

KBasic Framework

KBasic uses Qt as its toolkit to provide cross-platform abilities. Qt is the cross-platform C++ library of <http://www.trolltech.com/>. KBasic is the easiest way to get cross-platform development without the needs to learn C++ as it combines the expressive power of C++ with the familiarity and ease of use of VB6. The Qt API and tools are consistent across all supported platforms, enabling platform independent application development and deployment. Windows, Linux and Mac OS X are supported platforms.



The original documentation of Qt can be read here: <http://doc.trolltech.com/>

The KBasic Framework is not just a wrapper around the Qt library, but simplifies the using of Qt, without adding much overhead to its functionality. It enables you to easily write modern cross-platform applications using BASIC syntax and commands.

This new framework provides many improvements and enhancements over the previous releases. This overview covers the most important features.

KBasic will be continuesly improved as a result of feedback and suggestions from customers and the open source community.

Table of Contents

Control.....	3
Form.....	24
CommandButton.....	36
Label.....	39
CheckBox.....	42
RadioButton.....	44
TextBox.....	46
Frame.....	50
ComboBox.....	50
ListBox.....	56
DateBox.....	63
TimeBox.....	66
DateTimeBox.....	68
Timer.....	71
Tab.....	73
Image.....	75
TreeView.....	76
Listview.....	89
Box.....	90
Editor.....	91
Browser.....	93
MenuBarItem.....	95
ToolBarItem.....	96
Report.....	97
Header.....	97
Footer.....	97
ChildControl.....	97
String.....	98
Event.....	99
MenuBar.....	100
ToolBar.....	101
Math.....	102
Application.....	103
Forms.....	105
Pixmaps.....	107
Colors.....	107
Fonts.....	107
Paint.....	108

Control

This is the parent class of all controls providing common functionality for all controls.

It receives mouse, keyboard and other events from the window system, and paints a representation of itself on the screen.

A form itself is a control, which is a window with controls.

It receives mouse, keyboard and other events from the window system, and paints a representation of itself on the screen.

A [Form](#) itself is a control, which is a window with controls.

This class contains functionality for both controls and forms. Some of it makes only sense when working with forms, others when working with non-form controls.

Most important

Methods [Close](#) , [SetFocus](#) , [Open](#)

Properties [Background](#) , [X](#) , [Y](#) , [Width](#) , [Height](#) , [Focus](#) , [ControlType](#) , [FontName](#) , [FontSize](#) , [Enabled](#) , [Visible](#) , [FontColor](#)

Events [OnOpen](#) , [OnClose](#) , [OnLostFocus](#) , [OnGotFocus](#) , [OnHide](#) , [OnShow](#) , [OnTimer](#)

Attention for name collision

Be sure that you do not use variable names, which are the same as property names in the KBasic Framework, because properties are used before user defined variables by the compiler.

Example:

```
Dim X As Integer ' you ought to rename it to nX meaning Dim nX As Integer

Sub mySub()
    X = 10 ' this does not affect the variable declared above,
           'but assigns to the property X of the control

    ' nX = 10
End Sub
```

OVERVIEW

Close a form [Close](#) → [OnClose](#) , Open a form [Open](#) → [OnOpen](#) , [OpenHidden](#) → [OnOpen](#) ,

Hide a control/form [Hide](#) → [OnHide](#) , [Visible](#) = False → [OnHide](#) , Show a control/form [Show](#) → [OnShow](#) , [Visible](#) = True → [OnShow](#) ,

Set position of a control/form [Move](#) → [OnMove](#) , Change size of a control/form [Resize](#) → [OnResize](#) ,

[RepaintAlways](#) , [Repaint](#) , [OnPaint](#) ,

[Name](#) , [ControlType](#) , [Group](#) , [ParentForm](#) , [ParentControl](#) ,

[X](#) , [Y](#) , [Width](#) , [Height](#) , [GlobalX](#) , [GlobalY](#) , [OldX](#) , [OldY](#) , [OldWidth](#) , [OldHeight](#) , [LoadedX](#) , [LoadedY](#) ,

[LoadedWidth](#), [LoadedHeight](#), [Move](#), [Resize](#), [OnMove](#), [OnResize](#),
[Layout](#), [MinimumWidth](#), [MinimumHeight](#), [MaximumWidth](#), [MaximumHeight](#),
[Tag](#), [CSV](#), [SQL](#),
[SetFocus](#), [Focus](#), [FocusPolicy](#), [FocusProxy](#), [FocusOrder](#), [OnGotFocus](#), [OnLostFocus](#),
[FontName](#), [FontSize](#), [FontItalic](#), [FontBold](#), [FontUnderline](#), [FontColor](#),
[Enabled](#), [Visible](#),
[Background](#), [FontColor](#),
[StatusTip](#), [ToolTip](#), [WhatsThis](#),
[OnEnter](#), [OnExit](#),
[OnKeyDown](#), [OnKeyUp](#), [OnKeyPress](#),
[OnDbClick](#), [OnClick](#),
[OnMouseMove](#), [OnMouseDown](#), [OnMouseUp](#),
[TimerInterval](#), [OnTimer](#),
[Lower](#), [Raise](#),

METHODS

Close

Function Close() As Boolean

Closes this control. Returns true if the control was closed; otherwise returns false.

First it sends the control a [OnClose](#). The control is hidden if it does not cancel the close event. If it cancel the event, nothing happens.

Close events are delivered to the control no matter if the control is visible or not.

Only useable if control is a form control.

Hide

Sub Hide()

Hides the control. This function is equivalent to Visible = False. Results in [OnHide](#).

See also [Show](#)

Lower

Sub Lower()

Lowers the control to the bottom of the parent control's visible stack.

After this call the control will be visually behind any overlapping control.

Normally, you do not need this sub.

See also [Raise](#)

Raise

Sub Raise()

Raises this control to the top of the parent control's visible stack.

After this call the control will be visually in front of any overlapping control.

Normally, you do not need this sub.

See also [Lower](#)

RepaintAlways

Sub RepaintAlways()

Repaints the control directly by calling [OnPaint](#) immediately, unless the control is hidden.

If you need an immediate repaint, it is better to use [RepaintAlways](#), for example during animation. In almost all circumstances [Repaint](#) is better, as it permits to optimize for minimize flicker and speed.

Warning: If you call [RepaintAlways](#) in a function which may itself be called from [OnPaint](#), you may get infinite recursion. The [Repaint](#) function never causes recursion.

See also [Repaint](#)

Repaint

Sub Repaint()

This function does not cause an immediate repaint; instead it schedules a paint event for processing when KBasic returns to the main event loop. This permits KBasic to optimize for more speed and less flicker than a call to [RepaintAlways](#) does.

Calling it several times normally results in just one [OnPaint](#) call.

See also [RepaintAlways](#)

Show

Sub Show()

Shows the control and its child controls. This function is equivalent to Visible = True. Results in [OnShow](#).

See also [Hide](#)

SetFocus

Sub SetFocus()

Gives the keyboard input focus to this control (or its [FocusProxy](#)) if this control or one of its parents is the active form.

First, a [OnLostFocus](#) event is sent to the focus control to tell it that it is about to lose the focus. Then a [OnGotFocus](#) event is sent to the other control to tell it that it just received the focus.

[OnSetFocus](#) gives focus to a control regardless of its [FocusPolicy](#).

Be aware that if the control is hidden, it will not accept focus.

If the control is a form, it sets the form to be the active window.

An active window is a visible top-level window that has the keyboard input focus.

The Qt documentation says that

"This function performs the same operation as clicking the mouse on the title bar of a top-level window. On X11, the result depends on the Window Manager. If you want to ensure that the window is stacked on top as well you should also call [Raise](#). Note that the window must be visible, otherwise FormSetActive() has no effect. On Windows, if you are calling this when the application is not currently the active one then it will not make it the active window. It will change the color of the taskbar entry to indicate that the window has changed in some way."

Open

Function Open() As Boolean

Opens this control. Only useable together with forms.

First it sends the control an [OnOpen](#). The control is shown.

Only useable if control is a form control.

OpenHidden

Function OpenHidden () As Boolean

Opens this control. Only useable together with forms.

First it sends the control an [OnOpen](#). The control is NOT shown.

Only useable if control is a form control.

Move

Sub Move(X As Integer, Y As Integer)

Moves this control.

Sends the control an [OnMove](#).

Resize

Sub Resize(Width As Integer, Height As Integer)

Resizes this control.

Sends the control an [OnResize](#).

PROPERTIES

Name

Property Name As String (ReadOnly)

The name of the control.

Layout

Property Layout As String (ReadOnly)

Group

Property Group As String (ReadOnly)

Contains the group to which the control belongs. Normally, you need only to set it, if you need RadioButtons to be in exclusive mode (only one at a time may be selected). Or, if you need to handle the same event at one place for many controls.

Background

Property Background As String (ReadWrite)

Background might be a color or an image. If you set a color use this format &RRGGBB (RGB value) e.g. &00FF00 (green). An image can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

Color objects of the class Colors must not be used yet.

Example

```
Dim c As CommandButton
c = Control("Button0") ' Button0 is declared in current form
c.Background = "tux.jpg" ' relative path. File exists in current project
directory
c.Background = "&00FF00" ' set the background to green
c.Background = "Red" ' set the background to red
```

Predefined color values

“White”, “Black”, “Red”, “DarkRed”, “Green”, “DarkGreen”, “Blue”, “DarkBlue”, “Cyan”, “DarkCyan”, “Magenta”, “DarkMagenta”, “Yellow”, “DarkYellow”, “Gray”, “DarkGray”, “LightGray”.

Color objects of the class Colors must not be used yet.

X

Property X As Integer (ReadWrite)

Left position on the screen of the control.

Y

Property Y As Integer (ReadWrite)

Top position on the screen of the control.

Width

Property Width As Integer (ReadWrite)

Height

Property Height As Integer (ReadWrite)

GlobalX

Property GlobalX As Integer (ReadOnly)

Contains the x position relative to the entire screen and the parent control or parent form.

GlobalY

Property GlobalY As Integer (ReadOnly)

Contains the y position relative to the entire screen and the parent control or parent form.

OldX

Property OldX As Integer (ReadOnly)

Contains the old x value of that control before the current x was set.

OldY

Property OldY As Integer (ReadOnly)

Contains the old y value of that control before the current y was set.

OldWidth

Property OldWidth As Integer (ReadOnly)

Contains the old width value of that control before the current width was set.

OldHeight

Property OldHeight As Integer (ReadOnly)

Contains the old height value of that control before the current height was set.

LoadedX

ReadOnly LoadedX As Integer

LoadedY

ReadOnly LoadedY As Integer

LoadedWidth

ReadOnly LoadedWidth As Integer

LoadedHeight

ReadOnly LoadedHeight As Integer

MinimumWidth

ReadWrite MinimumWidth As Integer

Useful for extended layout management.

MinimumHeight

ReadWrite MinimumHeight As Integer

Useful for extended layout management.

MaximumWidth

ReadWrite MaximumWidth As Integer

Useful for extended layout management.

MaximumHeight

ReadWrite MaximumHeight As Integer

Useful for extended layout management.

Tag

ReadWrite Tag As String

This is freely to use for custom property values.

CSV

ReadWrite CSV As String

Used by some controls like [ComboBox](#) to store comma separated values (CSV).

SQL

ReadWrite SQL As String

This is not the binding sql name for the control. If you would like to bind your control to a table see [SQLName](#)

Contains a sql select statement for custom sql data filling.

Used by some controls like [ComboBox](#) to store values generated by a sql query.

This makes only sense when used together with [ComboBox](#), [TextBox](#), [TreeView](#) or [ListBox](#). Other controls are not supported.

For [TextBox](#), the sql fills the completer property and for [TreeView](#), it provides the data to be displayed.

It is important that when you select two columns the first one is treated as data and the other column(s) are treated as caption. The data can be used in code for further event handling.

```
SELECT id, name, code FROM myTable
```

ParentForm

ReadOnly ParentForm As String

This is used by sub forms to get information about the parent form.

ParentControl

ReadOnly ParentControl As String

Returns the parent control. Normally, it is the current form, but it could be a tab control as well.

For forms it is always an empty string "".

(DragDrop)

ReadWrite DragDrop As Boolean

Not implemented yet.

(MouseTracking)

ReadOnly MouseTracking As Boolean

Not implemented yet.

ControlType

ReadOnly ControlType As String

Possible values are e.g.

- CommandButton
- TextBox
- Editor
- Browser
- CheckBox
- RadioButton
- Label
- Tab
- Box

and much more!

Focus

ReadOnly Focus As Boolean

If the control has got the focus now.

FocusPolicy

ReadWrite FocusPolicy As String

FocusProxy

ReadOnly FocusProxy As String

FocusOrder

ReadOnly FocusOrder As Integer

Start with number 1 for the first control.

If FocusOrder is set to 0 (default), it will not be in the focus chain, managing tabbing trough controls. It will gain focus by accident.

Cursor

ReadWrite Cursor As String

Set the cursor for that control. If the mouse pointer is over the control, it will change to one of the

following shapes.

