Date Edit Control

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The PICS Date Edit OLE control is an edit control suitable for entering and selecting dates. The control functions much like a standard combo box control; however, instead of displaying a list of items, the Date Edit control displays a calendar from which the user can select a date.

The date edit control can manipulate dates from January 1, 1 to December 31, 9999. The date edit control accounts for the calendar reformation of 1753 and accurately displays dates prior to the reformation.

Appearance Property

{button See Also,AL(`BorderStyle',0,`0',`MAIN')} {button Properties,AL(`DateEdit Properties',0,`0',`MAIN')} {button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Returns or sets the paint style of a **DateEdit** control. Read only at run time.

Syntax

object.**Appearance**

The *object* placeholder represents an object expression that evaluates to **DateEdit** control.

Settings

The **Appearance** property settings are:

Setting	Description
0	Flat. Paints controls and forms with without visual effects.
1	(Default) 3D. Paints controls with three-dimensional effects.

Remarks

If set to 1 at design time, the **Appearance** property draws the control with three-dimensional effects. Setting the **Appearance** property to 1 also disables the BorderStyle Property.

numeric expression
Any expression that can be evaluated as a number. Elements of the expression can include any combination of keywords, variables, constants, and operators that result in a number.

Boolean expression
An expression that evaluates to either **True** or **False**.

string expression
Any expression that evaluates to a sequence of contiguous characters. Elements of the expression can include a function that returns a string, a string literal, a string constant, a string variable, a string **Variant**, or a function that returns a string **Variant** (**Var Type 8**).

object expression
An expression that specifies a particular object. The expression can include any of the object's containers. For example, if your application has an **Application** object that contains a **Document** object that contains a **Text** object, the following are valid object expressions:

```
Application.Document.Text
Application.Text
Document.Text
Text
```

constant
A named item that retains a constant value throughout the execution of a program, as opposed to a variable, whose value can change during execution. Each host application can define its own set of constants. Additional constants may be defined by the user with the **Const** statement. Constants can be used anywhere in your code in place of actual values. A constant may be a string or a numeric literal, another constant, or any combination that includes arithmetic or logical operators except **Is** or exponentiation. For example:

```
  Const A = “My String”
```

object library
A file with the .OLB extension that provides information to OLE Automation controllers (like Visual Basic) about available OLE Automation objects. You can use the Object Browser to examine the contents of an object library to get information about the objects provided.

Object Browser
A dialog box that lets you examine the contents of an object library to get information about the objects provided.

Control Panel
A set of programs that control your system configuration. You use Control Panel utilities to adjust hardware and software options, such as desktop colors, printer selections, date and number formats, fonts, and locale settings, such as language and system of measurement.

design time
The time during which you build an application in the development environment by adding controls, setting control or form properties, and so on. In contrast, during run time, you interact with the application as a user would.

run time
The time when code is running. During run time, you interact with the application as a user would.

multicolumn TreeView control
A multicolumn **TreeView** control is a **TreeView** control with its data arranged in a set of columns that go across the page. As opposed to the normal style, where all the data is arranged in a single column down the page.

Properties window
A window used to display or change properties of a selected form or control at design-time. Some custom controls have customized Property windows.

bitmap
An image represented by pixels and stored as a collection of bits in which each bit corresponds to one pixel. On color systems, more than one bit corresponds to each pixel. A bitmap usually has a .BMP filename extension.

icon
A graphical representation of an object or concept commonly used to represent minimized applications in Microsoft Windows. Essentially, an icon is a bitmap with a maximum size of 32 x 32 pixels. Icons have an .ICO filename extension.

metafile
A file that stores an image as graphical objects (lines, circles, polygons) rather than as pixels. There are two types of metafiles, standard and enhanced. Standard metafiles usually have a .WMF filename extension. Enhanced metafiles usually have an .EMF filename extension. Metafiles preserve an image more accurately than pixels when the image is resized.

Empty Branch Object

An empty **Branch** object is a **Branch** object that does not have an associated **TreeView** control item. Methods that return **Branch** objects may return an empty **Branch** object to signify the end of a list or a search failure. You can detect an empty **Branch** by using the **IsValid** member function.

BorderStyle Property

{button See Also,AL(`Appearance',0,`0',`MAIN')} {button Properties,AL(`DateEdit Properties',0,`0',`MAIN')} {button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Returns or sets the border style for a **DateEdit** object, read only at run time.

