horizontal extent of the control

Events

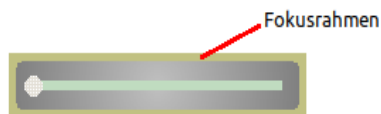
OnChange	: Returns the value of Knob1 (as integer)
OnChangeStr	: Returns the value of Knob1 as a string
OnChange3x	: Returns the values of Knob1,2,3 (as integer)
OnChange3xPlus	: Returns the values of Knob1,2,3 (as integer)+KnobIndex
OnChangeStr3x	: Returns the values of Knob1,2,3 as a string
OnClick	: Notification handler for mouse clicks
OnDragDrop	: This handler determines the action on an drop onto this control, in a drag-drop operation
OnDragOver	: Event handler for a control being dragged over this control
OnEndDrag	: Notification handler for the end of a dragging operation
OnEnter	: Handler for control receiving the focus
OnExit	: Handler for control loosing the focus; This is a good place for checking the finished user input
OnKeyDown	: Handler for keyboard key pressed
OnKeyPress	: Handler for a character entered by the user
OnKeyUp	: Handler for keyboard key released
OnMouseDown	: Event handler for mouse button going down
OnMouseEnter	: Event handler for mouse entering the area of the control
OnMouseLeave	: Event handler for mouse leaving the area of the control
OnMouseMove	: Event handler for mouse movement within the control
OnMouseUp	: Event handler for mouse button going up
OnMouseUp3xPlus	: Returns the values of Knob1,2,3 (as integer)+KnobIndex at MouseUp
OnMouseWheelDown	: Event handler for downward movement of mouse wheel
OnMouseWheelUp	: Event handler for upward movement of the mouse wheel
OnStartDrag	: Event handler for the start of a dragging operation

Description

You will find the MultiplexSlider in the Multi tab:

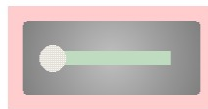


It is important to know that the MultiplexSlider is surrounded by a focus frame. As you can see here, the focused MultiplexSlider has an olive green frame. This means that the actual slider is smaller around the frame.

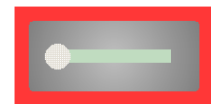


The [FocusColor](#) property can be used to set the colour of the focus frame. With [FocusAlphaBValue](#) the transparency of the focus frame can be controlled. The value 0 means transparent and 255 opaque.

Value 50:



Value 200:



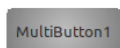
[FocusFrameWidth](#) determines the thickness of the frame.

If [FocusFrameOn](#) is set to false, the border is retained but the focus is not shown in colour.

With [ForegroundFocusOn](#), the focused MultiplexSlider has a coloured border. The colour of the border can be influenced with [FocusColor](#). This setting only makes sense if [FocusFrameOn](#) is set to false!



Here the red border shows that the slider has the focus.



To change the background colour of the MultiplexSlider you need the properties [ColorStart](#), [ColorEnd](#) and [ColorGradient](#). To get a single-coloured MultiplexSlider, ColorStart and ColorEnd must be the same. Otherwise, the composition of the three properties determines the colour appearance.

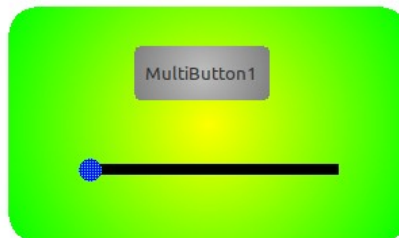
Here StartColor is clBlack and EndColor is clFuchsia.



Tip: If you set ColorStart and ColorEnd to clNone, the background gets the colour of the parent.



If the slider is sitting on a MultiPanel it takes on the colour of that panel (for clNone).



The [Style](#) property is used to set the desired geometric shape of the MultiplexSlider.

mssRect



If mssRoundRect is set, you can set the diameter of the corner rounding with the property [RndRectRadius](#).

mssRoundRect



RndRectRadius :=10;

mssRoundRect



RndRectRadius := 40;

If you want to add a coloured border to the MultiplexSlider, select the colour of the border in [BorderColor](#). No border is drawn with clNone. The width of the border is set with [BorderWidth](#).

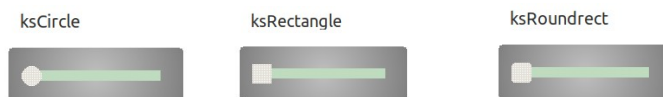


The properties of the knobs can be found under [Knob1Settings](#), [Knob2Settings](#) and [Knob3Settings](#).

While the first knob is visible by default, the second and third are invisible. [Knob1Settings.Visible](#), [Knob2Settings.Visible](#) and [Knob3Settings.Visible](#) are used to make the knobs visible or invisible.



The [Knob1Settings.KnobStyle](#), [Knob2Settings.KnobStyle](#) or [Knob3Settings.KnobStyle](#) property is used to set the desired geometric shape of the knob.

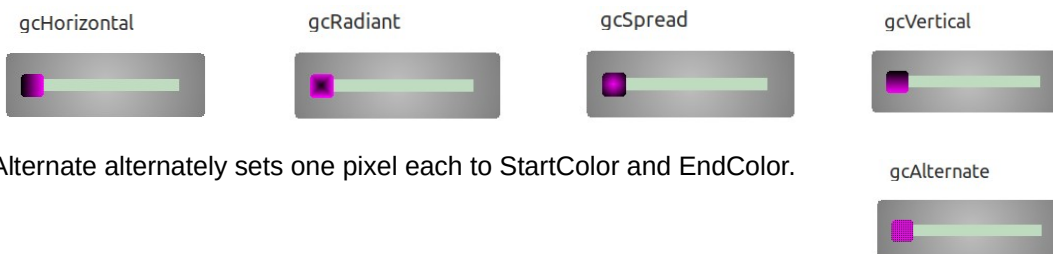


To change the background colour of the knob you need the properties

[Knob1Settings.KnobColorStart](#), [Knob2Settings.KnobColorStart](#) or [Knob3Settings.KnobColorStart](#)
[Knob1Settings.KnobColorEnd](#), [Knob2Settings.KnobColorEnd](#) or [Knob3Settings.KnobColorEnd](#) und
[Knob1Settings.ColorGradient](#), [Knob2Settings.ColorGradient](#) or [Knob3Settings.ColorGradient](#)

To get a unicoloured knob, ColorStart and ColorEnd must be the same. Otherwise, the composition of the three properties determines the colour appearance.

Here StartColor is clBlack and EndColor is clFuchsia.



Alternate alternately sets one pixel each to StartColor and EndColor.

By default, [Knob1Settings.HoverOn](#), [Knob2Settings.HoverOn](#) and [Knob3Settings.HoverOn](#) are set to true. This means that when a hover event occurs (the mouse moves over the knob) the appearance can be changed as desired with

[Knob1Settings.HoverStartColor](#), [Knob2Settings.HoverStartColor](#) or [Knob3Settings.HoverStartColor](#) and
[Knob1Settings.HoverEndColor](#), [Knob2Settings.HoverEndColor](#) or [Knob3Settings.HoverEndColor](#).

If you do not want this, set HoverOn to false.

[Knob1-3Settings.HoverEndColor](#) : The end colour of a hover event.

[Knob1-3Settings.HoverOn](#) : Allows you to show or hide a hover event.

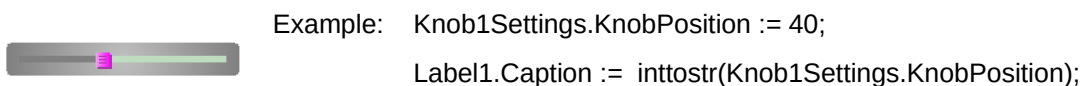
[Knob1-3Settings.HoverStartColor](#) : The start colour of a hover event.

Another possibility to add visual effects to the knobs are the properties [Knob1Settings.Design](#), [Knob2Settings.Design](#) or [Knob3Settings.Design](#) and [Knob1Settings.DesignColor](#), [Knob2Settings.DesignColor](#) or [Knob3Settings.DesignColor](#).

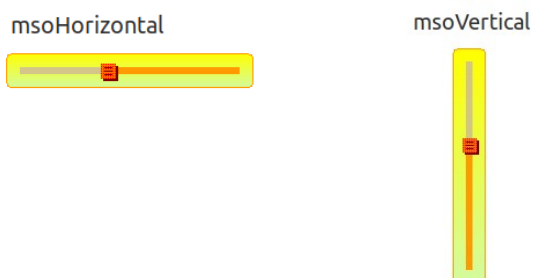
In the examples, DesignColor is clyellow. DesignColor is not used for kdDefault.



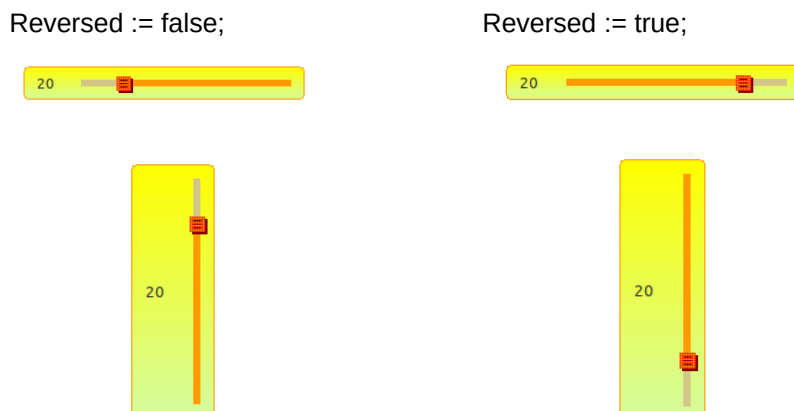
The [Knob1Settings.KnobPosition](#), [Knob2Settings.KnobPosition](#) or [Knob3Settings.KnobPosition](#) property can be used to set or query the position of the knob within the slider.