Possible values are:

- ArrowCursor
- UpArrowCursor
- CrossCursor
- WaitCursor
- IBeamCursor
- SizeVerCursor
- SizeHorCursor
- SizeFDiagCursor
- SizeBDiagCursor
- SizeAllCursor
- BlankCursor
- SplitVCursor
- SplitHCursor
- PointingHandCursor
- ForbiddenCursor
- OpenHandCursor
- ClosedHandCursor
- WhatsThisCursor
- BusyCursor

Cursor might be an image. An image can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

Example

```
Cursor = "tux.jpg" ' relative path. File exists in current project directory
```

FontName

ReadWrite FontName As String

e.g. "Arial", "Courier"

FontSize

ReadWrite FontSize As Integer

FontItalic

ReadWrite FontItalic As Boolean

FontBold

ReadWrite FontBold As Boolean

FontUnderline

ReadWrite FontUnderline As Boolean

FontColor

ReadWrite FontColor As String

If you set a color use this format &RRGGBB (RGB value) e.g. &00FF00 (green).

Color objects of the class Colors must not be used yet.

Example

```
Dim c As CommandButton  
c = Control("Button0") ' Button0 is declared in current form  
c.FontColor = "&00FF00" ' set the background to green
```

Predefined color values

“White”, “Black”, “Red”, “DarkRed”, “Green”, “DarkGreen”, “Blue”, “DarkBlue”, “Cyan”, “DarkCyan”, “Magenta”, “DarkMagenta”, “Yellow”, “DarkYellow”, “Gray”, “DarkGray”, “LightGray”.

Color objects of the class Colors must not be used yet.

Enabled

ReadWrite Enabled As Boolean

Visible

ReadWrite Visible As Boolean

StatusTip

ReadWrite StatusTip As String

Text is shown in the statusbar, if it is selected or so. Behaviour depends on the control type.

ToolTip

ReadWrite ToolTip As String

Behaviour depends on the control type.

SytleSheet

ReadWrite SytleSheet As String

Example for TextBox (QLineEdit) setting the background yellow:

```
QLineEdit { background: yellow }
```

Fore more information read <http://doc.trolltech.com/4.3/stylesheet.html>

WhatsThis

ReadWrite WhatsThis As String

Not implemented yet.

TimerInterval

ReadOnly TimerInterval As Integer

Calls [OnTimer](#) after TimerInterval milli seconds.

Only useable if control is a form control.

TimerInterval = 1000 means 1 second

[OnTimer](#) is only called after the form was opened.

(DrawOnPaint)

ReadOnly DrawOnPaint As Boolean

Not implemented yet: If it is set to false, the controls won't draw the default look. This is useful, if you want to provide custom drawing without the default look.

Only useable if control is NOT a form control.

Opacity

ReadWrite Opacity As Integer

This only works on Mac OS X and Windows 2000 or later. It might work on Linux as well.

Makes only sense for form controls. Set the transparent level for the form control. Only forms with no parent are affected, e.g. forms inside a mainwindow, do not change the transparent level.

Values from 0 to 100 are allowed.

- 0 means completely transparent.
 - 100 means completely visible.
-

(ContextMenu)

ReadWrite ContextMenu As String

The name of the menubar to be displayed when context menu for this control is requested.

Not implemented yet.

HIDDEN PROPERTIES

(BoxIndex)

in future versions:

ReadOnly BoxIndex As Integer

Internally used by KBasic.

(SplitIndex)

in future versions:

ReadOnly SplitIndex As Integer

Internally used by KBasic.

TabIndex

ReadOnly TabIndex As Integer

Internally used by KBasic to set up tabs.

(BoxX)

in future versions:

ReadOnly BoxX As Integer

Internally used by KBasic.

(BoxY)

in future versions:

ReadOnly BoxY As Integer

Internally used by KBasic.

ParentIndex

ReadOnly ParentIndex As Integer

Internally used by KBasic.

EVENTS

[Close](#) → [OnClose](#) , [Open](#) → [OnOpen](#) , [OpenHidden](#) → [OnOpen](#) ,

[Hide](#) → [OnHide](#) , [Visible](#) = False → [OnHide](#) , [Show](#) → [OnShow](#) , [Visible](#) = True → [OnShow](#) ,

[Move](#) → [OnMove](#) , [Resize](#) → [OnResize](#) ,

[RepaintAlways](#) → [OnPaint](#) , [Repaint](#) → [OnPaint](#) ,

[X](#) → [OnMove](#) , [Y](#) → [OnMove](#) , [Width](#) → [OnResize](#) , [Height](#) → [OnResize](#) ,

[SetFocus](#) → [OnLostFocus/OnGotFocus](#)

Mouse cursor comes over control → [OnEnter](#) , Mouse cursor was over control, but is now outside of it → [OnExit](#) ,

Key down first time → [OnKeyDown](#) , Key was down, but now released → [OnKeyUp](#) , Key was first time pressed and now continued pressed → [OnKeyPress](#) ,

Mouse double clicked on control → [OnDbClick](#) , Mouse double or only one time clicked on control → [OnClick](#) ,

Mouse cursor moves over control → [OnMouseMove](#) , Mouse button down first time → [OnMouseDown](#) , Mouse button was down, but now released → [OnMouseUp](#) ,

Every milli seconds of [TimerInterval](#) → [OnTimer](#) ,

OnOpen

OnOpen()

Only used by form controls.

OnClose

OnClose(ByRef Cancel As Boolean)

Only used by form controls. If you reimplement this sub and set Cancel = True, the form won't close.

(OnContextMenu)

OnContextMenu(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer)

Not implemented yet.

(OnDragEnter)

OnDragEnter(ByRef Cancel As Boolean, X As Integer, Y As Integer, Width As Integer, Height As Integer)

Not implemented yet.

(OnDragExit)

OnDragExit()

Not implemented yet.

(OnDragMove)

OnDragMove(ByRef Cancel As Boolean, X As Integer, Y As Integer, Width As Integer, Height As Integer)

Not implemented yet.

(OnDrop)

OnDrop(ByRef Cancel As Boolean, MimeData As String)

Not implemented yet.

OnEnter

OnEnter()

OnGotFocus

OnGotFocus()

Form objects do not receive this event.

OnLostFocus

OnLostFocus()

Form objects do not receive this event.

OnHide

OnHide()

OnKeyDown

OnKeyDown(KeyCode As Integer, Shift As Boolean, Control As Boolean, Alt As Boolean)

See [Key Codes](#) for possible key code values.

OnKeyUp

OnKeyUp(KeyCode As Integer, Shift As Boolean, Control As Boolean, Alt As Boolean)

See [Key Codes](#) for possible key code values.

OnKeyPress

OnKeyPress(KeyCode As Integer, Shift As Boolean, Control As Boolean, Alt As Boolean)

See [Key Codes](#) for possible key code values.

OnExit

OnExit()

OnDbClick

OnDbClick(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer, LeftButton As Boolean, RightButton As Boolean, MidButton As Boolean)

Note that the controls get a `OnMouseDown(...)` and an `OnMouseUp(...)` before the `OnDbClick(...)`.

OnClick

`OnClick(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer, LeftButton As Boolean, RightButton As Boolean, MidButton As Boolean)`

OnMouseMove

`OnMouseMove(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer, LeftButton As Boolean, RightButton As Boolean, MidButton As Boolean)`

Because of performance reasons, this event only works with Box and Form controls yet.

OnMouseDown

`OnMouseDown(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer, LeftButton As Boolean, RightButton As Boolean, MidButton As Boolean)`

OnMouseUp

`OnMouseUp(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer, LeftButton As Boolean, RightButton As Boolean, MidButton As Boolean)`

OnMove

`OnMove(X As Integer, Y As Integer, OldX As Integer, OldY As Integer)`

OnPaint

`OnPaint(X As Integer, Y As Integer, Width As Integer, Height As Integer)`

Currently, you may only use OnPaint with Box or Form objects.

Use the the static Paint object to do your custom drawing.

OnResize

`OnResize(Width As Integer, Height As Integer, OldWidth As Integer, OldHeight As Integer)`

OnShow

OnShow()

(OnPrint)

OnPrint()

It is called when your control is about to be printed in your report.

Not implemented yet.

OnTimer

OnTimer()

OnMouseWheel

OnMouseWheel(X As Integer, Y As Integer, GlobalX As Integer, GlobalY As Integer)

SQL PROPERTIES

SQLName

ReadOnly SQLName As String

Some controls may use this property only: [Label](#), [ListBox](#), [ComboBox](#), [TextBox](#), [DateBox](#), [TimeBox](#)

SQLRelation

ReadOnly SQLRelation As String

This makes only sense when used together with [ComboBox](#) or [ListBox](#). Other controls are not supported.

TableViewCaption

ReadOnly TableViewCaption As String

TableViewWidth

ReadOnly TableViewWidth As Integer

TableViewVisible

ReadOnly TableViewVisible As Boolean

Key Codes

On Mac OS X, Key.Control corresponds to the Command keys and Key.Meta corresponds to the Control keys.

Predefined key codes are:

Key.Escape , Key.Tab , Key.Backtab , Key.Backspace , Key.Return , Key.Enter , Key.Insert , Key.Delete , Key.Pause , Key.Print , Key.SysReq , Key.Clear , Key.Home , Key.End , Key.Left , Key.Up , Key.Right , Key.Down , Key.PageUp , Key.PageDown , Key.Shift , Key.Control , Key.Meta , Key.Alt , Key.AltGr , Key.CapsLock , Key.NumLock , Key.ScrollLock , Key.F1 , Key.F2 , Key.F3 , Key.F4 , Key.F5 , Key.F6 , Key.F7 , Key.F8 , Key.F9 , Key.F10 , Key.F11 , Key.F12 , Key.F13 , Key.F14 , Key.F5 , Key.F16 , Key.F17 , Key.F18 , Key.F19 , Key.F20 , Key.F21 , Key.F22 , Key.F23 , Key.F24 , Key.F25 , Key.F26 , Key.F27 , Key.F28 , Key.F29 , Key.F30 , Key.F31 , Key.F32 , Key.F33 , Key.F34 , Key.F35 , Key.Super_L , Key.Super_R , Key.Menu , Key.Hyper_L , Key.Hyper_R , Key.Help , Key.Direction_L , Key.Direction_R , Key.Space , Key.Any , Key.Exclam , Key.QuoteDbl , Key.NumberSign , Key.Dollar , Key.Percent , Key.Ampersand , Key.Apostrophe , Key.ParenLeft , Key.ParenRight , Key.Asterisk , Key.Plus , Key.Comma , Key.Minus , Key.Period , Key.Slash , Key.0 , Key.1 , Key.2 , Key.3 , Key.4 , Key.5 , Key.6 , Key.7 , Key.8 , Key.9 , Key.Colon , Key.Semicolon , Key.Less , Key.Equal , Key.Greater , Key.Question , Key.At , Key.A , Key.B , Key.C , Key. , Key.E , Key.F , Key.G , Key.H , Key.I , Key.J , Key.K , Key.L , Key.M , Key.N , Key.O , Key.P , Key.Q , Key.R , Key.S , Key.T , Key.U , Key.V , Key.W , Key.X , Key.Y , Key.Z , Key.BracketLeft , Key.Backslash , Key.BracketRight , Key.AsciiCircum , Key.Underscore , Key.QuoteLeft , Key.BraceLeft , Key.Bar , Key.BraceRight , Key.AsciiTilde , Key.nobreakspace , Key.exclamdown , Key.cent , Key.sterling , Key.currency , Key.yen , Key.brokenbar , Key.section , Key.diaeresis , Key.copyright , Key.ordfeminine , Key.guillemotleft , Key.notsign , Key.hyphen , Key.registered , Key.macron , Key.degree , Key.plusminus , Key.twosuperior , Key.threesuperior , Key.acute , Key.mu , Key.paragraph , Key.periodcentered , Key.cedilla , Key.onesuperior , Key.masculine , Key.guillemotright , Key.onequarter , Key.onehalf , Key.threequarters , Key.questiondown , Key.Agrave , Key.Aacute , Key.Acircumflex , Key.Atilde , Key.Adiaeresis , Key.Aring , Key.AE , Key.Ccedilla , Key.Egrave , Key.Eacute , Key.Ecircumflex , Key.Ediaeresis , Key.Igrave , Key.Iacute , Key.Icircumflex , Key.Idiaeresis , Key.ETH , Key.Ntilde , Key.Ograve , Key.Oacute , Key.Ocircumflex , Key.Otilde , Key.Odiaeresis , Key.multiply , Key.Ooblique , Key.Ugrave , Key.Uacute , Key.Ucircumflex , Key.Udiaeresis , Key.Yacute , Key.THORN , Key.ssharp , Key.division , Key.ydiaeresis , Key.Multi_key , Key.Codeinput , Key.SingleCandidate , Key.MultipleCandidate , Key.PreviousCandidate , Key.Mode_switch , Key.Kanji , Key.Muhenkan ,