Syntax

object.**BorderStyle**

The *object* placeholder represents an object expression that evaluates to **DateEdit** control.

Settings

The **BorderStyle** property settings for a **DateEdit** control are:

Setting	Description
0	(Default) None.
1	Fixed Single.

Remarks

The **BorderStyle** property is not active unless the Appearance Property is set to 0.

BackColor Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

BackColor returns or sets the background color of an object.

ForeColor returns or sets the foreground color used to display text and graphics in an object.

Syntax

object.**BackColor** [= *color*]

object.**ForeColor** [= *color*]

The **BackColor** and **ForeColor** property syntax's have these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.
color	A value or constant that determines the background or foreground colors of an object, as described in Settings.

Settings

Visual Basic uses the Microsoft Windows operating environment red-green-blue (RGB) color scheme. The settings for color are:

Setting	Description
Normal RGB colors	Colors specified by using the Color palette or by using the **RGB** or **QBColor** functions in code.
System default colors	Colors specified by system color constants listed in the Visual Basic (VB) object library in the Object Browser. The Windows operating environment substitutes the user's choices as specified in the Control Panel settings.

For all forms and controls, the default settings at design time are:

BackColor set to the system default color specified by the constant **vbWindowBackground**.

ForeColor set to the system default color specified by the constant **vbWindowText**.

Remarks

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF).   The high byte of a number in this range equals 0; the lower 3 bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively.   The red, green, and blue components are each represented by a number between 0 and 255 (&HFF).   If the high byte isn't 0, Visual Basic uses the system colors, as defined in the user's the Control Panel settings and by constants listed in the Visual Basic (VB) object library in the Object Browser.

To display text in the Windows operating environment, both the text color must be solid.   If the text color you've selected isn't displayed, the selected colors may be dithered, that is, comprised of up to three different-colored pixels.   If you choose a dithered color for the text, the nearest solid color will be substituted.

ForeColor Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

BackColor returns or sets the background color of an object.

ForeColor returns or sets the foreground color used to display text and graphics in an object.

Syntax

object.**BackColor** [= *color*]

object.**ForeColor** [= *color*]

The **BackColor** and **ForeColor** property syntax’s have these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.
color	A value or constant that determines the background or foreground colors of an object, as described in Settings.

Settings

Visual Basic uses the Microsoft Windows operating environment red-green-blue (RGB) color scheme. The settings for color are:

Setting	Description
Normal RGB colors	Colors specified by using the Color palette or by using the **RGB** or **QBColor** functions in code.
System default colors	Colors specified by system color constants listed in the Visual Basic (VB) object library in the Object Browser. The Windows operating environment substitutes the user's choices as specified in the Control Panel settings.

For all forms and controls, the default settings at design time are:

BackColor set to the system default color specified by the constant **vbWindowBackground**.

ForeColor set to the system default color specified by the constant **vbWindowText**.

Remarks

The valid range for a normal RGB color is 0 to 16,777,215 (&HFFFFFF). The high byte of a number in this range equals 0; the lower 3 bytes, from least to most significant byte, determine the amount of red, green, and blue, respectively. The red, green, and blue components are each represented by a number between 0 and 255 (&HFF). If the high byte isn't 0, Visual Basic uses the system colors, as defined in the user's the Control Panel settings and by constants listed in the Visual Basic (VB) object library in the Object Browser.

To display text in the Windows operating environment, both the text color must be solid. If the text color you've selected isn’t displayed, the selected colors may be dithered, that is, comprised of up to three different-colored pixels. If you choose a dithered color for the text, the nearest solid color will be substituted.

Font Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Returns a **Font** object.

Syntax

object.**Font**

The **Font** property syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.

Remarks

Use the **Font** property of an object to identify a specific **Font** object whose properties you want to use. For example, the following code changes the Bold property setting of a **Font** object identified by the **Font** property of a TextBox object:

```
txtFirstName.Font.Bold = True
```

ShowCentury Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Controls the way the year is displayed in a **DateEdit** control.

Syntax
object.**ShowCentury** [= [**True** | **False** ||

The **ShowCentry** property syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** Control.
TRUE | FALSE	A value **True** causes the **DateEdit** control to display the full year, including the century. Only the last two digits of the year are displayed if this property is set to **False**.

Remarks
When the **ShowCentury** property is set to **False**, dates edited in the **DateEdit** control retain the century of the original date. The user may change the century by selecting the appropriate year in the calendar dropdown.