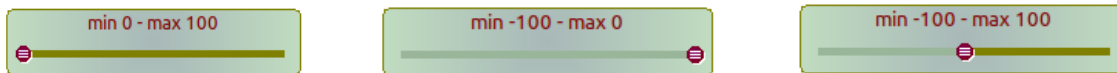
The orientation of the MultiplexSlider is achieved by means of [Orientation](#).



If the [Reversed](#) property is used, the max. value and the min. value are reversed.



The property **Max** represents the highest value in the value range of the MultiplexSlider and **Min** the lowest. Here with knob position 0.



If you are not yet sure about the value range of the MultiplexSlider or if you need a dynamic value range, you can use the properties **AutoRangeNegative** or **AutoRangePositive** and **AutoRangeValue**.

For example, if you want the max. value to increase by 20 units when it is reached, set the following:

AutoRangeNegative	<input type="checkbox"/> (False)
AutoRangePositive	<input checked="" type="checkbox"/> (True)
AutoRangeValue	20

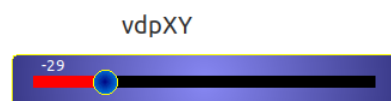
When the knob in the slider reaches the maximum value, it is increased by 20. If the knob reaches the max. value again, it is increased again. If the value falls below the original maximum value, the original value range is restored.

If you want to display the set value of the slider, you can use the properties of **ValueDisplaySettings**. To activate the display, a type other than **vdpNone** must be selected for **ValueDisplaySettings.Position**.



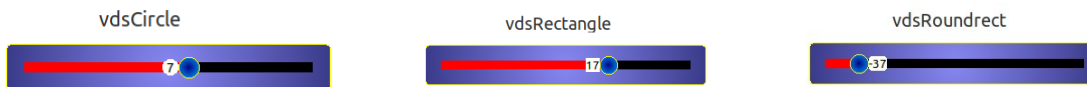
With the additional properties **ValueDisplaySettings.X** and **ValueDisplaySettings.Y**, the position of the display can be influenced for **vdpAboveRight**, **vdpBelowLeft**.

With **vdpXY**, the position of the display is freely selected in the slider with these properties.



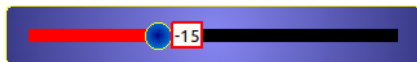
If the display is to have a coloured background, this is realised with the properties [ValueDisplaySettings.Style](#), [ValueDisplaySettings.ColorStart](#), [ValueDisplaySettings.ColorEnd](#) and [ValueDisplaySettings.ColorGradient](#).

The shape of the background is set with [Style](#). If [vdsNone](#) is selected, no background is drawn.



[ColorStart](#), [ColorEnd](#) and [ColorGradient](#) work in the same way as setting the background. See [31](#)

A coloured border around the display is created with [BorderColor](#) (cINone for invisible) and [Border.Width](#).

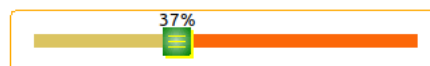


[ValueDisplaySettings.BorderColor](#) := cIRed;

[ValueDisplaySettings.BorderWidth](#) := 2;

With [ValueDisplaySettings.Font](#), you can set all attributes of the font, such as colour and size, as usual.

If you need the value to be displayed as a percentage, you can simply set [ValueDisplaySettings.InPercent](#) to true.



All available properties of the track can be set in the [TrackSettings](#).



[TrackSettings.TrackColor](#)

: The colour of the track

[TrackSettings.SelRangeColor](#)

: The colour of the selected area (cINone for invisible).

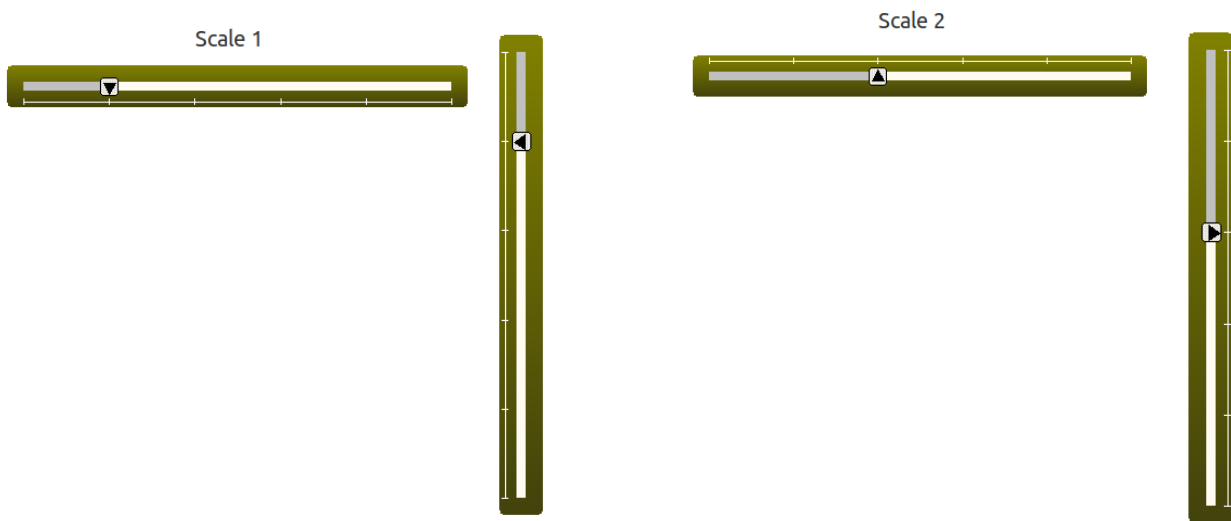
If you want to display an additional colour, you can do this by assigning a colour to the property [TrackSettings.ExtraColor](#) (cINone for invisible). With the properties [TrackSettings.ExtraRangeMax](#) and [TrackSettings.ExtraRangeMin](#) you determine the limits of this colour.



This slider has a value range of 0-100. [ExtraRangeMin](#) is set to 80 and [Max](#) to 100.

[ExtraColor](#) is cIRed.

The MultiplexSlider has two integrated scales. The settings for these can be found in the properties of [Scale1Settings](#) and [Scale2Settings](#). If you set [Scale1Settings.ScaleStyle](#) or [Scale2Settings.ScaleStyle](#) to `ssDash` or `ssCircle`, the respective scale becomes visible (`ssNone` makes it invisible).



[Scale1Settings.ScaleStyle](#) or [Scale2Settings.ScaleStyle](#):



[Scale1Settings.LineColor](#), [Scale2Settings.LineColor](#) : here `clRed`
[Scale1Settings.LineWidth](#), [Scale2Settings.LineWidth](#) : The thickness of the lines of the scale, here 2
[Scale1Settings.SmallMarkColor](#), [Scale2Settings.SmallMarkColor](#) : here `clLime`

With [Scale1Settings.SmallMarkInterval](#) or [Scale2Settings.SmallMarkInterval](#) the distance between the small marks can be set.



Value range here 0-100. Interval 10.

Thickness marks can also be displayed for better recognition. This is done with [Scale1Settings.BigMarksVisible](#) or [Scale2Settings.BigMarksVisible](#) (set to true).

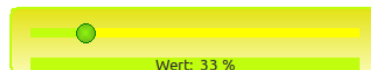
Here [Scale1Settings.BigMarkColor](#) or [Scale2Settings.BigMarkColor](#) `clYellow` and [Scale1Settings.BigMarkInterval](#) or [Scale2Settings.BigMarkInterval](#) 20.



If you want to use the built-in text label, you have to set the property [TextSettings.Position](#) under [TextSettings](#) to a value other than poNone (poNone makes invisible). If you set [TextSettings.AutoAd](#) to true, the value is automatically displayed in the text label. If you want to add more text to the label, you can do this with the properties [TextSettings.PreCaption](#) and [TextSettings.PostCaption](#). What is entered in PreCaption appears in front of the value, what is entered in PostCaption appears behind it.



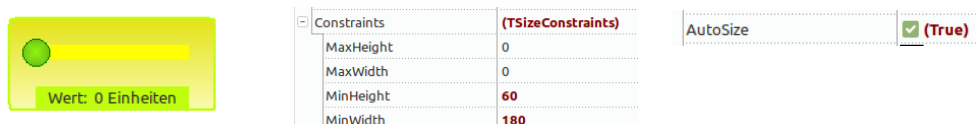
The [TextSettings.AdInPercent](#) property displays the value of the slider as a percentage.



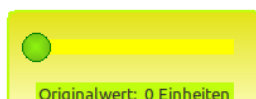
If the text label is too short, the width can be adjusted with [TextSettings.Width](#) (only effective with poLeft/poRight).

The same applies to the height, where [TextSettings.Height](#) must be used (only effective with poTop/poBottom).

The [AutoSize](#) property extends or shortens the entire slider to the necessary size of the text label. If the slider is shortened, the use of [Constraints](#) makes sense.