Key.Henkan , Key.Romaji , Key.Hiragana , Key.Katakana , Key.Hiragana_Katakana , Key.Zenkaku , Key.Hankaku , Key.Zenkaku_Hankaku , Key.Touroku , Key.Massyo , Key.Kana_Lock , Key.Kana_Shift , Key.Eisu_Shift , Key.Eisu_toggle , Key.Hangul , Key.Hangul_Start , Key.Hangul_End , Key.Hangul_Hanja , Key.Hangul_Jamo , Key.Hangul_Romaja , Key.Hangul_Jeonja , Key.Hangul_Banja , Key.Hangul_PreHanja , Key.Hangul_PostHanja , Key.Hangul_Special , Key.Dead_Grave , Key.Dead_Acute , Key.Dead_Circumflex , Key.Dead_Tilde , Key.Dead_Macron , Key.Dead_Breve , Key.Dead_Abovedot , Key.Dead_Diaeresis , Key.Dead_Abovering , Key.Dead_Doubleacute , Key.Dead_Caron , Key.Dead_Cedilla , Key.Dead_Ogonek , Key.Dead_Iota , Key.Dead_Voiced_Sound , Key.Dead_Semivoiced_Sound , Key.Dead_Belowdot , , Key.Dead_Hook , Key.Dead_Horn , Key.Back , Key.Forward , Key.Stop , Key.Refresh , Key.VolumeDown , Key.VolumeMute , Key.VolumeUp , Key.BassBoost , Key.BassUp , Key.BassDown , Key.TrebleUp , Key.TrebleDown , Key.MediaPlay , Key.MediaStopKey.MediaPrevious , Key.MediaNext , Key.MediaRecord , Key.HomePage , Key.Favorites , Key.Search , Key.Standby , Key.OpenUrl , Key.LaunchMail , Key.LaunchMedia , Key.Launch0 , Key.Launch1 , Key.Launch2 , Key.Launch3 , Key.Launch4 , Key.Launch5 , Key.Launch6 , Key.Launch7 , Key.Launch8 , Key.Launch9 , Key.LaunchA , Key.LaunchB , Key.LaunchC , Key.LaunchD , Key.LaunchE , Key.LaunchF , Key.MediaLast , Key.unknown , Key.Call , Key.Context1 , Key.Context2 , Key.Context3 , Key.Context4 , Key.Flip , Key.Hangup , Key.No , Key.Select , Key.Yes , Key.Execute , Key.Printer , Key.Play , Key.Sleep , Key.Zoom , Key.Cancel ,

Form

A form itself is a control, which is a window with controls.

This is the parent class of all forms providing common functionality for all forms. It receives mouse, keyboard and other events from the window system, and paints a representation of itself on the screen.

In your KBasic-application forms are not only masks for inputting and changing data, but they are the graphical interface of your application. In the eyes of the beholder, they are the application! By creating your application using forms, you control the program flow with events, which are raised in the forms.

Central meaning of forms: Each form in a KBasic application has got a form module with event procedures. Those event procedures react on events raised in the form. Additionally, every module can contain other non-event procedures.

A form module is part of every form, when you copy a form, its form module is automatically copied, too. If you delete a form, its form module is deleted as well. KBasic creates a form module automatically. So you need only to write the event procedures and other procedures.

Forms keep your KBasic program together!

The control class contains functionality for both controls and forms. Some of it makes only sense when working with forms, others when working with non-form controls.

Override the event methods of [Control](#), when you would like to implement your own controls for display data or interact with the user. Normally, you override `OnPaint(...)` at least.

Please read the control class overview [Control](#) as well.

Important methods:

[Close](#) , [Hide](#) , [Show](#) , [SetFocus](#) , [Open](#) , [OpenHidden](#) , [Move](#) , [Resize](#)

Important events:

- [OnOpen](#)
- [OnClose](#)
- [OnTimer](#)

Be sure that you implement some events like [OnOpen](#) and [OnClose](#).

```
Sub Form_OnOpen()  
    ' some initial custom code  
End Sub  
  
Sub Form_OnClose(ByRef Cancel As Boolean)  
    ' some custom code, to close files are other resource, or to check  
    ' if closing of form is currently possible or allowed  
  
    Cancel = True ' if you would like to abort the closing  
End Sub  
  
' opening a from  
Dim f As bernd ' assume that bernd is a form class created with the form  
designer  
f = New bernd
```



```
f.Open()  
  
' OR  
  
Forms.Open("bernd") ' for this call you must setup mainwindow in projects  
properties
```

Hidden SQL Operating If you need to change sql records without notice for the user, create a form with controls and set the SQLName's of the controls. Open this form using OpenHidden and use it like you would use a visible form with sql controls and bindings.

OVERVIEW

[Control](#), [ControlFocus](#), [ControlFirst](#), [ControlNext](#),
[ShowFullScreen](#), [ShowMaximized](#), [ShowMinimized](#), [ShowNormal](#),
[Load](#) → [OnLoad](#),
[BorderTop](#), [BorderBottom](#), [BorderLeft](#), [BorderRight](#),
[Flag](#), [Modal](#), [Dock](#), [OpenOnRun](#), [StopOnClose](#)
[Icon](#), [Caption](#), [IconCaption](#)
[First](#), [Next](#), [Previous](#), [Last](#), [GoTo](#), [Seek](#), [Len](#), [Pos](#), [OnGoTo](#)
[Insert](#), [Update](#), [Delete](#), [AddNew](#), [GoTo](#), [Requery](#), [SQLInsert](#), [SQLUpdate](#), [SQLDelete](#), [OnQuery](#),
[OnAddNew](#), [OnBeforeInsert](#), [OnBeforeUpdate](#), [OnBeforeDelete](#)
[Run](#), [Get](#)
[ClearFilter](#), [AddFilter](#), [SetFilter](#)
[SortAscending](#), [SortDescending](#)
[IsDirty](#), [OnDirty](#)

METHODS

Control

Function Control(ByRef Name As String) As [Control](#)

Returns the control object for a given control name.

Argument(s)

Name As String the control, which should be returned

Return Value control object

Example

```
Dim c As CommandButton  
c = Control("Button1") ' Button1 is declared as CommandButton in this form  
Print c.Caption
```

See also [ControlFirst](#), [ControlNext](#)

ControlFocus

Function ControlFocus() As String

ShowFullScreen

Sub ShowFullScreen()

Shows the form in full-screen mode.

To return from full-screen mode, call [ShowNormal](#).

The Qt documentation says that

“Full-screen mode works fine under Windows, but has certain problems under X. Depending on the window manager, this may or may not work.”

ShowMaximized

Sub ShowMaximized()

Shows the form maximized.

ShowMinimized

Sub ShowMinimized()

Shows the form minimized.

ShowNormal

Sub ShowNormal()

Restores size of the form after it has been maximized or minimized.

ControlFirst

Function ControlFirst() As String

Returns the name of first control of the form. If there is no control, it returns "".

See also [ControlNext](#)

ControlNext

Function ControlNext() As String

Returns the name of next control of the form. If there is no control, it returns "".

Example

```
Dim n As String
n = ControlFirst()
If n <> "" Then
    Do
        Dim c As Control
        c = Control(n)
        ' place your code here

        n = ControlNext()
    Loop While n <> ""
End If
```

See also [ControlFirst](#)

Load

Function Load(FormName As String) As Boolean

Loads all controls of the form from the form file and makes them ready. Do NOT call this method explicitly. It is automatically by KBasic.

PROPERTIES

TableView

ReadOnly TableView As Boolean

BorderTop

ReadOnly BorderTop As Integer

BorderBottom

ReadOnly BorderBottom As Integer

BorderLeft

ReadOnly BorderLeft As Integer

BorderRight

ReadOnly BorderRight As Integer

Flag

ReadOnly Flag As String

- For a tool window set: Tool + SystemMenu and Modal = True
 - For a dialog window set: Dialog + SystemMenu and Modal = True
-

Icon

ReadWrite Icon As String

Sets the icon of the form.

An icon can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

See also [Caption](#)

IconCaption

ReadWrite IconCaption As String

Normally, setting Caption is only useful.

Caption

ReadWrite Caption As String

Sets the window title of the form.

Value

Property Value As String (ReadWrite)

Same as [Caption](#). Provided for easy-use.

OldValue

Property OldValue As String (ReadOnly)

Modal

ReadOnly Modal As Boolean

StopOnClose

ReadOnly StopOnClose As Boolean

Stops the application, if the event OnClose returns true.

OpenOnRun

ReadOnly OpenOnRun As Boolean

Opens automatically this form on application start up.

Only useable if control is a form control.

Dock

ReadOnly Dock As String

May only be used, if there is a mainwindow (see properties of project. Additionally, myMenuBar menubar class must be created).

EVENTS

[ShowFullScreen](#) → [OnShow](#), [ShowMaximized](#) → [OnShow](#), [ShowMinimized](#) → [OnShow](#),
[ShowNormal](#) → [OnShow](#),

[Load](#) → [OnLoad](#),

[First](#) → [OnGoTo](#), [Next](#) → [OnGoTo](#), [Previous](#) → [OnGoTo](#), [Last](#) → [OnGoTo](#), [GoTo](#) → [OnGoTo](#),
[Seek](#) → [OnGoTo](#)

[Insert](#) → [OnBeforeInsert](#), [Update](#) → [OnBeforeUpdate](#), [Delete](#) → [OnBeforeDelete](#), [AddNew](#) →
[OnAddNew](#), [GoTo](#) → [OnGoTo](#)

Query first time loaded → [OnQuery](#), [Requery](#) → [OnQuery](#)

Record data changed by user → [OnDirty](#)

OnLoad

OnLoad()

It is called after a form has been created by New FORMNAME.

SQL METHODS

First

Function First() As Boolean

Next

Function Next() As Boolean

Previous

Function Previous() As Boolean

Last

Function Last() As Boolean

GoTo

Function GoTo(Position As Long) As Boolean

Insert

Function Insert() As Boolean

Update

Function Update() As Boolean

Delete

Function Delete() As Boolean

AddNew

Function AddNew() As Boolean

```
AddNew() ' create new empty record
SetRecord("name", "Bernd") ' set custom values
' Record!name = "Bernd"
Insert() ' actually insert his new record in database
```

Len

Function Len() As Long

Pos

Function Pos() As Long

IsDirty

Function IsDirty() As Boolean

Run

Function Run(SQLStatement As String) As Boolean

Get

Function Get(SearchFor As String, InTableDotField As String, ReturnFieldName As String) As AnyType

Example

```
Dim r As String = Get("99", "mytable.id", "name")
```

If the field is of sql database table type 'Text' you must always not forget to setup the the SearchFor correctly. e.g. 'mytext', 'bernd', without ' it will fail.

If InTableDotField is "*", the last found record with this function is used to provide the return type.

Seek

Function Seek(Filter As String = "", [Filter As String = ""]) As Boolean

Upto six filters are allowed. Move the current position until the record with matching entries could be found.

Example

```
Seek() ' select the control you wish to seek for, before  
Seek("id = 12", "name <> 'test'", "age > 12")
```

Warning

This function might be slow on recordsets with many thousands entries.

Requery

Sub Requery()

ClearFilter

Sub ClearFilter()

AddFilter

Sub AddFilter(String = "")

Unlike [SetFilter](#) AddFilter does NOT removes the previous custom filter set by the user. Removing previous filter set, must be done by using [ClearFilter](#).

If you you do not provide an argument, the filter will take effect on the last visited control with the operator "=".

Possible operators:

- "="
- "<>"
- ">"
- "<"

```
AddFilter() ' filters for the last visited control by the user with operator "="  
AddFilter("=") ' filters for the last visited control by the user  
AddFilter("<>") ' filters for the last visited control by the user  
AddFilter("name = 'Bernd'")
```

SetFilter

Sub SetFilter(String = "")

Unlike [AddFilter](#) SetFilter removes the previous custom filter set by the user.

If you do not provide an argument, the filter will take effect on the last visited control with the operator "=".

Possible operators:

- "="
- "<>"
- ">"
- "<"

```
SetFilter() ' filters for the last visited control by the user with operator "="
SetFilter("=") ' filters for the last visited control by the user
SetFilter("<>") ' filters for the last visited control by the user
SetFilter("name = 'Bernd'")
```

SortAscending

Sub SortAscending(String = "")

```
SortAscending("name")
```

SortDescending

Sub SortDescending(String = "")

```
SortDesc("name")
```

SQL PROPERTIES

SQLName

ReadWrite SQLName As String

SQLControls

ReadOnly SQLControls As String

If set empty, it will show all SQL controls (default), otherwise only determined controls are visible at runtime.

SQLInsert

If set 'False', it is not possible for the user to add new records to the database in this form. The 'Add New' button is not visible at all.

ReadWrite SQLInsert As Boolean

SQLUpdate

If set 'False', it is not possible for the user to change existing records to the database in this form.

ReadWrite SQLUpdate As Boolean

SQLDelete

If set 'False', it is not possible for the user to delete existing records in the database in this form. The 'Delete' button is not visible at all.

ReadWrite SQLDelete As Boolean

SQL EVENTS

OnGoTo

OnGoTo()

This event is raised after the current record position changed.