Value Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **Value** property represents the date in the **DateEdit** control.

Syntax

object [= date]

object.**Value** [= date]

The **Value** property syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** Control.
date	A date value.

Remarks

The **Value** property is the default property of the **DateEdit** control.

EnableCalendar Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **EnableCalendar** property controls whether or not the calendar drop down button is displayed on the **DateEdit** control.

Syntax

object. **EnableCalendar** [= [**True** | **False** ||

The **EnableCalendar** property syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** Control.
TRUE | *FALSE*	A value **True** causes the **DateEdit** control to display the calendar drop down button. The button is not displayed if this property is set to **False**.

Remarks

The **EnableCalendar** property controls whether or not the calendar drop down button is displayed on the **DateEdit** control. If the **EnableCalendar** property is set to **True**, the **DateEdit** control displays a calendar drop down button, allowing the user to set the date using the mouse. When set to **False**, no calendar drop down button is displayed.

DateFormat Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **DateFormat** property controls the display format of the date presented in the **DateEdit** control.

Syntax
object. **DateFormat** [= %format%|

The **DateFormat** property syntax has these parts:

Part	**Description**
object	An object expression that evaluates to a **DateEdit** Control.
%format%	Must be one of the following manifest constants

fmtMDY	Displays the date in the **mm/dd/yy** format.
fmtDMY	Displays the date in the **dd/mm/yy** format
fmtYMD	Displays the date in the **yy/mm/dd** format
fmtMonDY	Displays the date in the **Mon dd, yyyy** format.
fmtDMonY	Displays the date in the **dd Mon yyyy** format
fmtYMonD	Displays the date in the **yyyy Mon dd** format
fmtMonthDY	Displays the date in the **Month dd, yyyy** format.
fmtDMonthY	Displays the date in the **dd Month yyyy** format
fmtYMonthD	Displays the date in the **yyyy Month dd** format
fmtDayMDY	Displays the date in the **Day Mon dd, yyyy** format.
fmtDayDMY	Displays the date in the **Day dd Mon yyyy** format
fmtDayYMD	Displays the date in the **Day yyyy Mon dd** format

Enabled Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Returns or sets a value that determines whether a form or control can respond to user-generated events.

Syntax

object.**Enabled** [= *boolean*]

The **Enabled** property syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.
boolean	A Boolean expression that specifies whether *object* can respond to user-generated events.

Settings

The settings for boolean are:

Setting	Description
True	(Default) Allows *object* to respond to events.
False	Prevents *object* from responding to events.

Remarks

The **Enabled** property allows forms and controls to be enabled or disabled at run time.   For example, you can disable objects that don't apply to the current state of the application.   You can also disable a control used purely for display purposes, such as a text box that provides read-only information.

IgnoreKeyboard Property

{button See Also,AL(`Enabled',0,`0',`MAIN')} {button Properties,AL(`DateEdit Properties',0,`0',`MAIN')} {button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **IgnoreKeyboard** property controls whether nor not the **DateEdit** control will accept keyboard input.

Syntax

object. **IgnoreKeyboard** [= [**True** | **False**]]

The **IgnoreKeyboard** property syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** Control.
TRUE | *FALSE*	A value **True** disables keyboard input for the control. An enabled control can accept keyboard input. A value of **False** enables keyboard input for the control.

Remarks

The **IgnoreKeyboard** property works in much the same way as the Enabled Property. However, the **IgnoreKeyboard** only enables or disables keyboard input. A control with keyboard input disabled can still accept and respond to mouse events.

LeadingZero Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **LeadingZero** property controls the way numbers below 10 are displayed in the date.

Syntax
object. **LeadingZero** [= [**True** | **False**]]

The **LeadingZero** property syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** Control.
TRUE | *FALSE*	A value **True** causes all numbers below 10 to be displayed with a leading zero. For example, the date February 3, 1995 will be displayed as 02/03/1995. A value of **False** will not force a leading zero on numbers below 10.

hWnd Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Returns a handle to a **DateEdit** control. Read-only at run time, not available at design time.

Syntax

object.**hWnd**

The *object* placeholder represents an object expression that evaluates to **DateEdit** control.

Remarks

The Microsoft Windows operating environment identifies each form and control in an application by assigning it a handle, or **hWnd**. The **hWnd** property is used with Windows API calls. Many Windows operating environment functions require the **hWnd** of the active window as an argument.