If the text in the text label is extended for any reason, the length is adjusted here:



The background of the text label can be set with the properties [TextSettings.BackgrdColor](#) (clNone for no colour), [TextSettings.BorderColor](#) (clNone for invisible), [TextSettings.BorderWidth](#) and [TextSettings.Style](#).

mssRect

Zoom: 32 %

mssRoundRect

Zoom: 44 %

BackgrdColor	<div></div> clCream
BorderColor	<div></div> clBlack
BorderWidth	1

Zoom: 72 %

BackgrdColor	<div></div> clNone
BorderColor	<div></div> clNone

The position of the text within the TextLabel is determined by the properties

- [TextSettings.CaptionAlignment](#) : Alignment of the text in the caption (left,centre,right)
- [TextSettings.CaptionHorMargin](#) : The horizontal spacing of the text in the text rectangle (only effective with taLeftJustify).
- [TextSettings.CaptionLayout](#) : Alignment of the text in the caption (Top,Middle,Bottom) [Text-](#)
- [Settings.CaptionVerMargin](#) : The vertical spacing of the text in the text rectangle (only effective with tlTop)

taLeftJustify

Zoom: 0 %

taCenter

Zoom: 0 %

taRightJustify

Zoom: 0 %

tlTop

27 %

tlCenter

47 %

tlCenter

21 %

If you want to write the text from bottom to top in a vertical slider and change the font, you can do this with [TextSettings.Font](#)

Zoom: 48 %

CaptionAlignment	taLeftJustify
CaptionBelow	<input type="checkbox"/> (False)
CaptionHorMargin	4

Font	(TFont)
CharSet	DEFAULT_CHARSET
Color	<div></div> clRed
Height	-15
Name	Ubuntu
Orientation	900

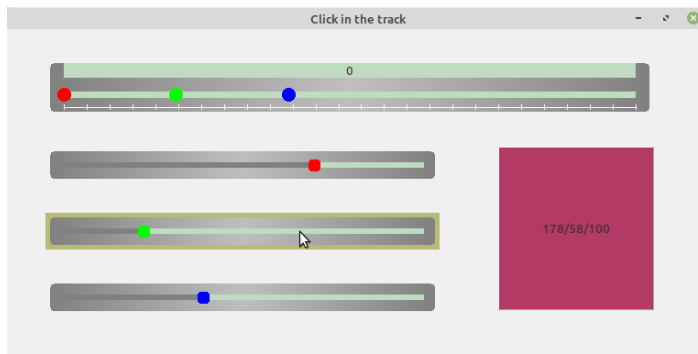
Zoom

0

%

If you want to display the letters below each other, you can use [TextSettings.CaptionBelow](#) (only active with poLeft and poRight).

If the [JumpToPosition](#) property is true, the knob jumps to this position when you click in the track. If two or three buttons are visible, the last button moved jumps.



When clicked here, the knob jumps to the position of the mouse pointer.

TMultiSeperator

Properties

Action	: The Action associated with the control
Align	: Specifies the placement of the control inside its Parent
Anchors	: The set of anchor definitions for this control
BackgrdImage	: Contains the image displayed in the control, only active with gcBitmap
BorderColor	: The color of the border, clNone makes invisible
BorderSpacing	: Determines the inner and outer border spacing for this control
BorderWidth	: The whidth of the border
ColorEnd	: The end color of the background (for color gradient),clNone makes unvisibel
ColorGradient	: The direction of the gradient
ColorStart	: The start color of the background (for color gradient),clNone makes unvisibel
Constraints	: The minimum and maximum Width and Height for the control
Cursor	: The shape of the mouse pointer, when the mouse is over this control
Enabled	: Determines whether the control reacts on mouse or keyboard input
Height	: The vertical size of the control
HelpContext	: The ID for context-sensitive Help on this control
HelpKeyword	: The keyword for context-sensitive Help on this control
HelpType	: Determines whether context-sensitive Help is selected by numeric ID or keyword
Hint	: The text to show in the Hint window for the control
Left	: The client coordinate of the left edge of the control
LineSettings	: The Settings of the line in the seperator
LineSettings.Color	: The color of the line
LineSettings.Design	: The number of lines
LineSettings.EndCap	: The shape of the line ends
LineSettings.LinesLength	: The lenght of the lines at dash,dashdot ...
LineSettings.LinesSpace	: The lenght of the space at dash,dashdot ...
LineSettings.Margin	: The distance from the line to the border
LineSettings.PenWidth	: The width of the line
LineSettings.Style	: The style of the line, cpsNull makes invisible
Orientation	: The orientation of the seperator
RndRctRadius	: Corner diameter if the geometric shape is RoundRect
ShowHint	: Enables the Hint display

Style

Top

Visible

Width

: The geometric shape of the separator

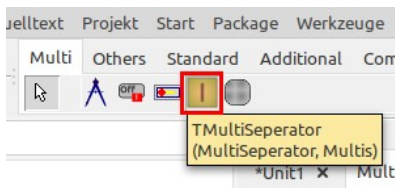
: The client coordinate of the top edge of the control

: Allows the control, and all of its children, to be displayed or hidden

: The horizontal extent of the control

Description

You can find the MultiSeparator in the Multi tab:



The MultiSeparator is a component that is intended for visual design. It can also be used as a separator between other controls.

With the [Orientation](#) property, the position of the separator can be switched between horizontal and vertical.

mvpVertical



mvpHorizontal



To change the background colour of the MultiSeparator you need the properties [ColorStart](#), [ColorEnd](#) and [ColorGradient](#). To get a single-coloured MultiSeparator, ColorStart and ColorEnd must be the same. Otherwise, the composition of the three properties determines the colour appearance.

Here StartColor is clMaroon and EndColor is \$0000B5FF.

gcSpread



gcRadiant



gcAlternate



Alternate alternately sets one pixel each to Start- and EndColor.

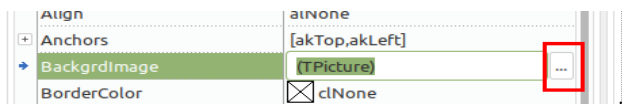
gcHorizontal



gcVertical



In order to be able to use the type gcBitmap, a bitmap with the property [BackgrdImage](#) must first be loaded. To do this, click in the box with the three dots at BackgrdImage in the OI.



The Load Image dialogue opens. Click on Load and select a bitmap (it also works with png). Finally, select gcBitmap under ColorGradient and the bitmap is used as the background.



The [Style](#) property is used to set the desired geometric shape of the MultiSeperator.



If mspRoundRect is set, you can set the diameter of the corner rounding with the property [RndRctRadius](#).

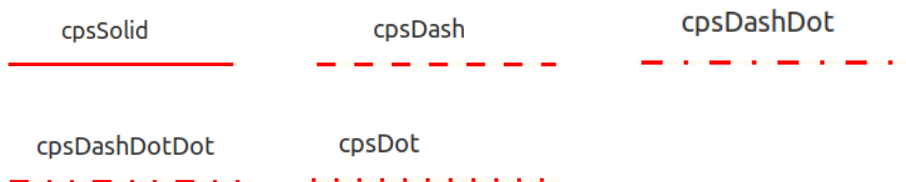
If [BorderColor](#) is set unequal to clNone, a border appears. Its thickness can be set with [BorderWidth](#).



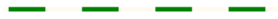
With the settings under [LineSettings](#), a line can be drawn in the separator.



By setting a [LineSettings.Style](#) (cpsNull makes invisible) the line becomes visible.



With [LineSettings.Color](#) the colour of the line can be influenced. [LineSettings.PenWidth](#) sets the thickness of the line.



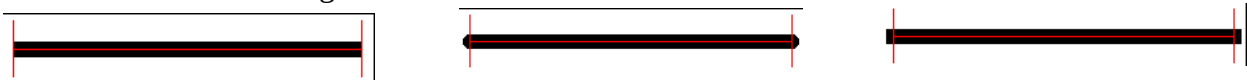
The length of the break between the lines can be set with [LineSettings.LinesSpace](#) and the length of the lines with [LineSettings.LinesLength](#).



With [LineSettings.EndCap](#) the end of the line can be adjusted:



For a better understanding of the line ends:



If you want to set the distance of the line left and right to the margin the same, this is done with [LineSettings.Margin](#).



With [LineSettings.Design](#) a double line can be created.



TMultiLayer

Properties

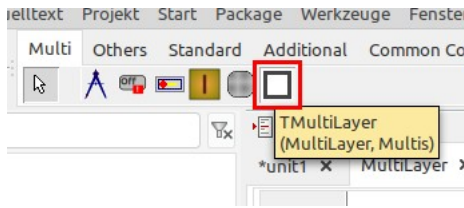
Align	: Specifies the placement of the control inside its Parent
Anchors	: The set of anchor definitions for this control
BorderSpacing	: Determines the inner and outer border spacing for this control
Color	: The background color of the control
Constraints	: The minimum and maximum Width and Height for the control
Cursor	: he shape of the mouse pointer, when the mouse is over this control
GroupIndex	: The index of the group to which the MultiLayer belongs
Height	: The vertical size of the control
HelpContext	: The ID for context-sensitive Help on this control
HelpKeyword	: The keyword for context-sensitive Help on this control.
HelpType	: Determines whether context-sensitive Help is selected by numeric ID or keyword
Hint	: The text to show in the Hint window for the control
Left	: The client coordinate of the left edge of the control
Top	: The client coordinate of the top edge of the control
Visible	: Allows the control, and all of its children, to be displayed or hidden, also at design time
Width	: The horizontal extent of the control

Public Procedures

procedure **InvalidateMultiPanel**;

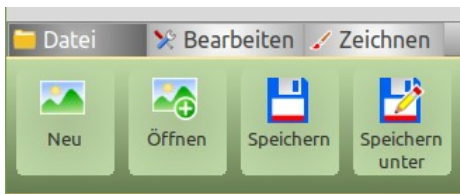
Description

You will find the MultiLayer in the Multi tab:



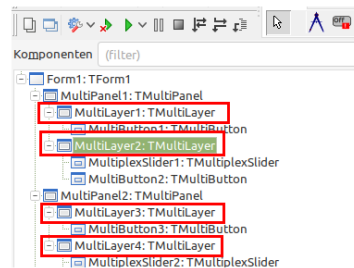
The MultiLayer is intended for simplified designing when clicking forms together. All child controls of TMultiLayer become invisible when TMultiLayer is set to [Visible := false](#). If the MultiLayer sits on a MultiPanel, its colour gradient is assumed as background.

Tip: If you place a MultiLayer on top of a MultiLayer, the first TWinControl behind it is taken as the parent!



Here, a TMultiLayer sits on a MultiPanel for each tab. By switching [Visible := true](#) to false, each tab can be designed as desired at design time.

Another way to switch layers during design time is to **double-click** on the MultiLayer you want to make visible in the Object Inspector treeview.



If MultiLayers have the same parent but are to be switched independently of each other, the [GroupIndex](#) must be set to different values.

