About Property

Description

The **About** property displays the about box for this component.   The about box contains version and copyright information.   It is only available at design time.

About SHORELINE SOFTWARE

About SHORELINE SOFTWARE

SHORELINE is a developer haven created to provide excellent products and support. Our developers have been creating MS-Windows based applications since 1985 and real life client server applications since 1987. We are bringing our real world experience into the design and development of unique controls. We are a TRANSDOMINION Company and are part of a software family including the PRISM Client Server Group. Our family of companies offer many services that include:

- Component development
- Application development (off-site)
- Multi-media design and development
- Install development
- Technology transfer
- Training

How to purchase any of our products:

You can order directly from SHORELINE at 800-261-9198 or contact your favorite dealer.SHORELINE accepts VISA, Mastercard or the Discover card. Dealer prices may vary.

How to contact SHORELINE SOFTWARE about our products or service:

We welcome all ideas, comments and suggestions. Please let us know what you think of our pricing and support policies. If you have a unique story about our products please share it with us; our developers love to hear how our products are being used!

CompuServe:	70541, 2436
US Mail:	35-31 Talcottville Road, #123
	Vernon, CT 06066-4030
Phone:	800-261-9198
Fax:	203-870-5727
Contact:	Glenn A. Field

SHORELINE is always looking for additional talent. We even work with several remote developers for our products. If you feel you have what it takes to be part of SHORELINE send us your cv or resume.
Where to send your resume

Alignment Property

See Also

Declaration

property Alignment: TTextAlignment;

Description

The **Alignment** property specifies how text is aligned within the component.

These are the possible values:

Value	Meaning
taLeft	Align text to the left side of the control
taCenter	Center text horizontally in the control
taRight	Align text to the right side of the control

Bitmap Property

See Also Example

Declaration

property Bitmaps : TBitMap;

Description

The **Bitmaps** property for HelpCloud contains a set of four TBitMap objects which are used to define the background for the four possible helpcloud locations. The Corner property designates which of these four bitmaps is displayed at any given time.

CaptionWidth

See Also

Declaration

property CaptionWidth : Integer;

Description

The **CaptionWidth** property sets the width of the help cloud caption in pixels. When you increase the CaptionWidth property value, the displayed **Text** property of the control becomes wider. If you decrease the value, the   displayed **Text** property of the control becomes narrower.

Color

See Also

These are the possible values of Color:

Value	**Meaning**
clBlack	Black
clMaroon	Maroon
clGreen	Green
clOlive	Olive green
clNavy	Navy blue
clPurple	Purple
clTeal	Teal
clGray	Gray
clSilver	Silver
clRed	Red
clLime	Lime green
clBlue	Blue
clFuchsia	Fuchsia
clAqua	Aqua
clWhite	White
clBackground	Current color of your Windows background
clActiveCaption	Current color of the title bar of the active window
clInactiveCaption	Current color of the title bar of inactive windows
clMenu	Current background color of menus
clWindow	Current background color of windows
clWindowFrame	Current color of window frames
clMenuText	Current color of text on menus
clWindowText	Current color of text in windows
clCaptionText	Current color of the text on the title bar of the active window
clActiveBorder	Current border color of the active window
clInactiveBorder	Current border color of inactive windows
clAppWorkSpace	Current color of the application workspace
clHighlight	Current background color of selected text
clHightlightText	Current color of selected text
clBtnFace	Current color of a button face
clBtnShadow	Current color of a shadow cast by a button
clGrayText	Current color of text that is dimmed
clBtnText	Current color of text on a button
clInactiveCaptionText	Current color of the text on the title bar of an inactive window
clBtnHighlight	Current color of the highlighting on a button

The second half of the colors listed here are Windows system colors. The color that appears depends on the color scheme users are using for Windows. Users can change these colors using the Control Panel in

Program Manager. The actual color that appears will vary from system to system. For example, the color fuchsia may appear more blue on one system than another.

When you use the Color dialog box to select a color, you are assigning a new color value to the dialog box's Color property. You can then use the value within the Color property, and assign it to the Color property of another control.

Corner

See Also

Declaration

property Corner : THelpCloudCorner;

Description

The **Corner** property defines at which corner location the help cloud appears at runtime.

Create Method

See Also

NOTE: For more information regarding Create methods please refer to the Delphi help.

Declaration

constructor Create;

Description

The **Create** method constructs a new object instance. Create returns an instance of the type being created, allocated on the global heap. As with all constructors, Create calls the NewInstance method to allocate the memory for the instance, and the InitInstance method to initialize the allocated memory before executing its own code.

By default, Create allocates the number of bytes returned by the InstanceSize method, and initializes the allocated memory to zeros.

When declaring new component types, always add the **override** directive if your new component declares a Create method. The Create method of TComponent is virtual, so to ensure that Delphi calls the correct constructor when a user drops your component on a form, you must override the Create method.

Note:

When you override the Create constructor in a descendant object type, you should call the inherited Create to complete the initialization of inherited fields and properties. Always use the inherited keyword when calling the inherited Create, rather than specifying the ancestor type, as calling AncestorType.Create actually constructs an additional instance of that ancestor type.

Destroy Method

See Also

NOTE: For more information regarding Destroy methods please refer to the Delphi help.

Declaration

destructor Destroy;

Description

The **Destroy** method destroys the object, component, or control and releases the memory allocated to it.

You seldom need to call Destroy. Objects designed with Delphi create and destroy themselves as needed, so you don't have to worry about it. If you construct an object by calling the **Create** method, you should call **Free** to release memory and dispose of the object.