Note   Because the value of this property can change while a program is running, never store the **hWnd** value in a variable.

Seperator Property

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **Seperator** property is the character used to separate the different parts of a date when the date is shown in the mm/dd/yy, dd/mm/yy or yy/mm/dd formats.

Syntax

object. **Seperator** [= SeperatorCharacter$]

The **Seperator** property syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** Control.
SeperatorCharacter$	A single character string that holds the seperator character. If the string contains more than one character, the first character of the string is used as the seperator character.

Remarks

The **Seperator** property is used to separate the different parts of the date. The seperator character is placed between the month, day and year portions of the date when the date is displayed in the mm/dd/yy, dd/mm/yy or yy/mm/dd formats.

GetDateString Method

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Returns a string representation of the date.

Important   This object requires either Microsoft Windows 95 or Microsoft Windows NT version 3.51 or higher.

Syntax

BSTR *object*.**GetDateString**

The **GetDateString** method syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.

Returns

The method returns a string that represents the date in the control.

AboutBox Method

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Use the **About** command to find the version number and other pertinent information about PICS.

You have specified a number outside of the range 1-12 in the **SetMonthLiteral** method. The first argument of the **SetMonthLiteral** method specifies which month literal is to be changed. This value must fall between the numbers 1 (for January) through 12 (for December).

See SetMonthLiteral Method

You have attempted to change the value of the **Appearance** property at run-time. The **Appearance** property is read-only at run time. You may change the value of this property in the property sheet for the Date Edit control at design time.

See Appearance Property

You have attempted to change the value of the **BorderStyle** property at run-time. The **BorderStyle** property is read-only at run time. You may change the value of this property in the property sheet for the Date Edit control at design time.

See BorderStyle Property

Contact ProtoView

You may contact ProtoView Development Corporation at:

Internet :	http://www.protoview.com
	info@protoview.com
CompuServe:	GO PROTOVIEW
	75300,3510
Phone:	(800) 231-8588
	(609) 655-5000
Fax:	(609) 655-5353

SetDayLiteral Method

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Sets the text used to represent the day of the week in the date string.

Important This object requires either Microsoft Windows 95 or Microsoft Windows NT version 3.51 or higher.

Syntax

object.**SetDayLiteral** (*index*, *literal*)

The **SetDayLiteral** method syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.
index	The day that is to be changed. Must be an integer number between 1 and 7, where 1 represents Sunday, 2 represents Monday, etc.
literal	The text string that will represent the day of the week.

Remarks

You can use this method to make the **DateEdit** control suitable for international programs.

SetMonthLiteral Method

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Sets the text used to represent the month in the date string.

Important This object requires either Microsoft Windows 95 or Microsoft Windows NT version 3.51 or higher.

Syntax

object.**SetMonthLiteral** (*index*, *literal*)

The **SetMonthLiteral** method syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.
index	The month that is to be changed. Must be an integer number between 1 and 12.
literal	The text string that will represent the month.

Remarks

You can use this method to make the **DateEdit** control suitable for international programs.

Change Event

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **Change** event is fired whenever the contents of the **DateEdit** control changes.

Syntax

*void object.***Change**()

The **Change** method syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.

HideDropDown Event

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **HideDropDown** event is fired whenever the calendar drop down is about to be hidden.

Syntax

*void object.***HideDropDown**()

The **HideDropDown** method syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.

SetAbbrvMonthLiteral Method

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

Sets the text to be used for abbreviated months in the date string.

Important This object requires either Microsoft Windows 95 or Microsoft Windows NT version 3.51 or higher.

Syntax

object.**SetAbbrvMonthLiteral** (*index*, *literal*)

The **SetAbbrvMonthLiteral** method syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.
index	The month that is to be changed. Must be an integer number between 1 and 12.
literal	The text string that will represent the month.

Remarks

You can use this method to make the **DateEdit** control suitable for international programs.

ShowDropDown Event

{button Properties,AL(`DateEdit Properties',0,`0',`MAIN')}{button Methods,AL(`DateEdit Methods',0,`0',`MAIN')} {button Events,AL(`DateEdit Events',0,`0',`MAIN')}

The **ShowDropDown** event is fired whenever the calendar drop down is about to be shown.

Syntax

*void object.***ShowDropDown**()

The **ShowDropDown** method syntax has these parts:

Part	Description
object	An object expression that evaluates to a **DateEdit** control.