If the background of the MultiPanel or other MultiControls is not redrawn in any constellation, a redrawing can be triggered with [InvalidateMultiPanel](#).

TMultiRadioGroup

Properties

Action	: The Action associated with the control
Align	: Specifies the placement of the control inside its Parent
Anchors	: The set of anchor definitions for this control
AutoSize	: Allows automatic adjustment of the size for the control, according to its content
BidiMode	: Customization (of text controls) in bidirectional reading environments
BorderColor	: The color of the border
BorderMargin	: The distance of the frame to the outer edge
BorderWidth	: The whidth of the border
BorderSpacing	: Determines the inner and outer border spacing for this control
Caption	: The headline of the radio group
CaptionLeft	: The coordinate of the left edge of the caption
CaptionTop	: The coordinate of the top edge of the caption
ColorEnd	: The end color of the RadioGroup (for color gradient)
ColorGradient	: The direction of the gradient
ColorStart	: The start color of the RadioGroup (for color gradient)
Constraints	: The minimum and maximum Width and Height for the control
Cursor	: The shape of the mouse pointer, when the mouse is over this control
DragCursor	: The cursor shape shown while the control is dragged
DragKind	: The operation when the control is dragged - Drag or Dock
DragMode	: Allows the user to drag the control
Enabled	: Determines whether the control reacts on mouse or keyboard input
FocusAlphaBValue	: How translucent the focusframe is (0=transparent, 255=opaque)
FocusColor	: The color of the Fokusframe/Foregroundfocus when the Control has the focus
FocusFrameOn	: Switches the focus frame on and off
FocusFrameWidth	: The whidth of the focus-frame
Font	: The font to be used for text display the caption
ForegroundFocusOn	: Indicates when the slider has focus, switches on off
GroupIndex	: The Index within the group of MultiRadioGroups
Height	: The vertical size of the control

HelpContext	: The ID for context-sensitive Help on this control
HelpKeyword	: The keyword for context-sensitive Help on this control
HelpType	: Determines whether context-sensitive Help is selected by numeric ID or keyword
Hint	: The text to show in the Hint window for the control
Left	: The client coordinate of the left edge of the control
RadioButtons	: Opens the editor to add radio buttons
RadioButtons.Item[Index].ButtonColor	: The color of the radiobutton
RadioButtons.Item[Index].ButtonSelColor	: The color of the selected radiobutton
RadioButtons.Item[Index].Caption	: The text that the user writes in the radiobutton
RadioButtons.Item[Index].CaptionAlignment	: Alignment of the text in the caption (left, center, right)
RadioButtons.Item[Index].CaptionHorMargin	: The horizontal distance of the text in the text rectangle (only effective with taLeftJustify)
RadioButtons.Item[Index].CaptionLayout	: Alignment of the text in the caption (top, center, bottom)
RadioButtons.Item[Index].CaptionVerMargin	: The vertical distance of the text in the text rectangle (only effective with tlTop)
RadioButtons.Item[Index].CaptionWordbreak	: Allows a line break in the caption
RadioButtons.Item[Index].Color	: The background colour of the RadioButton
RadioButtons.Items[Index].DisplayName	: The name that is displayed in the TreeView of the ObjectInspector
RadioButtons.Items[Index].Enabled	: Determines whether the control reacts on mouse or keyboard input
RadioButtons.Items[Index].Font	: The font to be used for text display the caption
RadioButtons.Item[Index].HoverColor	: The color of a hoverevent
RadioButtons.Item[Index].HoverStyle	: Whether a hover event is drawn as a frame only or full-surface
RadioButtons.Item[Index].ImageIndex	: The Index of a Image in a ImageList
RadioButtons.Item[Index].ImageLeft	: The coordinate of the left edge of a Image
RadioButtons.Item[Index].Images	: A list for including images
RadioButtons.Item[Index].ImageTop	: The coordinate of the top edge of a Image
RadioButtons.Items[Index].ImageWidth	: The unique width of all images in the list
RadioButtons.Items[Index].ParentFont	: Uses the font from the Parent when enabled
RadioButtons.Item[Index].Selected	: Determines if a radio button is selected
RadioButtons.Items[Index].Tag	: Can be used to store an integer value in the component
RndRctRadius	: Corner diameter if the geometric shape is RoundRect
Rows	: Number of lines when Wordbreak is active
Style	: The geometric shape of the RadioGroup

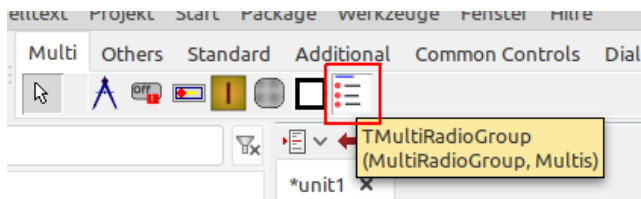
TabOrder	: Determines the sequence of control navigation when the user presses the Tab key
TabStop	: Allows the user to navigate to this control, by pressing the Tab key
Tag	: Can be used to store an integer value in the component
Top	: The client coordinate of the top edge of the control
Visible	: Allows the control, and all of its children, to be displayed or hidden
Width	: The horizontal extent of the control

Events

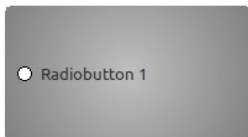
OnChange	: Returns the Index of the RadioButton
OnClick	: Notification handler for mouse clicks
OnDragDrop	: This handler determines the action on an drop onto this control, in a drag-drop operation
OnDragOver	: Event handler for a control being dragged over this control
OnEndDrag	: Notification handler for the end of a dragging operation
OnEnter	: Handler for control receiving the focus
OnExit	: Handler for control loosing the focus; This is a good place for checking the finished user input
OnKeyDown	: Handler for keyboard key pressed
OnKeyPress	: Handler for a character entered by the user
OnKeyUp	: Handler for keyboard key released
OnMouseDown	: Event handler for mouse button going down
OnMouseEnter	: Event handler for mouse entering the area of the control
OnMouseLeave	: Event handler for mouse leaving the area of the control
OnMouseMove	: Event handler for mouse movement within the control
OnMouseUp	: Event handler for mouse button going up
OnStartDrag	: Event handler for the start of a dragging operation

Description

The MultiRadioGroup can be found in the Multi tab:

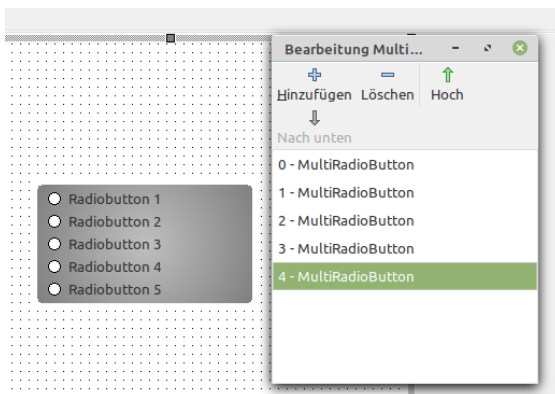


If you set the MultiRadioGroup to the form, it has only one RadioButton.



This one radio button cannot be deleted either!

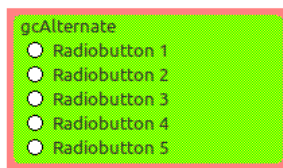
To add more radio buttons, click on [RadioButtons](#) in the Object Inspector. Now an editor opens in which further radio buttons can be created.



In the TreeView of the Object Inspector, you can now see the individual MultiRadioButtons under the MultiRadioGroup.



It is important to know that the MultiRadioGroup is surrounded by a focus frame. As you can see here, the focused MultiRadioGroup has a red frame. This means that the actual RadioGroup is smaller by the frame.

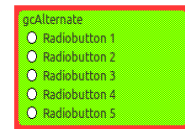


The [FocusColor](#) property can be used to set the colour of the focus frame. With [FocusAlphaBValue](#) the transparency of the focus frame can be controlled. The value 0 means transparent and 255 opaque. [FocusFrameWidth](#) determines the thickness of the frame.

Value 50:

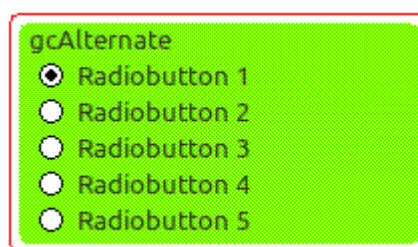


Value 200:

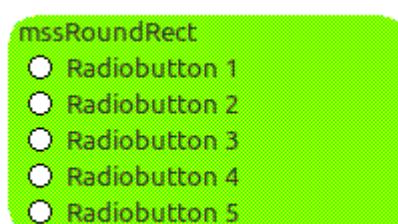
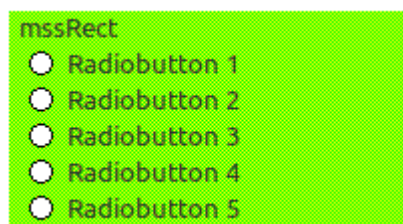


If [FocusFrameOn](#) is set to false, the border is retained but the focus is not displayed in colour.

With [ForegroundFocusOn](#) the focused MultiRadioGroup has a border. The colour of the border can be influenced with [FocusColor](#). With [FocusFrameWidth](#) the distance to the actual RadioGroup can be set. It can be useful to set [FocusFrameOn](#) to false.

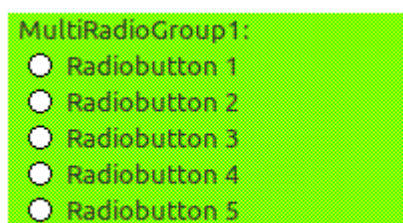


The [Style](#) property is used to set the desired geometric shape of the MultiRadioGroup.