OnQuery

OnQuery()

This event is raised, before the SQL query of the form is executed. May be launched by setting a filter or sorting, by clicking on the 'Refresh' button...

OnAddNew

OnAddNew()

This event is raised, after the user clicked on the 'Add New' button.

OnDirty

OnDirty()

This event is raised, after the user changed some value(s) in the current control binding to the current record. The changed values of the controls need to be updated in the record of the database table by clicking on the 'Update' button, because so the current record is marked as dirty and the user is asked to save those changes into database, when trying to move to another record.

OnBeforeInsert

OnBeforeInsert(ByRef Cancel As Boolean)

OnBeforeUpdate

OnBeforeUpdate(ByRef Cancel As Boolean)

OnBeforeDelete

OnBeforeDelete(ByRef Cancel As Boolean)

CommandButton

A control for the form object, provides a push button.

The command button, or push button, is perhaps the most commonly used control in any graphical user interface. Push (click) a button to command the computer to perform some action, or to answer a question. Typical buttons are OK, Apply, Cancel, Close, Yes, No and Help.

A command button is rectangular and typically displays a text label describing its action. An underlined character in the label (signified by preceding it with an ampersand in the text) indicates an accelerator key. Command buttons can display a textual label or an icon.

Most important

Methods None

Properties [Icon](#) , [Caption](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

PROPERTIES

Default

Property Default As Boolean (ReadOnly)

This property holds whether the CommandButton is the default button.

If it is to true then the CommandButton will be pressed if the user presses the Enter (or Return) key in a dialog.

Regardless of focus, if the user presses Enter: If there is a default button the default button is pressed; otherwise, if there are one or more autoDefault buttons.

The default button behavior is provided only in dialogs. Buttons can always be clicked from the keyboard by pressing Spacebar when the button has focus.

Flat

Property Flat As Boolean (ReadOnly)

If it is true, the CommandButton appears as flat button. This is only useful to change the visual appearance.

Icon

Property Icon As String (ReadWrite)

Either an icon or caption is visible on a CommandButton. If you set an icon, the caption will not be displayed. An icon can be an absolute path to an image file (png, jpg,...) like

c:\myfolder\myimage.png or can be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

Example

```
Dim c As CommandButton  
c = Control("Button1") ' Button1 is declared as CommandButton in this form  
c.Icon= "tux.jpg" ' relative path. File exists in current project directory
```

See also [Caption](#)

Key

Property Key As String (ReadOnly)

Holds the information, which key press would raise the event connected to this CommandButton, e.g. Ctrl+O. The strings “Ctrl”, “Shift”, “Alt” and “Meta” are recognized.

Example

```
Shift+L  
Alt+U  
Alt+Shift+U  
Ctrl+Alt+C
```

Caption

Property Caption As String (ReadWrite)

Holds the text to be displayed on the CommandButton.

Example

```
Dim c As CommandButton  
c = FormControl("Button1") ' Button1 is declared as CommandButton in this form  
c.Caption = "Hello World!"
```

See also [Icon](#)

Value

Property Value As String (ReadWrite)

Same as [Caption](#). Provided for easy-use.

OldValue

Property OldValue As String (ReadOnly)

EVENTS

OnEvent

Sub OnEvent()

This event is raised, whenever the user presses the CommandButton, either by mouse or key.

Example

```
Sub Button1_OnEvent()  
    Print "Button1 pressed"  
End Sub
```

Label

A control for the form object, provides a text or image display.

It is used for displaying text or an image. No user interaction functionality is provided. You can display HTML text with it too.

Most important

Methods None

Properties [Icon](#) , [Caption](#)

Events None

Please read the control class overview [Control](#) as well.

PROPERTIES

Shape

Property Shape As Boolean (ReadWrite)

If it is true, a shape appears behind the label.

OpenLinks

Property OpenLinks As Boolean (ReadWrite)

If it is true, the label opens the default browser, when [Caption](#) contains link in HTML code, which was clicked by the user. The label will look like a link only at runtime. In the form designer it will look like any ordinary label.

Set the following properties:

```
Caption = www.kbasic.com  
OpenLinks = True  
Feature = LinksAccessibleByMouse;
```

WordWrap

Property WordWrap As Boolean (ReadWrite)

If it is true, Label shows its Caption in several lines.

Scale

Property Scale As Boolean (ReadWrite)

If it is true, the image of the label is shown as big as the geometry of label is. The text of the label is

not affected.

Alignment

Property Alignment As String (ReadWrite)

How [Caption](#) should be displayed? Should it be on top inside the boundaries of the control for example?

Feature

Property Feature As String (ReadOnly)

How [Caption](#) should be displayed?

InputControl

Property InputControl As String (ReadOnly)

When the user presses the shortcut key indicated by this label, the keyboard focus is transferred to the Control defined by InputControl.

This mechanism is only available for Labels that contain plain text in which one letter is prefixed with an ampersand, &. This letter is set as the shortcut key. The letter is displayed underlined, and the '&' is not displayed, but only at runtime. At runtime you must hit ALT+Letter, e.g. ALT+F if you have "&Find" as label text.

Icon

Property Icon As String (ReadWrite)

Either an icon or caption is visible. If you set an icon, the caption will not be displayed. An icon can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

Example

```
Dim c As Label
c = Control("Label0") ' Label0 is declared as CommandButton in this form
c.Icon = "tux.jpg" ' relative path. File exists in current project directory
```

See also [Caption](#)

Caption

Property Caption As String (ReadWrite)

Holds the text to be displayed.

Example

```
Dim c As Label0  
c = Control("Label0") ' Label0 is declared as CommandButton in this form  
c.Caption = "Hello World!"
```

See also [Icon](#)

Value

Property Value As String (ReadWrite)

Same as [Caption](#). Provided for easy-use.

OldValue

Property OldValue As String (ReadOnly)

CheckBox

A control for the form object, provides a checkbox with a text label.

It can be switched on (checked) or off (unchecked).

If you would like to have a group of CheckBoxes, use the [Group](#) property and set all CheckBoxes to the same group.

Most important

Methods None

Properties [Value](#) , [Icon](#) , [Caption](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

PROPERTIES

Value

Property Value As Boolean (ReadWrite)

OldValue

Property OldValue As Boolean (ReadOnly)

Key

Property Key As String (ReadOnly)

Holds the information, which key press would raise the event connected to this Control, e.g. Ctrl+O. The strings "Ctrl", "Shift", "Alt" and "Meta" are recognized.

Example

```
Shift+L  
Alt+U  
Alt+Shift+U  
Ctrl+Alt+C
```

Icon

Property Icon As String (ReadWrite)

An icon can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can

be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

Example

```
Dim c As CheckBox
c = Control("CheckBox0") ' CheckBox0 is declared in this form
c.Icon = "tux.jpg" ' relative path. File exists in current project directory
```

See also [Caption](#)

Caption

Property Caption As String (ReadWrite)

Holds the text to be displayed on the Control.

Example

```
Dim c As CheckBox
c = Control("CheckBox0") ' CheckBox0 is declared in this form
c.Caption = "Hello World!"
```

See also [Icon](#)

EVENTS

OnEvent

Sub OnEvent()

This event is raised, whenever the user presses the Control, either by mouse or key.

Example

```
Sub CheckBox0_OnEvent()
    Print "CheckBox0 pressed"
End Sub
```

RadioButton

A control for the form object, provides a radio button with a text label.

It can be switched on (checked) or off (unchecked).

If you would like to have a group of RadioButtons, use the [Group](#) property and set all RadioButtons to the same group. **Exclusive RadioButtons are declared by setting the property Group to the same value on all radio buttons you like to group.**

Most important

Methods None

Properties [Value](#) , [Icon](#) , [Caption](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

PROPERTIES

Value

Property Value As Boolean (ReadWrite)

OldValue

Property OldValue As Boolean (ReadOnly)

Key

Property Key As String (ReadOnly)

Holds the information, which key press would raise the event connected to this Control, e.g. Ctrl+O. The strings "Ctrl", "Shift", "Alt" and "Meta" are recognized.

Example

```
Shift+L  
Alt+U  
Alt+Shift+U  
Ctrl+Alt+C
```

Icon

Property Icon As String (ReadWrite)

An icon can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can

be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

Example

```
Dim c As RadioButton  
c = Control("RadioButton0") ' RadioButton0 is declared in this form  
c.Icon = "tux.jpg" ' relative path. File exists in current project directory
```

See also [Caption](#)

Caption

Property Caption As String (ReadWrite)

Holds the text to be displayed on the Control.

Example

```
Dim c As RadioButton  
c = Control("RadioButton0") ' RadioButton0 is declared in this form  
c.Caption = "Hello World!"
```

See also [Icon](#)

EVENTS

OnEvent

Sub OnEvent()

This event is raised, whenever the user presses the Control, either by mouse or key.

Example

```
Sub RadioButton0_OnEvent()  
    Print "RadioButton0 pressed"  
End Sub
```

TextBox

A control for the form object, is a one-line text editor.

It allows the user to enter and edit a single line of plain text with a useful collection of editing functions, including undo and redo, cut and paste, and drag and drop.

Most important

Methods [IsValid](#)

Properties [Value](#) , [InputMask](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

METHODS

IsValid

Function IsValid () As Boolean

Returns true, if input mask and validation is valid for the current input text.

SelectAll

Sub SelectAll()

Sorry. Not implemented yet.

PROPERTIES

Alignment

Property Alignment As String (ReadOnly)

How [Value](#) should be displayed. Should it be on top inside the boundaries of the control for example?

Value

Property Value As String (ReadWrite)

OldValue

Property OldValue As String (ReadOnly)

InputMask

Property InputMask As String (ReadOnly)

Holds the validation input mask.

Possible characters may be (Qt original documentation)

- A ASCII alphabetic character required. A-Z, a-z.
- a ASCII alphabetic character permitted but not required.
- N ASCII alphanumeric character required. A-Z, a-z, 0-9.
- n ASCII alphanumeric character permitted but not required.
- X Any character required.
- x Any character permitted but not required.
- 9 ASCII digit required. 0-9.
- 0 ASCII digit permitted but not required.
- D ASCII digit required. 1-9.
- d ASCII digit permitted but not required (1-9).
- # ASCII digit or plus/minus sign permitted but not required.
- H Hexadecimal character required. A-F, a-f, 0-9.
- h Hexadecimal character permitted but not required.
- B Binary character required. 0-1.
- b Binary character permitted but not required.
- > All following alphabetic characters are uppercased.
- < All following alphabetic characters are lowercased.
- ! Switch off case conversion.
- \ Use \ to escape the special characters listed above to use them as separators

The mask consists of a string of mask characters and separators, optionally followed by a semicolon and the character used for blanks: the blank characters are always removed from the text after editing. The default blank character is space.

Example

```
000.000.000.000;_      IP address; blanks are _.  
HH:HH:HH:HH:HH:HH;_    MAC address  
0000-00-00             ISO Date; blanks are space  
>AAAAA-AAAAA-AAAAA-AAAAA-AAAAA;#      License number; blanks are - and all  
(alphabetic) characters are converted to uppercase.
```

ReadOnly

Property ReadOnly As Boolean (ReadWrite)

If it is true, the textbox value cannot be changed by the user.

ValidatorDouble

Property ValidatorDouble As Boolean (ReadOnly)

If it is true, the textbox value maybe a double value.

ValidatorInteger

Property ValidatorInteger As Boolean (ReadOnly)

If it is true, the textbox value maybe an integer value.

Maximum

Property Maximum As Integer (ReadOnly)

Defines how many characters may be entered.

Completer

Property Completer As String (ReadWrite)

Contains a list of words to be used to auto-complete text input. Words may be separated by ; or ,

EVENTS

OnEvent

Sub OnEvent()

This event is raised, whenever the user finished editing, which means that Return or Enter key is pressed or the TextBox loses focus. Note that if there is a validator or [InputMask](#) set and enter/return is pressed, the Event() will only be raised if the input follows the [InputMask](#) and one of the validators.

Example

```
Sub TextBox0_OnEvent()  
    Print "TextBox0 pressed"  
End Sub
```


Frame

A control for the form object, provides a group box frame with a title.

Most important

Methods None

Properties [Caption](#)

Events None

Please read the control class overview [Control](#) as well.

PROPERTIES

Caption

Property Caption As String (ReadWrite)

Holds the text to be displayed.

Example

```
Dim c As Frame
c = Control("Frame0") ' Frame0 is declared as Frame in this form
c.Caption = "Hello World!"
```

Value

Property Value As String (ReadWrite)

Same as [Caption](#). Provided for easy-use.

OldValue

Property OldValue As String (ReadOnly)

ComboBox

A control for the form object, is a combined button and popup list.

A combobox is a selection control which displays the current item and can pop up a list of items. Comboboxes provide a means of showing the user's current choice out of a list of options in a way that takes up the minimum amount of screen space.

Whenever you use Index, KBasic checks if Index is greater equal 1 and smaller equal Len(). If not, the command will not be executed, e.g. You use Insert(10, "test"), but there are only 4 elements yet, the Insert will fail. First element has got Index = 1.