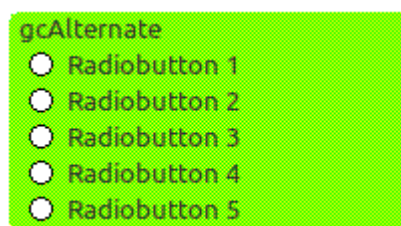
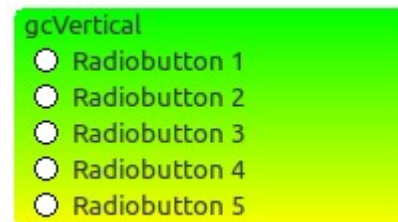
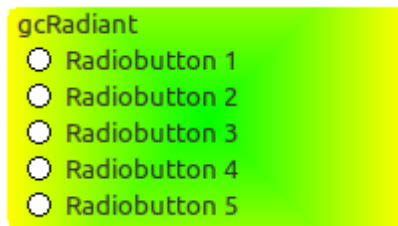
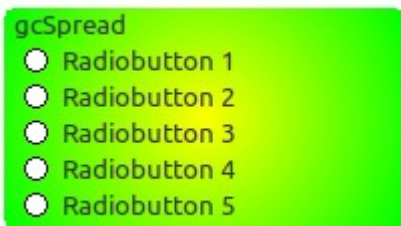
If [mssRoundRect](#) is set, you can set the diameter of the corner rounding with the property [RndRctRadius](#).

With [Caption](#) you can add a heading to the MultiRadioGroup. See also [56](#)



The background can be influenced with the properties [ColorStart](#), [ColorEnd](#) and [ColorGradient](#).

Here [ColorStart](#) is [clLime](#) and [ColorEnd](#) is [clYellow](#):

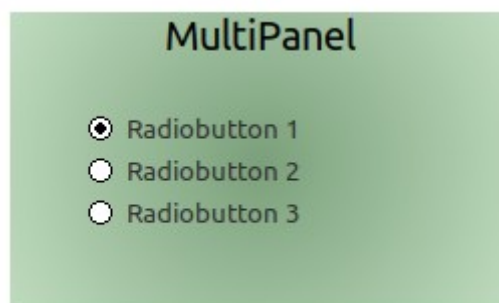


Alternate alternately sets a pixel to Start or EndColor.

If you set either ColorStart or ColorEnd to cNone, the background becomes transparent.

MultiRadioGroup1:

- ☐ Radiobutton 1
- ☐ Radiobutton 2
- ☐ Radiobutton 3
- ☐ Radiobutton 4
- ☐ Radiobutton 5



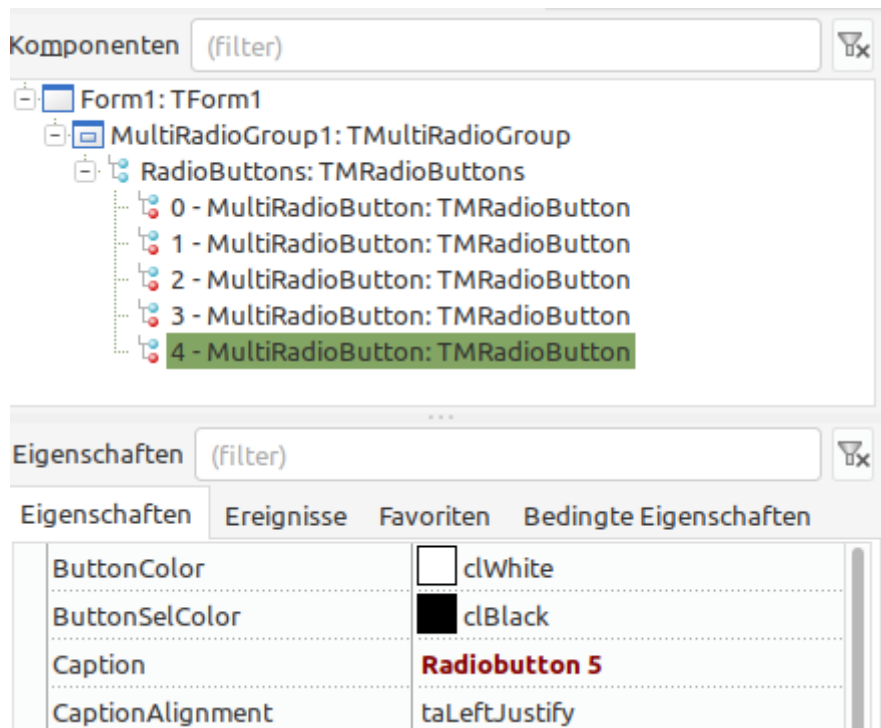
If the MultiRadioGroup sits on a MultiPanel, it becomes visible.

If the property [GroupIndex](#) is given a value other than 0, several MultiRadioGroups that are located on the same parent and have the same index can be connected together.



This allows any arrangement to be realised. You can navigate with the arrow keys, select with the space bar.

The settings for the individual MultiRadioButtons can be made separately for each button. The only exception is the font size. If this is changed for one button, it affects all other radio buttons. To easily access the view of the radio buttons, the component tree should be visible in the object inspector. If it is invisible, click with the right mouse button in the OI and tick Show component tree. Now you can simply select the individual MultiRadioButtons.



With the `RadioButtons.Items[Index].Font` settings (except the font size), different settings can be made for all buttons.



The `RadioButtons.Item[Index].ButtonColor` property is used to fill the background of the RadioButton.



Here Orange.

`RadioButtons.Item[Index].Selected` determines whether a RadioButton is selected. With `RadioButtons.Item[Index].ButtonSelColor` the colour of the selection can be determined.



Here Yellow.

With the property `RadioButtons.Item[Index].Caption` the text behind the RadioButton can be set. With `RadioButtons.Item[Index].CaptionAlignment`, `RadioButtons.Item[Index].CaptionHorMargin` the horizontal alignment of the text can be influenced.



With `RadioButtons.Item[Index].CaptionLayout` and `RadioButtons.Item[Index].CaptionVerMargin` the text can be aligned vertically.

With `RadioButtons.Item[Index].CaptionWordbreak` the text is wrapped if necessary.



However, the number of rows required must be specified with `Rows`.

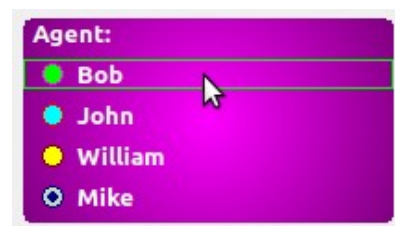
[RadioButtons.Item\[Index\].Color](#) sets the background of the RadioButton, `clNone` makes it transparent.



[RadioButtons.Item\[Index\].HoverColor](#) changes the colour of the background of the RadioButton when you move the mouse over it, `clNone` makes it transparent.



[RadioButtons.Item\[Index\].HoverStyle](#) can be used to select whether the hover event is to be fully filled or only displayed as a frame.



Images can also be inserted with the help of an image list. To do this, select the ImageList at [RadioButtons.Item\[Index\].Images](#). Select the desired image at [RadioButtons.Item\[Index\].ImageIndex](#) and determine the position of the image with [RadioButtons.Item\[Index\].ImageLeft](#) and [RadioButtons.Item\[Index\].ImageTop](#).

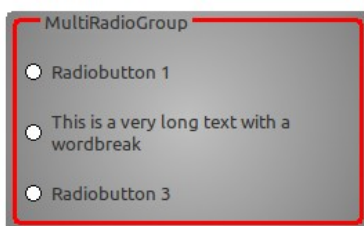


If you want to give the RadioGroup a border, change [BorderColor](#) to the desired colour. This means `clNone` makes the border invisible. With [BorderWidth](#) you can set the thickness of the border.



If you want to move the border inwards, you can do this with [BorderMargin](#). If you then move a caption over the border, the appearance of a GroupBox is created.

Moving the caption is done with [CaptionLeft](#) and [CaptionTop](#).



TMultiCheckGroup

Properties

Action	: The Action associated with the control
Align	: Specifies the placement of the control inside its Parent
Anchors	: The set of anchor definitions for this control
AutoSize	: Allows automatic adjustment of the size for the control, according to its content
BidiMode	: Customization (of text controls) in bidirectional reading environments
BorderColor	: The color of the border
BorderMargin	: The distance of the frame to the outer edge
BorderWidth	: The whidth of the border
BorderSpacing	: Determines the inner and outer border spacing for this control
Caption	: The headline of the checkgroup
CaptionLeft	: The coordinate of the left edge of the caption
CaptionTop	: The coordinate of the top edge of the caption
Checkboxes	: Opens the editor to add Checkboxes
CheckBoxes.Item[Index].ButtonColor	: The color of the checkbox
CheckBoxes.Item[Index].ButtonSelBackColor	: The backgroundcolor in the selected CheckBox
CheckBoxes.Item[Index].ButtonSelColor	: The color of the selected checkbox
CheckBoxes.Item[Index].Caption	: The text that the user writes in the checkbox
CheckBoxes.Item[Index].CaptionAlignment	: Alignment of the text in the caption (left, center, right)
CheckBoxes.Item[Index].CaptionHorMargin	: The horizontal distance of the text in the text rectangle (only effective with taLeftJustify)
CheckBoxes.Item[Index].CaptionLayout	: Alignment of the text in the caption (top, center, bottom)
CheckBoxes.Item[Index].CaptionVerMargin	: The vertical distance of the text in the text rectangle (only effective with tlTop)
CheckBoxes.Item[Index].CaptionWordbreak	: Allows a line break in the caption
CheckBoxes.Item[Index].Color	: The background colour of the checkbox
Checkboxes.Items[Index].DisplayName	: The name that is displayed in the TreeView of the ObjectInspector
CheckBoxes.Item[Index].Enabled	: Determines whether the control reacts on mouse or keyboard input
CheckBoxes.Items[Index].Font	: The font to be used for text display the caption
CheckBoxes.Item[Index].HoverColor	: The color of a hoverevent
CheckBoxes.Item[Index].HoverStyle	: Whether a hover event is drawn as a frame only or full-surface
CheckBoxes.Item[Index].ImageIndex	: The Index of a Image in a ImageList
CheckBoxes.Item[Index].ImageLeft	: The coordinate of the left edge of a Image
CheckBoxes.Item[Index].Images	: A list for including images
CheckBoxes.Item[Index].ImageTop	: The coordinate of the top edge of a Image

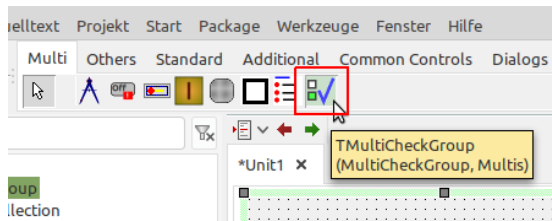
<code>CheckBoxes.Item[Index].ImageWidth</code>	: The unique width of all images in the list
<code>CheckBoxes.Items[Index].ParentFont</code>	: Uses the font from the Parent when enabled
<code>CheckBoxes.Item[Index].Selected</code>	: Determines if a checkbox is selected
<code>CheckBoxes.Item[Index].SelectedStyle</code>	: The character that is displayed in a selected box
<code>CheckBoxes.Items[Index].Tag</code>	: Can be used to store an integer value in the component
<code>ColorEnd</code>	: The end color of the checkGroup (for color gradient)
<code>ColorGradient</code>	: The direction of the gradient
<code>ColorStart</code>	: The start color of the checkGroup (for color gradient)
<code>Constraints</code>	: The minimum and maximum Width and Height for the control
<code>Cursor</code>	: The shape of the mouse pointer, when the mouse is over this control
<code>DragCursor</code>	: The cursor shape shown while the control is dragged
<code>DragKind</code>	: The operation when the control is dragged - Drag or Dock
<code>DragMode</code>	: Allows the user to drag the control
<code>Enabled</code>	: Determines whether the control reacts on mouse or keyboard input
<code>FocusAlphaBValue</code>	: How translucent the focusframe is (0=transparent, 255=opaque)
<code>FocusColor</code>	: The color of the Fokusframe/Foregroundfocus when the Control has the focus
<code>FocusFrameOn</code>	: Switches the focus frame on and off
<code>FocusFrameWidth</code>	: The whidth of the focus-frame
<code>Font</code>	: The font to be used for text display the caption
<code>ForegroundFocusOn</code>	: Indicates when the slider has focus, switches on off
<code>GroupIndex</code>	: The Index within the group of MultiCheckGroups
<code>Height</code>	: The vertical size of the control
<code>HelpContext</code>	: The ID for context-sensitive Help on this control
<code>HelpKeyword</code>	: The keyword for context-sensitive Help on this control
<code>HelpType</code>	: Determines whether context-sensitive Help is selected by numeric ID or keyword
<code>Hint</code>	: The text to show in the Hint window for the control
<code>Left</code>	: The client coordinate of the left edge of the control
<code>RndRctRadius</code>	: Corner diameter if the geometric shape is RoundRect
<code>Rows</code>	: Number of lines when Wordbreak is active
<code>Style</code>	: The geometric shape of the CheckGroup
<code>TabOrder</code>	: Determines the sequence of control navigation when the user presses the Tab key
<code>TabStop</code>	: Allows the user to navigate to this control, by pressing the Tab key
<code>Tag</code>	: Can be used to store an integer value in the component
<code>Top</code>	: The client coordinate of the top edge of the control
<code>Visible</code>	: Allows the control, and all of its children, to be displayed or hidden
<code>Width</code>	: The horizontal extent of the control

Events

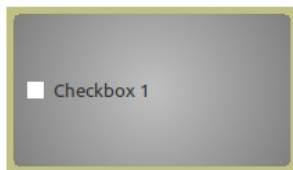
OnChange	: Returns the Index of the checkbox
OnClick	: Notification handler for mouse clicks
OnDragDrop	: This handler determines the action on an drop onto this control, in a drag-drop operation
OnDragOver	: Event handler for a control being dragged over this control
OnEndDrag	: Notification handler for the end of a dragging operation
OnEnter	: Handler for control receiving the focus
OnExit	: Handler for control loosing the focus; This is a good place for checking the finished user input
OnKeyDown	: Handler for keyboard key pressed
OnKeyPress	: Handler for a character entered by the user
OnKeyUp	: Handler for keyboard key released
OnMouseDown	: Event handler for mouse button going down
OnMouseEnter	: Event handler for mouse entering the area of the control
OnMouseLeave	: Event handler for mouse leaving the area of the control
OnMouseMove	: Event handler for mouse movement within the control
OnMouseUp	: Event handler for mouse button going up
OnStartDrag	: Event handler for the start of a dragging operation

Description

You can find the MultiCheckGroup in the Multi tab:

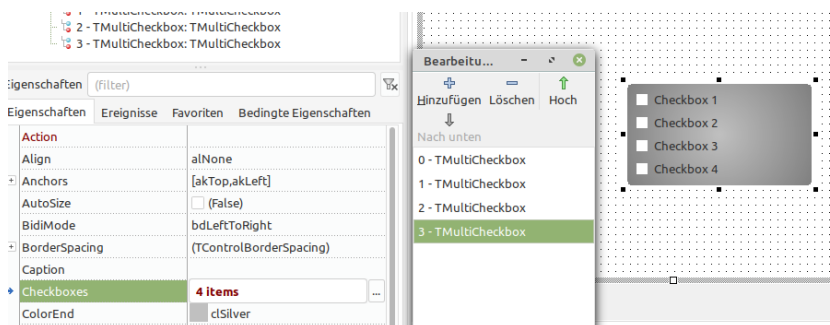


If you set the MultiCheckGroup to the form, it has only one CheckBox.

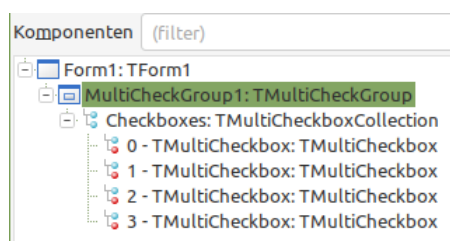


This one CheckBox cannot be deleted either!

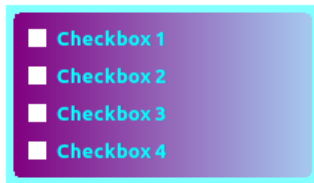
To add more CheckBoxes, click on [Checkboxes](#) in the Object Inspector. Now an editor opens in which further CheckBoxes can be created.



In the tree view of the Object Inspector, the individual MultiCheckBoxes can now be seen under the group MultiCheckGroup.

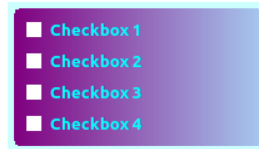


It is important to know that the MultiCheckGroup is surrounded by a focus frame. As you can see here, the focused MultiCheckGroup has a light blue frame. This means that the actual CheckGroup is smaller by the frame.

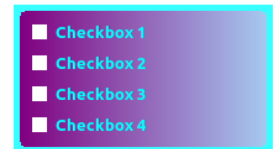


The [FocusColor](#) property can be used to set the colour of the focus frame. With [FocusAlphaBValue](#) the transparency of the focus frame can be controlled. The value 0 means transparent and 255 opaque. [FocusFrameWidth](#) determines the thickness of the frame.

Value 50:

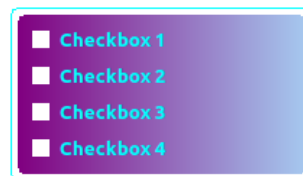


Value 200:

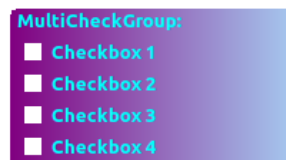


If [FocusFrameOn](#) is set to false, the border is retained but the focus is not displayed in colour.

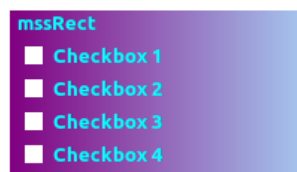
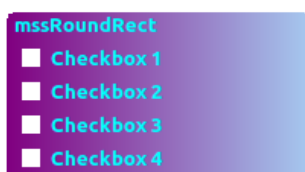
With [ForegroundFocusOn](#) the focused MultiCheckGroup has a border. The colour of the border can be influenced with [FocusColor](#). With [FocusFrameWidth](#) the distance to the actual CheckGroup can be set. It can be useful to set [FocusFrameOn](#) to false.



With [Caption](#) you can add a heading to the MultiCheckGroup. See also [66](#)



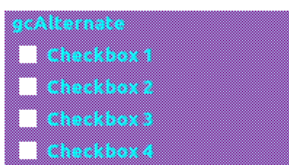
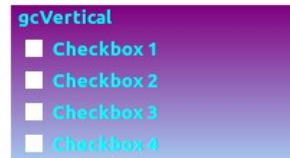
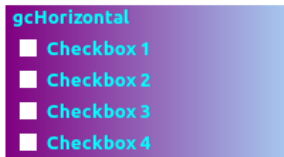
The [Style](#) property is used to set the desired geometric shape of the MultiCheckGroup.



If [mssRoundRect](#) is set, you can set the diameter of the corner rounding with the property [RndRctRadius](#).

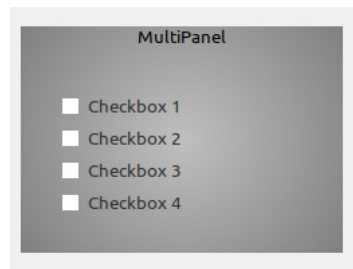
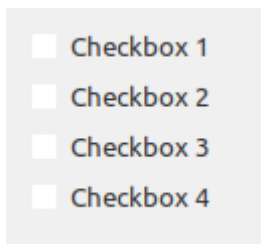
The background can be influenced with the properties [ColorStart](#), [ColorEnd](#) and [ColorGradient](#).

Here ColorStart is clPurple and ColorEnd is clSkyBlue:



Alternate alternately sets a pixel to Start or EndColor.