Most important

Methods [Find](#)

Properties [Value](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

METHODS

Len

Function Len() As Integer

Find

Function Find(Caption As String) As Integer

Searches all items and return the index.

The first entry has got index = 1.

Returns 0 if item could not be found.

The search is case-sensitive means that it must exactly match the search string.

HidePopUp

Sub HidePopUp()

ShowPopUp

Sub ShowPopUp()

Insert

Sub Insert(Index As Integer, Caption As String)

The first entry has got index = 1.

Insert

Sub Insert(Index As Integer, Icon As String, Caption As String)

The first entry has got index = 1.

Append

Sub Append(Caption As String)

Append

Sub Append(Icon As String, Caption As String)

Remove

Sub Remove(Index As Integer)

The first entry has got index = 1.

RemoveAll

Sub RemoveAll()

Select

Sub Select(Index As Integer)

The first entry has got index = 1.

Select

Sub Select(Caption As String)

The search is case-sensitive means that it must exactly match the search string.

Selected

Function Selected() As Integer

The first entry has got index = 1.

Caption

Function Caption() As String

Index

Function Index() As Integer

Data

Function Data() As Long

Caption

Function Caption(Index As Integer) As String

SetCaption

Sub SetCaption(Index As Integer, Caption As String)

SetIcon

Sub SetIcon(Index As Integer, Icon As String)

Tag

Function Tag(Index As Integer) As String

SetTag

Sub SetTag(Index As Integer, Tag As String)

PROPERTIES

Value

Property Value As String (ReadWrite)

Set the current selected item.

The search is case-sensitive means that it must exactly match the search string.

OldValue

Property OldValue As String (ReadOnly)

Contains the previous selected item.

DoubleEntry

Property DoubleEntry As Boolean (ReadOnly)

If it is true, it is possible to have entries with the same caption many times in the combobox list.

InsertPolicy

Property InsertPolicy As String (ReadOnly)

Possible values are

- NoInsert
 - InsertAtTop
 - InsertAtCurrent
 - InsertAtBottom
 - InsertAfterCurrent
 - InsertBeforeCurrent
 - InsertAlphabetically
-

Editable

Property Editable As Boolean (ReadOnly)

If it is true, it is possible for the user to select or enter by keyboard.

Maximum

Property Maximum As Integer (ReadOnly)

How many entries are allowed.

MaximumVisible

Property MaximumVisible As Integer (ReadOnly)

How many items are visible at one time, means how long is the list you can see in the popup.

Flat

Property Flat As Boolean (ReadOnly)

If it is true, appears as flat control. This is only useful to change the visual appearance.

EVENTS

OnEvent

Sub OnEvent(Index As Integer, Caption As String)

This event is raised, when the user selects a new item.

The first item has got index = 1.

ListBox

A control for the form object, provides a list of selectable, read-only items.

This is typically a single-column list in which either no item or one item is selected, but it can also be used in many other ways.

Whenever you use Index, KBasic checks if Index is greater equal 1 and smaller equal Len(). If not, the command will not be executed, e.g. You use Insert(10, "test"), but there are only 4 elements yet, the Insert will fail. First element has got Index = 1.

Most important

Properties [Value](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

METHODS

Len

Function Len() As Integer

Sort

Sub Sort(Descending As Boolean)

Insert

Sub Insert(Index As Integer, Caption As String)

Insert

Sub Insert(Index As Integer, Icon As String, Caption As String)

Append

Sub Append(Caption As String)

Append

Sub Append(Icon As String, Caption As String)

Remove

Sub Remove(Index As Integer)

Index = 1 means the first entry

Index = 2 means the second entry and so on

RemoveAll

Sub RemoveAll()

Select

Sub Select(Index As Integer)

Select

Sub Select(Caption As String)

Searches case-sensitive.

Selected

Function Selected() As Integer

Caption

Function Caption() As String

Index

Function Index() As Integer

Data

Function Data() As Long

Caption

Function Caption(Index As Integer) As Integer

SetCaption

Sub SetCaption(Index As Integer, Caption As String)

SetIcon

Sub SetIcon(Index As Integer, Icon As String)

Tag

Function Tag(Index As Integer) As Integer

SetTag

Sub SetTag(Index As Integer, Tag As String)

CheckState

Function CheckState(Index As Integer) As String

SetCheckState

Sub SetCheckState(Index As Integer, CheckState As String)

Possible Value	Description
Unchecked	The item is unchecked.
PartiallyChecked	The item is partially checked. Items in hierarchical models may be partially checked if some, but not all, of their children are checked.
Checked	The item is checked.

Flag

Function Flag(Index As Integer) As String

SetFlag

Sub SetFlag(Index As Integer, Flag As String)

SetBackground

Sub SetBackground(Index As Integer, ColorId As String)

SetBackground

Sub SetBackground(Index As Integer, ColorId As String, BrushStyle As String)

IsHidden

Function IsHidden(Index As Integer) As Boolean

SetHidden

Sub SetHidden(Index As Integer, IsHidden As Boolean)

IsSelected

Function IsSelected(Index As Integer) As Boolean

SetSelected

Sub SetSelected(Index As Integer, IsSelected As Boolean)

StatusTip

Function StatusTip(Index As Integer) As String

SetStatusTip

Sub SetStatusTip(Index As Integer, StatusTip As String)

ToolTip

Function ToolTip(Index As Integer) As String

SetToolTip

Sub SetToolTip(Index As Integer, ToolTip As String)

TextAlignment

Function TextAlignment(Index As Integer) As String

SetTextAlignment

Sub SetTextAlignment(Index As Integer, TextAlignment As String)

Possible Value	Description (Qt documentation says)
AlignLeft	Aligns with the left edge.
AlignRight	Aligns with the right edge.
AlignHCenter	Centers horizontally in the available space.
AlignJustify	Justifies the text in the available space.
AlignTop	Aligns with the top.
AlignBottom	Aligns with the bottom.
AlignVCenter	Centers vertically in the available space.
AlignCenter	Centers in both dimensions.
AlignAbsolute	If the widget's layout direction is RightToLeft (instead of LeftToRight, the default), AlignLeft refers to the right edge and AlignRight to the left edge. This is normally the desired behavior. If you want AlignLeft to always mean "left" and AlignRight to always mean "right", combine the flag with AlignAbsolute.
AlignLeading	Synonym for AlignLeft.
AlignTrailing	Synonym for AlignRight.
AlignHorizontal_Mask	AlignLeft Or AlignRight Or AlignHCenter Or AlignJustify Or AlignAbsolute
AlignVertical_Mask	AlignTop Or AlignBottom Or AlignVCenter

PROPERTIES

Value

Property Value As String (ReadWrite)

OldValue

Property OldValue As String (ReadOnly)

Sorted

Property Sorted As Boolean (ReadOnly)

This property holds whether sorting is enabled.

SelectionMode

Property SelectionMode As String (ReadOnly)

AlternatingRowColors

Property AlternatingRowColors As Boolean (ReadOnly)

Flat

Property Flat As Boolean (ReadOnly)

If it is true, appears as flat control. This is only useful to change the visual appearance.

EVENTS

OnEvent

Sub OnEvent(Index As Integer, Caption As String)

This event is raised, when the user changes the selection.

DateBox

A control for the form object for editing dates.

It allows the user to edit dates by using the keyboard or the arrow keys to increase and decrease date values. Dates appear in accordance with the format set.

Most important

Methods [None](#)

Properties [Value](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

PROPERTIES

Value

Property Value As String (ReadWrite)

Format is yyyy-MM-dd, e.g. 2007-12-31

OldValue

Property OldValue As String (ReadOnly)

Maximum

Property Maximum As String (ReadWrite)

Format is yyyy-MM-dd, e.g. 2007-12-31

Minimum

Property Minimum As String (ReadWrite)

Format is yyyy-MM-dd, e.g. 2007-12-31

Format

Property Format As String (ReadWrite)

Change the format of the date box for the user. It does not affect the format of [Maximum](#) or [Minimum](#) or [Value](#).

Date related

d	the day as number without a leading zero (1 to 31)
dd	the day as number with a leading zero (01 to 31)
ddd	the abbreviated localized day name (e.g. 'Mon' to 'Sun').
dddd	the long localized day name.
M	the month as number without a leading zero (1-12)
MM	the month as number with a leading zero (01-12)
MMM	the abbreviated localized month name (e.g. 'Jan' to 'Dec').
MMMM	the long localized month name (e.g. 'January' to 'December').
yy	the year as two digit number (00-99)
yyyy	the year as four digit number

Time related

h	the hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)
hh	the hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)
m	the minute without a leading zero (0 to 59)
mm	the minute with a leading zero (00 to 59)
s	the second without a leading zero (0 to 59)
ss	the second with a leading zero (00 to 59)
z	the milliseconds without leading zeroes (0 to 999)
zzz	the milliseconds with leading zeroes (000 to 999)
AP	use AM/PM display. AP will be replaced by either "AM" or "PM".
ap	use am/pm display. ap will be replaced by either "am" or "pm".

PopUp

Property PopUp As Boolean (ReadOnly)

Flat

Property Flat As Boolean (ReadOnly)

If it is true, appears as flat control. This is only useful to change the visual appearance.

EVENTS

OnEvent

Sub OnEvent()

TimeBox

A control for the form object for editing times.

It allows the user to edit times by using the keyboard or the arrow keys to increase and decrease time values. Times appear in accordance with the format set.

Most important

Methods [None](#)

Properties [Value](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

PROPERTIES

Value

Property Value As String (ReadWrite)

Format is hh:mm:ss, e.g. 12:12:12 or 23:10:12 (means 11:10 12 seconds PM)

Maximum

Property Maximum As String (ReadWrite)

Format is hh:mm:ss, e.g. 12:12:12 or 23:10:12 (means 11:10 12 seconds PM)

Minimum

Property Minimum As String (ReadWrite)

Format is hh:mm:ss, e.g. 12:12:12 or 23:10:12 (means 11:10 12 seconds PM)

Format

Property Format As String (ReadWrite)

Change the format of the time box for the user. It does not affect the format of [Maximum](#) or [Minimum](#) or [Value](#).

Date related

d the day as number without a leading zero (1 to 31)

dd the day as number with a leading zero (01 to 31)

ddd the abbreviated localized day name (e.g. 'Mon' to 'Sun').
dddd the long localized day name.
M the month as number without a leading zero (1-12)
MM the month as number with a leading zero (01-12)
MMM the abbreviated localized month name (e.g. 'Jan' to 'Dec').
MMMM the long localized month name (e.g. 'January' to 'December').
yy the year as two digit number (00-99)
yyyy the year as four digit number

Time related

h the hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)
hh the hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)
m the minute without a leading zero (0 to 59)
mm the minute with a leading zero (00 to 59)
s the second without a leading zero (0 to 59)
ss the second with a leading zero (00 to 59)
z the milliseconds without leading zeroes (0 to 999)
zzz the milliseconds with leading zeroes (000 to 999)
AP use AM/PM display. AP will be replaced by either "AM" or "PM".
ap use am/pm display. ap will be replaced by either "am" or "pm".

PopUp

Property PopUp As Boolean (ReadOnly)

Flat

Property Flat As Boolean (ReadOnly)

If it is true, appears as flat control. This is only useful to change the visual appearance.

EVENTS

OnEvent

Sub OnEvent()

DateTimeBox

A control for the form object for editing dates and times.

It allows the user to edit dates and times by using the keyboard or the arrow keys to increase and decrease date and time values. Dates and times appear in accordance with the format set.

Most important

Methods [None](#)

Properties [Value](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

PROPERTIES

Value

Property Value As String (ReadWrite)

Format is yyyy-MM-dd hh:mm:ss, e.g. 2007-12-31 23:10:12

OldValue

Property OldValue As String (ReadOnly)

Maximum

Property Maximum As String (ReadWrite)

Format is yyyy-MM-dd hh:mm:ss, e.g. 2007-12-31 23:10:12

Minimum

Property Minimum As String (ReadWrite)

Format is yyyy-MM-dd hh:mm:ss, e.g. 2007-12-31 23:10:12

Format

Property Format As String (ReadWrite)

Change the format of the date box for the user. It does not affect the format of [Maximum](#) or

[Minimum](#) or [Value](#).

Date related

d	the day as number without a leading zero (1 to 31)
dd	the day as number with a leading zero (01 to 31)
ddd	the abbreviated localized day name (e.g. 'Mon' to 'Sun').
dddd	the long localized day name.
M	the month as number without a leading zero (1-12)
MM	the month as number with a leading zero (01-12)
MMM	the abbreviated localized month name (e.g. 'Jan' to 'Dec').
MMMM	the long localized month name (e.g. 'January' to 'December').
yy	the year as two digit number (00-99)
yyyy	the year as four digit number

Time related

h	the hour without a leading zero (0 to 23 or 1 to 12 if AM/PM display)
hh	the hour with a leading zero (00 to 23 or 01 to 12 if AM/PM display)
m	the minute without a leading zero (0 to 59)
mm	the minute with a leading zero (00 to 59)
s	the second without a leading zero (0 to 59)
ss	the second with a leading zero (00 to 59)
z	the milliseconds without leading zeroes (0 to 999)
zzz	the milliseconds with leading zeroes (000 to 999)
AP	use AM/PM display. AP will be replaced by either "AM" or "PM".
ap	use am/pm display. ap will be replaced by either "am" or "pm".