If you set either ColorStart or ColorEnd to clNone, the background becomes transparent.



If the MultiRadioGroup sits on a MultiPanel, it becomes visible.

FocusFrameOn should better be set to false.

If the property [GroupIndex](#) is given a value other than 0, several MultiCheckGroups that are located on the same parent and have the same index can be connected together.

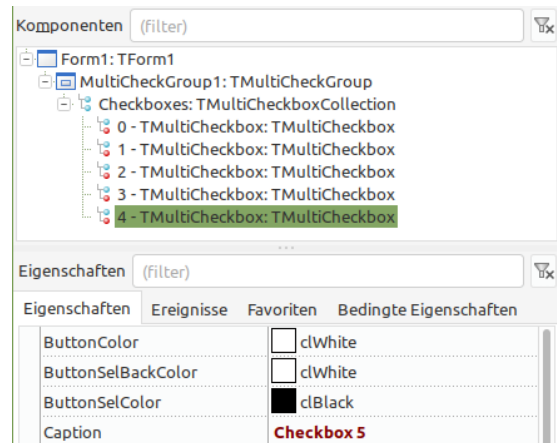
Form1

This is a MultiPanel with MultiCheckGroups

<input type="checkbox"/> Checkbox 1	<input checked="" type="checkbox"/> Checkbox 5	<input type="checkbox"/> Checkbox 8	<input type="checkbox"/> Checkbox 12
<input type="checkbox"/> Checkbox 2	<input type="checkbox"/> Checkbox 6	<input type="checkbox"/> Checkbox 9	<input type="checkbox"/> Checkbox 13
<input checked="" type="checkbox"/> Checkbox 3	<input type="checkbox"/> Checkbox 7	<input type="checkbox"/> Checkbox 10	
<input type="checkbox"/> Checkbox 4		<input checked="" type="checkbox"/> Checkbox 11	

This allows any arrangement to be realised. You can navigate with the arrow keys and select with the space bar.

The settings for the individual MultiCheckBoxes can be made separately for each checkbox. The only exception is the font size. If this is changed for one checkbox, it affects all other checkboxes. To easily access the view of the checkboxes, the component tree should be visible in the object inspector. If it is invisible, click with the right mouse button in the OI and tick Show component tree. Now you can simply select the individual checkboxes.



With the `Checkboxes.Items[Index].Font` settings (except the font size) different settings can be made for all buttons.



The property `CheckBoxes.Item[Index].ButtonColor` is used to fill the background of the CheckBox.



Here `clLime`.

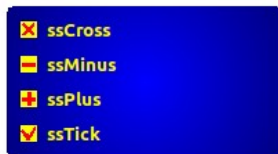
`CheckBoxes.Item[Index].Selected` determines whether a CheckBox is selected. With `CheckBoxes.Item[Index].ButtonSelColor` the colour of the selection can be determined and with `CheckBoxes.Item[Index].ButtonSelBackColor` the background of a selected CheckBox.



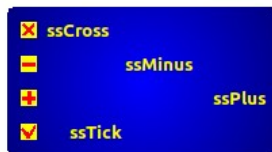
`ButtonSelColor` is here `clRed`.

`ButtonSelBackColor` is here `clYellow`.

`CheckBoxes.Item[Index].SelectedStyle` determines which character is drawn in a selected CheckBox.

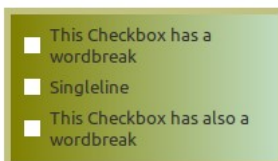


With the property `CheckBoxes.Item[Index].Caption` the text behind the CheckBox can be set. With `CheckBoxes.Item[Index].CaptionAlignment`, `CheckBoxes.Item[Index].CaptionHorMargin` the alignment of the text in the horizontal can be influenced.

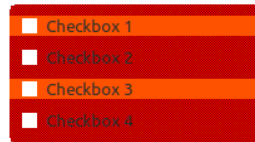


With `CheckBoxes.Item[Index].CaptionLayout` and `CheckBoxes.Item[Index].CaptionVerMargin` the text can be aligned vertically.

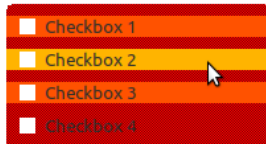
With `CheckBoxes.Item[Index].CaptionWordbreak` the text is wrapped if needed. However, the number of rows required must be specified with `Rows`.



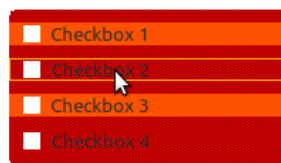
`CheckBoxes.Item[Index].Color` sets the background of the checkbox including the caption, `clNone` makes it transparent.



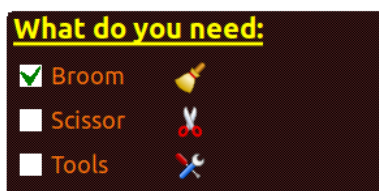
`CheckBoxes.Item[Index].HoverColor` changes the colour of the background of the RadioButton when you move the mouse over it, `clNone` makes it transparent.



With `CheckBoxes.Item[Index].HoverStyle` you can choose whether the hover event is fully filled or only displayed as a frame.

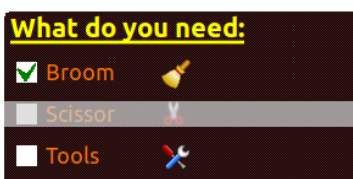


Images can also be inserted with the help of an image list. To do this, select the `ImageList` at `CheckBoxes.Item[Index].Images`. Select the desired image at `CheckBoxes.Item[Index].ImageIndex` and determine the position of the image with `CheckBoxes.Item[Index].ImageLeft` and `CheckBoxes.Item[Index].ImageTop`.



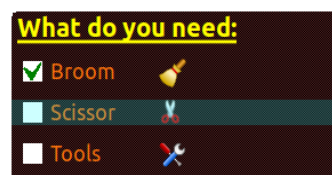
With `CheckBoxes.Item[Index].ImageWidth` the size of the image can be changed. It is recommended to scale only smaller!

If you set `CheckBoxes.Item[Index].Enabled` to false, a selection can be disabled.