PopUp

Property PopUp As Boolean (ReadOnly)

Flat

Property Flat As Boolean (ReadOnly)

If it is true, appears as flat control. This is only useful to change the visual appearance.

EVENTS

OnEvent

Sub OnEvent()

Timer

Most important

Methods [IsRunning](#) , [Start](#) , [Stop](#)

Properties [Intervall](#) , [Enabled](#)

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

A form control may use as many timer controls as you desire. Be aware that the form control comes with a built-in timer. See [TimerInterval](#) for more details.

METHODS

IsRunning

Function IsRunning() As Boolean

Start

Sub Start()

Stop

Sub Stop()

PROPERTIES

Interval

Property Interval As Integer (ReadWrite)

Enabled

ReadWrite Enabled As Boolean

If you it is set True, the timer event will start automatically after opening the form. Do not forget to set [Interval](#) of the timer.

If you would like to manually start the timer set Enabled=False and use [Start](#) and [Stop](#) to manage the timer. Do not forget to set [Interval](#) of the timer.

EVENTS

OnEvent

Sub OnEvent()

This event is raised, whenever the timer interval is reached.

Tab

Whenever you use Index, KBasic checks if Index is greater equal 1 and smaller equal Len(). If not, the command will not be executed. First element has got Index = 1.

Most important

Methods [Select](#) , [Selected](#)

Properties None

Events [OnEvent](#)

Please read the control class overview [Control](#) as well.

METHODS

Select

Sub Select(Index As Integer)

First item is index = 0.

Selected

Function Selected() As Integer

First item is index = 0.

SetTabEnabled

Sub SetTabEnabled(Index As Integer, Enable As Boolean)

IsTabEnabled

Function IsTabEnabled(Index As Integer) As Boolean

SetToolTip

Sub SetToolTip(Index As Integer, ToolTip As String)

ToolTip

Function ToolTip(Index As Integer) As String

SetCaption

Sub SetCaption(Index As Integer, Caption As String)

Caption

Function Caption(Index As Integer) As String

SetIcon

Sub SetIcon(Index As Integer, Icon As String)

HIDDEN PROPERTIES

Pages

ReadOnly Pages As String

Normally, you do not use this property. It is used by KBasic to organize your tab control.

EVENTS

OnEvent

Sub OnEvent(Index As Integer)

This event is raised, when the current tab index is changed.

Image

A control for the form object, provides a image representation.

Most important

Methods None

Properties [Value](#)

Events None

Please read the control class overview [Control](#) as well.

PROPERTIES

Value

Property Value As String (ReadWrite)

Contains the path of the image to be displayed. It can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

Might be an identifier of the pixmap array (Pxmmaps).

OldValue

Property OldValue As String (ReadOnly)

TreeView

A control providing information in a tree structure. It might be used as list view as well, because you can create lists with several columns, but the coding is the same.

Please read the control class overview [Control](#) as well.

Whenever you use Index, KBasic checks if Index is greater equal 1 and smaller equal Len(). If not, the command will not be executed, e.g. You use Insert(10, "test"), but there are only 4 elements yet, the Insert will fail. First element has got Index = 1.

METHODS

AppendChild

Function AppendChild(Caption As String) As Long

AppendChild

Function AppendChild(Icon As String, Caption As String) As Long

AppendChild

Function AppendChild(Id As Long, Caption As String) As Long

AppendChild

Function AppendChild(Id As Long, Icon As String, Caption As String) As Long

ChildCount

Function ChildCount(Id As Long) As Integer

Child

Function Child(Id As Long, Index As Integer) As Long

Tag

Function Tag(Id As Long, Column As Integer) As String

SetTag

Sub SetTag(Id As Long, Column As Integer, Tag As String)

CheckState

Function CheckState(Id As Long, Column As Integer) As String

SetCheckState

Sub SetCheckState(Id As Long, Column As Integer, CheckState As String)

Data

Function Data(Id As Long) As Long

Sorry. Not implemented yet.

Flag

Function Flag(Id As Long) As String

SetFlag

Sub SetFlag(Id As Long, Flag As String)

Flag	Qt documentation says
ItemIsSelectable	It can be selected.
ItemIsEditable	It can be edited.
ItemIsDragEnabled	It can be dragged.
ItemIsDropEnabled	It can be used as a drop target.
ItemIsUserCheckable	It can be checked or unchecked by the user.
ItemIsEnabled	The user can interact with the item.
ItemIsTristate	The item is checkable with three separate states.

SetBackground

Sub SetBackground(Id As Long, Column As Integer, ColorId As String)

SetBackground

Sub SetBackground(Id As Long, Column As Integer, ColorId As String, BrushStyle As String)

SetFontColor

Sub SetFontColor(Id As Long, Column As Integer, ColorId As String)

SetFontColor

Sub SetFontColor(Id As Long, Column As Integer, ColorId As String, BrushStyle As String)

SetFont

Sub SetFont(Id As Long, Column As Integer, FontId As String)

Caption

Function Caption(Id As Long, Column As Integer) As String

SetCaption

Sub SetCaption(Id As Long, Column As Integer, Caption As String)

SetIcon

Sub SetIcon(Id As Long, Column As Integer, Icon As String)

IndexOfChild

Function IndexOfChild(Id As Long, ChildId As Long) As Integer

InsertChild

Function InsertChild(Id As Long, Index As Integer, Caption As String) As Long

InsertChild

Function InsertChild(Id As Long, Index As Integer, Icon As String, Caption As String) As Long

IsExpanded

Function IsExpanded(Id As Long) As Boolean

SetExpanded

Sub SetExpanded(Id As Long, IsExpanded As Boolean)

IsHidden

Function IsHidden(Id As Long) As Boolean

SetHidden

Sub SetHidden(Id As Long, IsHidden As Boolean)

IsSelected

Function IsSelected(Id As Long) As Boolean

SetSelected

Sub SetSelected(Id As Long, IsSelected As Boolean)

Please read [SetFlag](#) as well.

StatusTip

Function StatusTip(Id As Long, Column As Integer) As String

SetStatusTip

Sub SetStatusTip(Id As Long, Column As Integer, StatusTip As String)

ToolTip

Function ToolTip(Id As Long, Column As Integer) As String

SetToolTip

Sub SetToolTip(Id As Long, Column As Integer, ToolTip As String)

RemoveChild

Sub RemoveChild(Id As Long, Index As Integer)

TextAlignment

Function TextAlignment(Id As Long, Column As Integer) As String

SetTextAlignment

Sub SetTextAlignment(Id As Long, Column As Integer, TextAlignment As String)

Possible Value	Description (Qt documentation says)
AlignLeft	Aligns with the left edge.
AlignRight	Aligns with the right edge.
AlignHCenter	Centers horizontally in the available space.
AlignJustify	Justifies the text in the available space.
AlignTop	Aligns with the top.
AlignBottom	Aligns with the bottom.
AlignVCenter	Centers vertically in the available space.
AlignCenter	Centers in both dimensions.
AlignAbsolute	If the widget's layout direction is RightToLeft (instead of LeftToRight, the default), AlignLeft refers to the right edge and AlignRight to the left edge. This is normally the desired behavior. If you want AlignLeft to always mean "left" and AlignRight to always mean "right", combine the flag with AlignAbsolute.
AlignLeading	Synonym for AlignLeft.

AlignTrailing	Synonym for AlignRight.
AlignHorizontal_Mask	AlignLeft Or AlignRight Or AlignHCenter Or AlignJustify Or AlignAbsolute
AlignVertical_Mask	AlignTop Or AlignBottom Or AlignVCenter

ColumnCount

Function ColumnCount(Id As Long) As Integer

Append

Function Append(Caption As String) As Long

Append

Function Append(Icon As String, Caption As String) As Long

ClosePersistentEditor

Sub ClosePersistentEditor(Id As Long, Column As Integer)

Normally, you would call this sub.

ColumnCount

Function ColumnCount() As Integer

CurrentColumn

Function CurrentColumn() As Integer

CurrentItem

Function CurrentItem() As Long

FindItem

Function FindItem(Caption As String, Flag As String, Column As Integer) As Long

Flag	Qt documentation says
MatchExactly	Performs QVariant-based matching.
MatchFixedString	Performs string-based matching. String-based comparisons are case-insensitive unless the MatchCaseSensitive flag is also specified.
MatchContains	The search term is contained in the item.
MatchStartsWith	The search term matches the start of the item.
MatchEndsWith	The search term matches the end of the item.
MatchCaseSensitive	The search is case sensitive.
MatchRegExp	Performs string-based matching using a regular expression as the search term.
MatchWildcard	Performs string-based matching using a string with wildcards as the search term.
MatchWrap	Perform a search that wraps around, so that when the search reaches the last item in the model, it begins again at the first item and continues until all items have been examined.
MatchRecursive	Searches the entire hierarchy.

HeaderItem

Function HeaderItem() As Long

IndexOfTopLevelItem

Function IndexOfTopLevelItem(Id As Long) As Integer

Insert

Function Insert(Index As Integer, Caption As String) As Long

Insert

Function Insert(Index As Integer, Icon As String, Caption As String) As Long

InvisibleRootItem

Function InvisibleRootItem() As Long

ItemAt

Function ItemAt(X As Integer, Y As Integer) As Long

Returns the item at position X/Y.

OpenPersistentEditor

Sub OpenPersistentEditor(Id As Long, Column As Integer)

Normally, you would call this sub.

FirstSelectedItem

Function FirstSelectedItem() As Long

NextSelectedItem

Function NextSelectedItem() As Long

SetColumnCount

Sub SetColumnCount(Count As Integer)

SetCurrentItem

Sub SetCurrentItem(Id As Long)

SetCurrentItem

Sub SetCurrentItem(Id As Long, Column As Integer)

SetHeaderLabel

Sub SetHeaderLabel(Caption As String)

SetHeaderLabel

Sub SetHeaderLabel(Caption As String, Column As Integer)

SortColumn

Function SortColumn() As Integer

SortItems

Sub SortItems(Column As Integer, Descending As Boolean)

RemoveTopLevelItem

Sub RemoveTopLevelItem(Index As Integer)

TopLevelItem

Function TopLevelItem(Index As Integer) As Long

TopLevelItemCount

Function TopLevelItemCount() As Integer

IsColumnHidden

Function IsColumnHidden(Column As Integer) As Boolean

SetColumnHidden

Sub SetColumnHidden(Column As Integer, IsColumnHidden As Boolean)

IsSortingEnabled

Function IsSortingEnabled() As Boolean

SetSortingEnabled

Sub SetSortingEnabled(IsSortingEnabled As Boolean)

IsItemsExpandable

Function IsItemsExpandable() As Boolean

SetItemsExpandable

Sub SetItemsExpandable(IsItemsExpandable As Boolean)

IsAllColumnsShowFocus

Function IsAllColumnsShowFocus() As Boolean

SetAllColumnsShowFocus

Sub SetAllColumnsShowFocus(IsAllColumnsShowFocus As Boolean)

ColumnWidth

Function ColumnWidth(Column As Integer) As Integer

SetColumnWidth

Sub SetColumnWidth(Column As Integer, Width As Integer)

CollapseAll

Sub CollapseAll()

ExpandAll

Sub ExpandAll()

SelectAll

Sub SelectAll()

ShowColumn

Sub ShowColumn(Column As Integer)

CollapseItem

Sub CollapseItem(Id As Long)

ExpandItem

Sub ExpandItem(Id As Long)

ScrollToItem

Sub ScrollToItem(Id As Long)

IsAlternatingRowColors

Funciton IsAlternatingRowColors() As Boolean

SetAlternatingRowColors

Sub SetAlternatingRowColors(IsAlternatingRowColors As Boolean)

ScrollToBottom

Sub ScrollToBottom()

ScrollToTop

Sub ScrollToTop()

PROPERTIES

SelectionMode

Property SelectionMode As String (ReadOnly)

AlternatingRowColors

Property AlternatingRowColors As Boolean (ReadOnly)

Flat

Property Flat As Boolean (ReadOnly)

If it is true, appears as flat control. This is only useful to change the visual appearance.

EVENTS

OnEvent

Sub OnEvent()

This event is raised, whenever the user changes the selection.

OnCurrentItemChanged

Sub OnCurrentItemChanged(IdCurrent As Long, IdPrevious As Long)

OnItemActivated

Sub OnItemActivated(Id As Long, Column As Integer)

OnItemChanged

Sub OnItemChanged(Id As Long, Column As Integer)

OnItemClicked

Sub OnItemClicked(Id As Long, Column As Integer)

OnItemCollapsed

Sub OnItemCollapsed(Id As Long)

OnItemDoubleClicked

Sub OnItemDoubleClicked(Id As Long, Column As Integer)

OnItemEntered

Sub OnItemEntered(Id As Long, Column As Integer)

OnItemExpanded

Sub OnItemExpanded(Id As Long)

OnItemPressed

Sub OnItemPressed(Id As Long, Column As Integer)

OnItemSelectionChanged

Sub OnItemSelectionChanged()

ListView

A control providing information in a list structure. It is actually a TreeView.