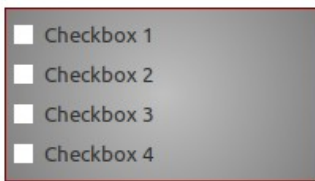


At runtime, the colour and transparency of the disabled bar can be set.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
    MultiCheckGroup1.Checkboxes.Items[1].DisabledColor:= clAqua;
    MultiCheckGroup1.Checkboxes.Items[1].DisabledAlphaBValue:= 50;
end;
```

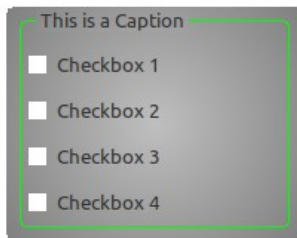


If you want to give the CheckGroup a border, change [BorderColor](#) to the desired colour. This means `clNone` makes the border invisible. With [BorderWidth](#) you can set the thickness of the border.



If you want to move the border inwards, you can do this with [BorderMargin](#). If you then move a caption over the border, the appearance of a GroupBox is created.

Moving the caption is done with [CaptionLeft](#) and [CaptionTop](#).



TMultiSwitch

Properties

Action	: The Action associated with the control
Align	: Specifies the placement of the control inside its Parent
Anchors	: The set of anchor definitions for this control
BestTextHeight	: Automatically adjusts the text height to the size of the control
BidiMode	: Customization (of text controls) in bidirectional reading environments
BorderColor	: The color of the border (clNone = no Border)
BorderSpacing	: Determines the inner and outer border spacing for this control
ButtonColor	: The color of the button, Rollimage only visible when ButtonColor = clNone
CaptionAlignment	: Alignment of the text in the caption (left, center, right)
CaptionHorMargin	: The horizontal distance of the text in the text rectangle (only effective with taLeftJustify)
CaptionLayout	: Alignment of the text in the caption (top, center, bottom)
CaptionVerMargin	: The vertical distance of the text in the text rectangle (only effective with tlTop)
Constraints	: The minimum and maximum Width and Height for the control
Cursor	: Contains the shape for the mouse pointer when the mouse is over the control
Direction	: Specifies whether the button is on the right or left at the start
DisabledColor	: The colour at Enable:=false, only at runtime!
DragCursor	: The cursor shape shown while the control is dragged
DragKind	: The operation when the control is dragged - Drag or Dock
DragMode	: Allows the user to drag the control
Enable	: Determines whether the control reacts on mouse or keyboard input
EnabledBlendFactor	: How translucent is the DisabledColor (1=opaque,0=transparent), only at runtime!
FocusAlphaBValue	: How translucent the focusframe is (0=transparent, 255=opaque)
FocusedBlendFactor	: How translucent is the focusColor (1=opaque,0=transparent), only at runtime!
FocusColor	: The color when the Control has the focus (clNone = no focus is shown)
FocusFrameOn	: Switches the focus frame on and off
FocusFrameWidth	: The whidth of the focus-frame
Font	: The font to be used for text display in this button
ForegroundFocusOn	: Indicates when the button has focus, switches on off
GroupIndex	: Indicates when the button has focus, switches on off
Height	: The vertical size of the control.
HelpContext	: The ID for context-sensitive Help on this control
HelpKeyword	: The keyword for context-sensitive Help on this control

HelpType	: Determines whether context-sensitive Help is selected by numeric ID or keyword
Hint	: The text to show in the Hint window for the control
HoverBlendFaktor	: How translucent is the HoverColor (1=opaque,0=transparent), only at runtime!
HoverColor	: The color of a hoverevent (clNone = no hover)
ImgSizeFactor	: To compensate if images with <>64px are loaded with LoadFromFile
Left	: The client coordinate of the left edge of the control
LeftBgrdColor	: The Left background colour
LeftCaption	: The caption that is displayed when the button is on the left
LeftImageIndex	: The Index of the loaded left image
NewRollImage	: Starts the property editor to select loaded images
PopupMenu	: A context-sensitive menu that pops up when the right mouse button is clicked over this control
RightBgrdColor	: The Right background colour
RightCaption	: The caption that is displayed when the button is on the right
RightImageIndex	: The Index of the loaded right image
Roll	: Determines whether the RollButton (Image) rotates
Rotation	: The steps by rotation
Speed	: The speed at which the button moves
SwitchMode	: Der Modus mit dem der Schalter betätigt wird, klicken oder schieben
TabOrder	: Determines the sequence of control navigation when the user presses the Tabkey
TabStop	: Allows the user to navigate to this control, by pressing the Tabkey
Top	: The client coordinate of the top edge of the control
Visible	: Allows to show or hide the control, and all of its children
Width	: The horizontal size of the control.

Public procedures

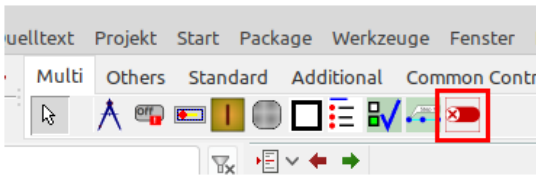
procedure LoadImagesfromFile(LeftFilename,RightFilename: string);

Events

OnChange	: Returns the Index of the checkbox
OnClick	: Notification handler for mouse clicks
<u>OnDirection</u>	: Returns the direction of the switch, aLeft and aRight return true or false.
OnDragDrop	: This handler determines the action on an drop onto this control, in a drag-drop operation
OnDragOver	: Event handler for a control being dragged over this control
OnEndDrag	: Notification handler for the end of a dragging operation
OnEnter	: Handler for control receiving the focus
OnExit	: Handler for control loosing the focus; This is a good place for checking the finished user input
OnKeyDown	: Handler for keyboard key pressed
OnKeyPress	: Handler for a character entered by the user
OnKeyUp	: Handler for keyboard key released
<u>OnLeft</u>	: Returns a true value if the button arrives on the left-hand side.
OnMouseDown	: Event handler for mouse button going down
OnMouseEnter	: Event handler for mouse entering the area of the control
OnMouseLeave	: Event handler for mouse leaving the area of the control
OnMouseMove	: Event handler for mouse movement within the control
OnMouseUp	: Event handler for mouse button going up
<u>OnRight</u>	: Returns a true value if the button arrives on the right-hand side.
OnStartDrag	: Event handler for the start of a dragging operation

Description

You can find the MultiSwitch in the Multi tab:



The idea for this switch originated in a contribution from the German Lazarus Forum:

<https://www.lazarusforum.de/viewtopic.php?p=137567#p137567>

If you place the MultiSwitch on the form, it looks like this:



It has a width of 60px and a height of 26px. This aspect ratio is always retained when scaling. All loaded images in the scroll button (here the grey cross) have a size of 64px. The maximum size is 175x76px. The images I loaded into the resource are from Roland Hahn (aka "Ally"). Many thanks to Roland!

<https://www.lazarusforum.de/viewtopic.php?f=1&t=14263>

It is important to know that the MultiSwitch is surrounded by a focus frame. As you can see here the focussed MultiSwitch has an orange frame. Conversely, this means that the actual switch is smaller around the frame.



The `FocusColor` property can be used to set the colour of the focus frame. `FocusAlphaBValue` can be used to control the transparency of the focus frame can be regulated. The value 0 means transparent and 255 opaque. `FocusFrameWidth` determines the thickness of the frame.

Value 50:



Value 200:



If you set `FocusFrameOn` to false, the frame is retained but the focussing is not displayed in colour.

With `ForegroundFocusOn` the focused MultiSwitch changes its colour. This is influenced with `FocusColor` and `FocusedBlendFactor` (between 0=transparent and 1=opaque).

not focussed:



focussed 0.2 Blendfaktor:



focussed 0.8 Blendfaktor:



The [Direction](#) property can be used to set whether the scroll button is initially on the left or right.

msLeft:



msRight:



[SwitchMode](#) toggles between clicking and sliding to switch the switch.

If you select msClick, a click on the switch is enough to change the status. If msSlide is set, you must drag the scroll button to the other side with the mouse. You can recognise that msSlide is active by a hand cursor.



If the switch has the focus, it can be switched with the enter button!

When changing the scroll button from one side to the other, the button starts to rotate around itself. If you do not want this to happen, you can set the [Roll](#) property to false.



With [Speed](#) you can set the rotation speed in ms. [Rotation](#) sets the angle by which the scroll button rotates per step.

[LeftCaption](#) sets the text that is displayed when the Rollbutton is on the left:



[RightCaption](#) sets the text that is displayed when the Rollbutton is on the right:



If the [BestTextHeight](#) property is true, an attempt is made to display the text as high as possible. If the text is too long, you can set BestTextHeight to false and set the font size under Font. The colour of the [Font](#) can also be changed there.



The [LeftBgrdColor](#) and [RightBgrdColor](#) properties can be used to change the background colour of the control. Left and Right always refer to the position of the Rollbutton.



If you change [BorderColor](#) to `<> clNone`, a border is drawn in the selected colour.



If you do not want to use a roll image, but rather a single-colour button, you must set [ButtonColor](#) to `<> clNone`. The button is then drawn in the selected colour.



To get a hover effect, set [HoverColor](#) to `<> clNone`. The transparency of the hover colour can be changed with [HoverBlendFaktor](#). Where 0 is transparent and 1 is opaque.

Without Hover



Blendfactor 0.2



Blendfactor 0.5



If you set [Enable](#) to false so that the switch no longer reacts to inputs, this is indicated by a different colour. The colour can be set with [DisabledColor](#) and the transparency with [EnabledBlendFaktor](#). Both properties are only available at run-time!

Enabled:

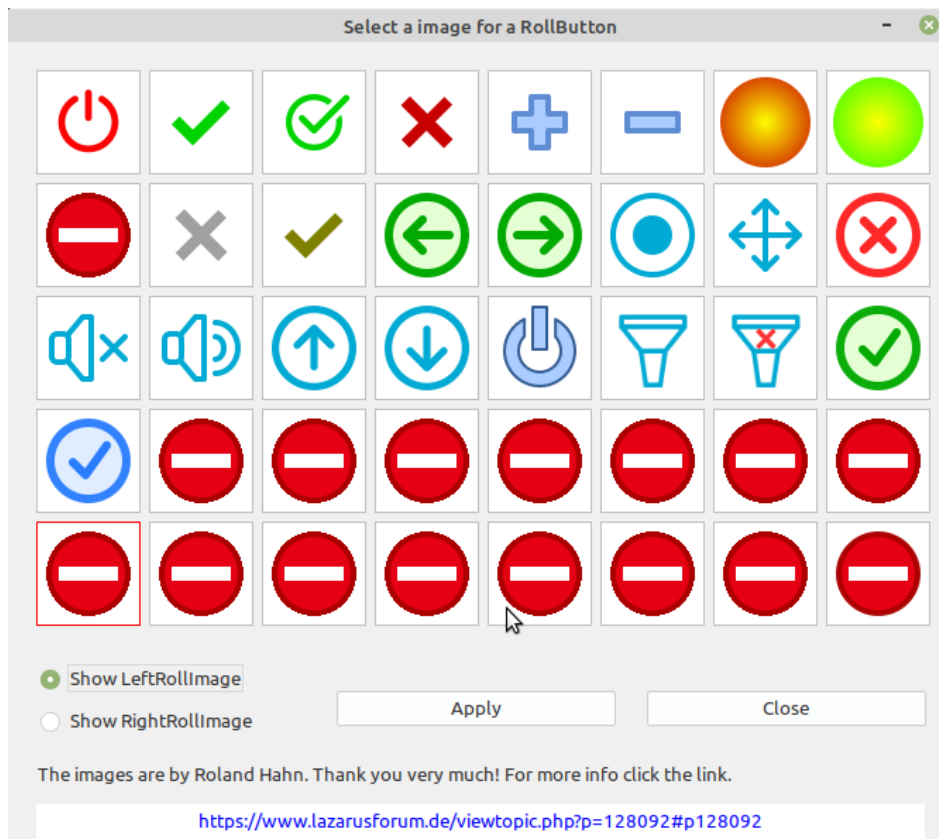


Disabled:



Roll images can be loaded using the property editor. To do this, click on [NewRollImage](#) in the object inspector.

The editor opens:



First select the side for which you want to select an image at the bottom left. Click on Apply to transfer the image to the MultiSwitch and display it immediately.

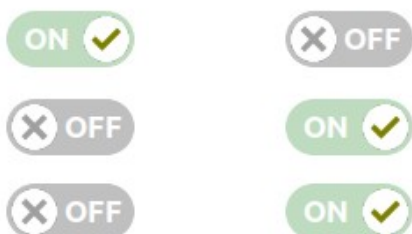
As you can see, there are still some free spaces to load your own images.

The resource file is located in the Multis directory.

Of course, you can also simply enter the relevant index of the desired image in the OI under [LeftImageIndex](#) or [RightImageIndex](#) (if you know it).

With [LoadImagesFromFile](#)(LeftFilename,RightFilename: string) images can be loaded from a directory. It is important that both images must have the same size! Preferably you should use images with 64px. The [ImgSizeFactor](#) property can be used to adjust the size of the loaded images (only at runtime).

If you want a group of MultiSwitches of which only one should be switched at a time, you must set the [GroupIndex](#) to $\langle \rangle 0$.



The MultiSwitch, which initially stands alone, determines the switching direction.

In addition to the usual events, there are also three special MultiSwitch events.

The **OnLeft** event returns a true value when the scroll button arrives on the left-hand side of the control.

The **OnRight** event returns a true value when the scroll button arrives on the right-hand side of the control.

The **OnDirection** event returns the direction (i.e. msRight or msLeft) and one Boolean value each for aLeft and aRight.