Example

Your form contains of TreeView0(control type TreeView).

```
Sub Form_OnOpen()  
  
    TreeView0.SetHeaderLabel("name", 0)  
    TreeView0.SetHeaderLabel("age", 1)  
    TreeView0.SetHeaderLabel("city", 2)  
    TreeView0.SetHeaderLabel("cat", 3)  
  
    Dim id As Long  
    Dim i As Integer  
  
    TreeView0.SetColumnWidth(0, 200)  
    TreeView0.SetColumnWidth(1, 100)  
    TreeView0.SetColumnWidth(2, 100)  
    TreeView0.SetColumnWidth(3, 100)  
  
    Do  
        i = i + 1  
        If i > 10 Then Goto r  
        id = TreeView0.AppendChild("")  
  
        TreeView0.SetCaption(id, 0, "bernd" & i)  
        TreeView0.SetCaption(id, 1, "28" & i)  
        TreeView0.SetCaption(id, 2, "Frankfurt" & i)  
        TreeView0.SetCaption(id, 3, "Balthasar" & i)  
        TreeView0.SetIcon(id, 0, "button_ok.png")  
    Loop While( True )  
  
    r:  
    TreeView0.SetColumnWidth(0, 200)  
    TreeView0.SetColumnWidth(1, 100)  
    TreeView0.SetColumnWidth(2, 100)  
    TreeView0.SetColumnWidth(3, 100)  
  
End Sub
```

Box

A control for the form object, which provides a with custom color filled rectangle. Useful for separating controls.

You might want to use it to override the event methods of [Control](#), when you would like to implement your own controls for display data or interact with the user. Normally, you override OnPaint(...) at least.

Example

Your form contains of Box0 (control type Box).

```
Sub Box0_OnPaint(X As Integer, Y As Integer, Width As Integer, Height As Integer)
    DrawRect(11, 22, 33, 44)
End Sub
```

Editor

A control for the form object, provides a powerful single-page rich text editor.

It is an advanced WYSIWYG viewer/editor supporting rich text formatting using HTML-style tags. It is optimized to handle large documents and to respond quickly to user input. It can display a large HTML subset, including tables and images.

The property "Value" contains the text of this control. "OldValue" is there as well. Use method "Append(String)" to quickly append text to the current text.

Properties:

- Property **Value** As String

Returns the text as plain text without RTF formatting.

- Property **OldValue** As String (ReadOnly)
- Property **ReadOnly** As Boolean
- Property **WordWrap** As Boolean
- Property **Flat** As Boolean (ReadOnly)

If it is true, appears as flat control. This is only useful to change the visual appearance.

- Property **Comment1Begin** As String (ReadOnly)
- Property **Comment1End** As String (ReadOnly)
- Property **Comment2Begin** As String (must be one character only / ReadOnly)

Means a one line comment, starts with one character till end of line (EOL)

- Property **Keywords** As String (CSV/ReadOnly)
- Property **Commands** As String (CSV/ReadOnly)
- Property **CommentColor** As String (ReadOnly)
- Property **KeywordColor** As String (ReadOnly)
- Property **CommandColor** As String (ReadOnly)

Methods:

- Sub **SetFontPointSize**(FontPointSize As Double)
- Sub **SetFontFamily**(FontFamily As String)
- Sub **SetFontBold**(IsFontBold As Boolean)
- Sub **SetFontItalic**(IsFontItalic As Boolean)
- Sub **SetFontUnderline**(IsFontUnderline As Boolean)
- Function **Line**() As Integer
- Function **Column**() As Integer
- Function **Selected**() As String
- Sub **Undo**()

- Sub **Redo()**
- Sub **SetTabChangesFocus**(IsTabChangesFocus As Boolean)
- Sub **RemoveAll()**
- Sub **Copy()**
- Sub **Paste()**
- Sub **Cut()**
- Sub **SelectAll()**
- Sub **InsertHtml**(Text As String)
- Sub **InsertPlainText**(Text As String)
- Sub **Append**(Text As String)

Events:

- Sub **OnEvent()** is called whenever the text has changed.

Browser

A control for the form object, provides a powerful single-page HTML viewer.

It is an advanced WYSIWYG viewer supporting rich text formatting using HTML-style tags. It can display a large HTML subset, including tables and images.

The property “**Value**” contains the text of this control. There is “**OldValue**” as well. Important properties are “**HomeURL**”, “**OpenLinks**”. Changing the Value only works, if no HomeURL has been set.

e.g. local file URL scheme, HomeURL =
`file:///C:/kbasic16/kbide/examples/projects/browser.kbasic_project/test.html`

OpenLinks=True means that clicked URL leads to open the default browser and showing the page there.

At this time only local files will be displayed. If you would like to display non-local files use the function LoadExternalBrowserWithHTML of [Application](#) instead.

It is possible to display files of the project (html files in your project directory) by relative path, if you need so, copy all html files and related files in your project directory and set HomeURL to the first page of these html files.

Methods are

- Function **BackwardAvailable()** As Boolean
- Function **ForwardAvailable()** As Boolean
- Sub **Backward()**
- Sub **Forward()**
- Sub **Reload()**

Properties are

- Property **Flat** As Boolean (ReadOnly)

If it is true, appears as flat control. This is only useful to change the visual appearance.

Events are

- Sub **OnEvent**(BackwardAvailable As Boolean, ForwardAvailable As Boolean, NewURL As String)

ProgressBar

A control for the form object, provides a horizontal progress bar.

A progress bar is used to give the user an indication of the progress of an operation and to reassure them that the application is still running. The progress bar uses the concept of steps; you give it the total number of steps and the number of steps completed so far and it will display the percentage of steps that have been completed.

MenuBarItem

A control for the menubar object, provides a powerful access to menubar items.

*Use the following static function of the class **MenuBar** to get the menubar item of the current menubar.*

- Function **MenuBar**(Name As String) As MenuItem

```
Dim i As MenuItem = MenuBar.MenuItem("File")  
i.Enabled = True
```

Properties are

- **Name** As String (ReadOnly)
- **ControlType** As String (ReadOnly)
- **Caption** As String
- **Tag** As String
- **Separator** As Boolean (ReadOnly)
- **Enabled** As Boolean
- **Icon** As String
- **Checked** As Boolean

An icon can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

- **Key** As String (ReadOnly)
- **StatusTip** As String
- **ParentControl** As String (ReadOnly/Hidden)
- **ParentIndex** As Integer (ReadOnly/Hidden)

Events are

- Sub **OnEvent**()

This event is raised, whenever the user click on the menubar item.

ToolBarItem

A control for the toolbar object, provides a powerful access to toolbar items.

*Use the following static function of the class **ToolBar** to get the toolbar item of the desired toolbar.*

- Function **ToolBar**(ToolBar As String, Name As String) As ToolBarItem

```
Dim i As ToolBarItem = ToolBar.ToolBarItem("Default", "File")
i.Enabled = True
```

Properties are

- **Name** As String (ReadOnly)
- **ControlType** As String (ReadOnly)
- **Tag** As String
- **Separator** As Boolean (ReadOnly)
- **Enabled** As Boolean
- **Icon** As String

An icon can be an absolute path to an image file (png, jpg,...) like c:\myfolder\myimage.png or can be an relative path to the current project like myimage.png (which is present in the current project directory). Relative paths are recommended.

- **ToolTip** As String
- **StatusTip** As String
- **WhatsThis** As String
- **ParentIndex** As Integer (ReadOnly/Hidden)

Events are

- Sub **OnEvent**()

This event is raised, whenever the user click on the toolbar item.

Report

The class Report is based on the class Form.

Header

A control used in reports only. Useful for creating sections on report pages.

Footer

A control used in reports only. Useful for creating sections on report pages.

ChildControl

Currently, you can only use a name of a form for displaying a child form in another form.

The property “**Value**” contains the name of the child control.

Methods are

- Function **Form()** As Form

Example

```
' change backgroundcolor in blue
ChildControl0.Background = "Blue" ' WRONG! a child control is a place holder
for a form only

Dim f As Form = ChildControl0.Form() ' RIGHT! Get the form of the child
control and use this for changing properties
f.Background = "Blue" ' change backgroundcolor in blue

Dim l As Label = f.Control("Label2")
l.Caption = "Donnerstag" ' change the caption of the label in the sub form
```

String

A string is a 8 Byte String. It is internally stored as an array of 8-byte characters with trailing 0. If you need Unicode you must use QString (part of the Qt-Bindings, not implemented yet) instead. If a Framework class method uses a String datatype it is actually Unicode QString, not a 8 Byte String.

The 'String' class is a special class, so you do not need to instantiate it with 'New', because it is automatically done for you by KBasic.

Methods are

- Function **Len()** As Integer ' returns the length of a string.
- Function **InStr**([Start As Integer ,] Sub As String) As Integer ' finds one string inside another.
- Function **Val()** As Double ' returns the numerical value of the string.
- Function **Asc()** As Integer ' returns the ASCII code for a character.
- Function **Left**(howMany As Integer) As String ' returns a string containing the first characters of a string.
- Function **Right**(howMany As Integer) As String ' returns the remaining string after truncating the source string's length according to the desired length and returns the truncated string.
- Function **LCase()** As String ' returns a new string. It contains the source string converted to all lower case.
- Function **UCase()** As String ' returns a new string. It contains the source string converted to all upper case.
- Function **Trim()** As String ' removes the source string's leading and trailing spaces.
- Function **RTrim()** As String ' function removes the source string's last spaces.
- Function **LTrim()** As String ' removes the source string's trailing spaces, from the end of the source string.
- Function **Mid**(start As Integer [, length As Integer]) As String ' get the part of a string
- Function **StrComp**(string [, compare]) As Integer ' compares to strings
- Function **Replace**(pattern As String, replace As String) As String ' replaces string occurrences with another string
- Function **StrReverse()** As String ' returns a given string reversed

Event

Declare this class in class file myEvent. It is used to receive several special events. Be sure that you create an object of that class by writing exactly as in the function “Main” below: “Dim kbEvent As New myEvent()”. Do NOT change the name kbEvent.

```
Class myEvent Inherits Event

    Sub Forms_OnFormGotFocus(FormName As String)
        Print "Forms_OnFormGotFocus! " & FormName
    End Sub

End Class

Function Main()
    Dim kbEvent As New myEvent()

    Application.Run()
End Function
```

MenuBar

You may use one MenuBar object in your application only. Name your menubar “myMenuBar” in the project window’s file, which you would like to have used by the compiler for building your application.

KBasic adds two menus to your menubar at the end. First one is the window menu, which is for handling the window list and the help menu creating two menu entries ‘Contents’ and ‘About’.

It is planned to support dynamic created menubars and toolbars at runtime.

Use the following static function of the class MenuBar to get the menubar item of the current menubar.

- Static Function **MenuBarItem**(Name As String) As MenuItem

```
Dim i As MenuItem = MenuBar.MenuBarItem("File")  
i.Enabled = True
```

Events are

- Sub **Contents_OnEvent()**
- Sub **About_OnEvent()**

```
Sub Contents_OnEvent()  
    Print "Contents clicked"  
End Sub
```

```
Sub About_OnEvent()  
    Print "About clicked"  
End Sub
```

ToolBar

You may use one ToolBar object in your application only. Name your toolbar “myToolBar” in the project window’s file, which you would like to have used by the compiler for building your application.

It is planned to support dynamic created menubars and toolbars at runtime.

*Use the following static function of the class **ToolBar** to get the toolbar item of the desired toolbar.*

- Function **ToolBarItem**(ToolBar As String, Name As String) As ToolBarItem

```
Dim i As ToolBarItem = ToolBar.ToolBarItem("Default", "File")  
i.Enabled = True
```

Math

The 'Math' class is a special class, so you do not need write "Math." in front of any static method.

Methods are

- Static Function **Abs** (numerical expression) As Double ' returns the absolute value of an argument.
- Static Function **Atn** (number) As Double ' returns the arctangent value of the argument 'number' in radians
- Static Function **Cos** (number) As Double ' returns the cosine of the argument 'number' in radians.
- Static Function **Sin** (number) As Double ' returns the sine of the argument 'number' in radians.
- Static Function **Exp** (number) As Double ' returns the exponential value of 'number'.
- Static Function **Log** (n As Double) As Long ' returns a the natural logarithm of a number.
- Static Function **Sgn**(number) As Integer ' returns the sign of the argument 'number'.
- Static Function **Fix**(number) As Long ' cuts off the trail of a number
- Static Function **Int**(number) As Long ' returns the next integer number
- Static Function **Sqr**(number) As Long ' returns the square root of the argument 'number'.
- Static Function **Tan**(number) As Long ' returns the tangent of the argument 'number' in radians.
- Static Function **Rnd**(number) As Double ' returns an integer pseudo-random number between 0 and int(EXPR)-1 inclusive.
- Static Function **Min** (numeric expression1, numeric expression2) As Double ' returns the minor value of two values
- Static Function **Max** (numeric expression1, numeric expression2) As Double ' returns the major value of two values
- Static Function **Fact** (numeric expression1) As Double ' returns the mathematical fact (n!)

Application

If an image file in your project is named `application_splash.png`, KBasic will automatically set this image file for displaying it as splash when your application starts. If an image file in your project is named `application_icon.png`, KBasic will automatically set this image file as icon file for your application. The application name is automatically set by the project name in the project property window.

Properties:

May only be used, if there is a mainwindow (see properties of project. Additionally, myMenuBar menubar class must be created):

- Static Property **X** As Integer (ReadWrite)
- Static Property **Y** As Integer (ReadWrite)
- Static Property **Width** As Integer (ReadWrite)
- Static Property **Height** As Integer (ReadWrite)

Methods:

May only be used, if there is a mainwindow (see properties of project. Additionally, myMenuBar menubar class must be created):

- Static Sub **ShowFullScreen()**
- Static Sub **ShowMaximized()**
- Static Sub **ShowMinimized()**
- Static Sub **ShowNormal()**
- Static Sub **SetFocusNext()**
- Static Sub **SetFocusPrevious()**
- Static Sub **ArrangeIcons()**
- Static Sub **Cascade()**
- Static Sub **CloseActive()**
- Static Sub **CloseAll()**
- Static Sub **Tile()**
- Static Sub **SetScrollBarsEnabled(Boolean)**

May be used anytime:

- Static Sub **MsgBox**(Title As String, Caption As String)
- Static Sub **Run()** ' this is used by KBasic to run your project. Do not call this method directly.
- Static Sub **Stop()**
- Static Function **ScreenWidth()** As Integer
- Static Function **ScreenHeight()** As Integer

- Static Function **DirectoryName()** As String
- Static Function **FileName()** As String
- Static Sub **SetStyleSheet**(Text As String)

Maybe a file in the project directory or an absolute path.

- Static Sub **SetLayoutDirection**(RightToLeft As Boolean)
- Static Sub **DoEvents()**
- Static Sub **LoadExternalBrowserWithHTML**(FileName As String)

Must be an absolute path name, meaning with “c:\...\...\...” on Windows.

- Static Function **IsSoundAvailable()** As Boolean
- Static Sub **SetCaption**(String)
- Static Sub **SetIcon**(String)

Maybe a file in the project directory or an absolute path.

- Static Sub **SetWaitCursor()**
- Static Sub **UnsetWaitCursor()**
- Static Function **ArgumentsAsString** As String()

Returns the arguments as given to the application as one single string.

Some examples

```
Application.X = 0 ' set the mainwindow to all left  
Application.Stop() ' halts your program and exits it immediately
```


Forms

It is the controller of your forms.

It opens and closes the forms (or activate or deactivate them). There are two types of objects in kbasic: visual objects, and non-visual objects. A visual object is a control and visible at runtime and lets users interact with your application; it has a screen position, a size and a foreground color. Examples of visual objects are forms and buttons. An invisible object is not visible at runtime, such as a timer. Some objects can contain other components, such as an application window containing a button. With KBasic, you add visual objects/controls to your forms to assemble applications.

Projects keep your work together. When developing an application in kbasic, you work mainly with projects. A project is a collection of files that make up your application. You create a project to manage and organize these files. KBasic provides an easy yet sophisticated system to manage the collection of files that make up a project. The project window shows each item in a project. Starting a new application with KBasic begins with the creation of a project. So before you can construct an application, you need to create a new project. A project consists of many separate files collected in one project directory, where one *.kbasic_project file is and many other files:

- *.kbasic_module
- *.kbasic_class
- *.kbasic_form
- and more

The Forms class may only be used, if there is a mainwindow (see properties of project). Additionally, myMenuBar menubar class must be created.

Open a form

Opening is easy use the following code:

```
Dim f As FORMNAME = New FORMNAME : f.Open()

' E.g. if you form is named Form1 you have to write

    Dim f As Form1 = New Form1 : f.Open()

' OR

Forms.Open("FORMNAME") ' for this call you must setup mainwindow in projects
properties
```

Methods:

- Static Function **Close**(String) As Boolean
- Static Function **Focus**() As String

Returns the name of the form, which has got the focus currently.

- Static Function **First**() As String

Returns the name of the first form in the form list.

Example

```
Dim n As String
```

```
n = Forms.First()  
If n <> "" Then  
    Do  
        Dim f As Form  
        f = Forms.Form(n)  
        ' place your code here  
        n = Forms.Next()  
    Loop While n <> ""  
End If
```

- Static Function **Next()** As String

Returns the name of the next form in the form list. If it returns an empty string, there is no further form.

- Static Function **Form(String)** As Form
- Static Sub **ShowFullScreen(String)**
- Static Sub **ShowMaximized(String)**
- Static Sub **ShowMinimized(String)**
- Static Sub **ShowNormal(String)**
- Static Sub **Show(String)**
- Static Sub **Hide(String)**
- Static Sub **SetFocus(String)**

Sets the focus to the form you wish.

- Static Function **IsOpen(String)** As Boolean
- Static Function **Open(String)** As Boolean

Events in 'Event' class:

- Static Sub **Forms_OnFormGotFocus(FormName As String)**

It is possible that FormName is "", which means no form has got focus right now. If so, you ought to set the menubar and toolbar entries disabled or enabled as it is expected to work, when no form has got focus.

Pixmaps

See the paint project example for seeing how to use the pixmaps class.

Pixmaps is a list of Pixmap.

Methods Of Pixmaps:

- Static Function **SetPixmap**(FileName As String) As Boolean
- Static Function **Pixmap**(String) As Pixmap

Colors

Colors is a list of Color.

Methods Of Colors:

- Static Sub **SetColor**(ColorId As String, R As Integer, G As Integer, B As Integer, A As Integer)

A = 0 means fully transparent, A = 255 means fully visible

- Static Function **Color**(ColorId As String) As Color

Methods Of Color:

- Sub **SetColor**(R As Integer, G As Integer, B As Integer, A As Integer)

Predefined colors are:

Color.White, Color.Black, Color.Red, Color.DarkRed, Color.Green, Color.DarkGreen, Color.Blue, Color.DarkBlue, Color.Cyan, Color.DarkCyan, Color.Magenta, Color.DarkMagenta, Color.Yellow, Color.DarkYellow, Color.Gray, Color.DarkGray, Color.LightGray, Color.Color0, Color.Color1, and Color.Transparent.

Beware that you use the predefined color objects for property control colors with the right syntax.

```
Background = Color.Red ' !Wrong!
Background = "Color.Red" ' right
Background = "Red" ' right
```

Fonts

Fonts is a list of Font.

Methods Of Fonts:

- Static Sub **SetFont**(FontId As String, Name As String, Size As Integer, Italic As Boolean, Bold As Boolean, Underline As Boolean)
- Static Function **Font**(String) As Font

Methods Of Font:

- Sub **SetFont**(Name As String, Size As Integer, Italic As Boolean, Bold As Boolean, Underline As Boolean)

Paint

You might want to use it to override the event methods of [Control](#), when you would like to implement your own controls for display data or interact with the user. You must override `OnPaint(...)` and use the following functions.

Example

Your form contains of `Box0` (control type `Box`).

```
Sub Box0_OnPaint(X As Integer, Y As Integer, Width As Integer, Height As Integer)
    DrawRect(11, 22, 33, 44)
End Sub
```

Use the following paint functions:

- Static Sub **DrawArc**(X As Integer, Y As Integer, Width As Integer, Height As Integer, StartAngle As Integer, SpanAngle As Integer)

The Qt documentation says

*The StartAngle and SpanAngle must be specified in 1/16th of a degree, i.e. a full circle equals 5760 (16 * 360). Positive values for the angles mean counter-clockwise while negative values mean the clockwise direction. Zero degrees is at the 3 o'clock position.*

- Static Sub **DrawChord**(X As Integer, Y As Integer, Width As Integer, Height As Integer, StartAngle As Integer, SpanAngle As Integer)
- Static Sub **DrawEllipse**(X As Integer, Y As Integer, Width As Integer, Height As Integer)
- Static Sub **DrawLine**(X1 As Integer, Y1 As Integer, X2 As Integer, Y2 As Integer)
- Static Sub **DrawPie**(X As Integer, Y As Integer, Width As Integer, Height As Integer, StartAngle As Integer, SpanAngle As Integer)
- Static Sub **DrawPixmap**(X As Integer, Y As Integer, Width As Integer, Height As Integer, PixmapId As String, SX As Integer, SY As Integer, SWidth As Integer, SHeight As Integer)
- Static Sub **DrawTiledPixmap**(X As Integer, Y As Integer, Width As Integer, Height As Integer, PixmapId As String, SX As Integer, SY As Integer)
- Static Sub **DrawPixmap**(X As Integer, Y As Integer, Width As Integer, Height As Integer, PixmapObject As Pixmap, SX As Integer, SY As Integer, SWidth As Integer, SHeight As Integer)
- Static Sub **DrawTiledPixmap**(X As Integer, Y As Integer, Width As Integer, Height As Integer, PixmapObject As Pixmap, SX As Integer, SY As Integer)
- Static Sub **DrawPixmap**(X As Integer, Y As Integer, PixmapId As String)
- Static Sub **DrawTiledPixmap**(X As Integer, Y As Integer, Width As Integer, Height As Integer, PixmapId As String)
- Static Sub **DrawPixmap**(X As Integer, Y As Integer, PixmapObject As Pixmap)
- Static Sub **DrawTiledPixmap**(X As Integer, Y As Integer, Width As Integer, Height As Integer, PixmapObject As Pixmap)

- Static Sub **DrawPoint**(X As Integer, Y As Integer)
- Static Sub **DrawRect**(X As Integer, Y As Integer, Width As Integer, Height As Integer)
- Static Sub **DrawRoundRect**(X As Integer, Y As Integer, Width As Integer, Height As Integer, XRnd As Integer, YRnd As Integer)
- Static Sub **DrawText**(X As Integer, Y As Integer, Text As String)
- Static Sub **SetFont**(FontId As String)
- Static Sub **SetFont**(FontObject As Font)
- Static Sub **SetPen**(ColorId As String)
- Static Sub **SetPen**(ColorId as String, Size As Integer, PenStyle As Long, PenCapStyle As Long, PenJoinStyle As Long)
- Static Sub **SetPen**(ColorObject As Color)
- Static Sub **SetPen**(ColorObject As Color, Size As Integer, PenStyle As Long, PenCapStyle As Long, PenJoinStyle As Long)
- Static Sub **SetPenPixmap**(PixmapObject As Pixmap)
- Static Sub **SetBrush**(ColorId As String)
- Static Sub **SetBrush**(ColorId As String, BrushStyle As Long)
- Static Sub **SetBrush**(ColorObject As Color)
- Static Sub **SetBrush**(ColorObject As Color, BrushStyle As Long)
- Static Sub **SetBrushPixmap**(PixmapObject As Pixmap)
- Static Sub **SetOpacity**(Double)
- Static Sub **SetBackgroundFilled**(Boolean)
- Static Sub **SetBackground**(ColorId As String)
- Static Sub **SetBackground**(ColorObject As Color)
- Static Sub **SetBackgroundPixmap**(PixmapObject As Pixmap)
- Static Sub **SetBackgroundPixmap**(PixmapId As String)
- Static Sub **FillRect**(X As Integer, Y As Integer, Width As Integer, Height As Integer, ColorId As String)
- Static Sub **FillRect**(X As Integer, Y As Integer, Width As Integer, Height As Integer, ColorObject As Color)
- Static Sub **FillRectPixmap**(X As Integer, Y As Integer, Width As Integer, Height As Integer, PixmapId As String)
- Static Sub **FillRectPixmap**(X As Integer, Y As Integer, Width As Integer, Height As Integer, PixmapObject As Pixmap)
- Static Sub **SetLayoutDirection**(RightToLeft As Boolean)

Possible values for BrushStyle are:

- Paint.NoBrush, Paint.SolidPattern, Paint.Dense1Pattern, Paint.Dense2Pattern, Paint.Dense3Pattern, Paint.Dense4Pattern, Paint.Dense5Pattern, Paint.Dense6Pattern, Paint.Dense7Pattern, Paint.HorPattern, Paint.VerPattern, Paint.CrossPattern, Paint.BDiagPattern, Paint.FDiagPattern, Paint.DiagCrossPattern, Paint.LinearGradientPattern, Paint.ConicalGradientPattern, Paint.RadialGradientPattern, Paint.TexturePattern

Possible values for PenStyle are:

- Paint.NoPen, Paint.SolidLine, Paint.DashLine, Paint.DotLine, Paint.DashDotLine, Paint.DashDotDotLine, Paint.CustomDashLine

Possible values for PenCapStyle are:

- Paint.FlatCap, Paint.SquareCap, Paint.RoundCap

Possible values for PenJoinStyle are:

- Paint.MiterJoin, Paint.BevelJoin, Paint.RoundJoin, Paint.SvgMiterJoin