Example for Alignment property

Example

This example moves the text of the control to the right of the HelpCloud1, during the creation of TForm1:

To see the **Alignment** property work **<u>during design time</u>** try the following:

1) **Save all of your current work**

2) Create a new project

- 2a) Select New Project from the Delphi File menu

3) **Place the THelpCloud on the form**

- 3a) Click on the VisualPROS-1 tab found in VCL tabs
- 3b) Select the HelpCloud icon and double click

4) **Place the TEdit on the form**

- 4a) Click on the STANDARD tab found in VCL tabs
- 4b) Select the Edit icon and double click

5) **Change the HelpCloud Alignment property**

- 5a) Click on Object Inspector window
- 5b) Select HelpCloud1 control from the component list
- 5c) Select the drop down list for the **Alignment** property
- 5d) Select taLeft from the list
- 5e) HelpCloud1 should be at the bottom of the form expanded to the width of the form

6) **Change the Edit1 property to see the actual alignment change**

- 6a) Click on Object Inspector window
- 6b) Select Edit1 control from the component list
- 6c) Select the Hint property and add the text you want to display in the HelpCloud
- 6d) Select the ShowHint property and select True from the dropdown list
- 6e) HelpCloud1 should be at the bottom of the form expanded to the width of the form

To see the **Align** property work **<u>during program execution</u>** try the following:

1) **Save all of your current work**

2) Create a new project

- 2a) Select New Project from the Delphi File menu

3) **Place the THelpCloud on the form**

- 3a) Click on the VisualPROS-1 tab found in VCL tabs
- 3b) Select the HelpCloud icon and double click

4) **Create the actual code for the alignment change**

- 4a) Double click on TForm1 in a blank area
- 4b) Type in the following line of code in the edit window after the begin.

```
HelpCloud1.Align := alBottom
```

Your code should look like the following

```
procedure TForm1.FormCreate(Sender: TObject);
begin
        HelpCloud1.Align := alBottom
end;
end.
```

5) **Try the application and see the results**

- 5a) Press F9 and watch your new form

Example for Bitmaps Property

Example

This example assigns a bitmap for the background of a HelpCloud component named HelpCloud1. This example will use the arcade.bmp found in your windows directory:

To assign the **Bitmaps** property **during design time** do the following:

1) Save all of your current work

2) Create a new project

- 2a) Select New Project from the Delphi File menu

3) Place a HelpCloud instance on the form

- 3a) Click on the VisualPROS-1 tab found in VCL tabs
- 3b) Select the HelpCloud icon and click on the form

4) Change the Bitmaps property

- 4a) Click on the Object Inspector window
- 4b) Double click the **Bitmaps** property
- 4c) Double click on one of the four indented properties (LowerLeft through UpperRight)
- 4d) Click Load from the Picture Editor dialog
- 4e) In the Load Picture dialog navigate to your MS-Windows subdirectory
- 4f) Double click the arcade.bmp found in the file list
- 4g) Click OK on the Picture Editor dialog
- 4h) Set the Corner property to indicate the above selected bitmap corner location

5) Try the application and see the results

- 5a) Press F9 and point your mouse on hint enabled components.

To assign the **Bitmaps** property **during program execution** do the following:

1) Save all of your current work

2) Create a new project

- 2a) Select New Project from the Delphi File menu.
- 2b) Add several buttons and/or other hint enabled components to the form.
- 2c) Add text to the hint property of each component.
- 2d) Set the ShowHint Property of each component to true.

3) Place a HelpCloud instance on the form

- 3a) Click on the VisualPROS-1 tab found in VCL tabs
- 3b) Select the HelpCloud icon and click on the form

4) Create the actual code for the bitmap change

- 4a) Double click on a blank area of the form
- 4b) Type the following lines of code into the FormCreate procedure

NOTE: Substitute your MS-Windows path if it is different than C:\WINDOWS\

```
Form1.Helpcloud1.BitMaps.UpperRight.LoadFromFile('c:\windows\arcade.bmp');
Form1.HelpCloud1.Corner :=UpperRight;
```

Your code should look like the following;

```
procedure TForm1.FormCreate(Sender: TObject);
begin
        Form1.Helpcloud1.BitMaps.UpperRight.LoadFromFile('c:\windows\arcade.bmp');
        Form1.HelpCloud1.Corner :=UpperRight;
end;
end.
```

5) Try the application and see the results

- 5a) Press F9 and point your mouse on hint enabled components.
  If you receive an error check your pathname of the bitmap file.

Font Property

See Also

Declaration

property Font: TFont;

Description

The **Font** property is a font object that controls the attributes of text written on or in the component, or sent to the printer. To modify a font, change the value of the Color, Name, Size or Style properties of the font object.

Key Property

The most significant properties: Generally having an effect on the behavior of the component. You probably want to learn these properties first.

Methods

Class methods are procedures and functions that operate on a class, instead of an instance of the class. The implementation of the class method must not depend on the run-time values of any object fields.

To declare a class method, you put the reserved word class in front of the procedure, or function keyword, that starts the definition.

In the defining declaration of a class method, the identifier Self represents the class for which the method was activated. The type of Self in a class method is class of ClassType, where ClassType is the class type for which the method is implemented. Since Self does not represent an object reference in a class method, it is not possible to use Self to access fields, properties and normal methods. It is however possible to call constructors and other class methods through Self.

A class method can be invoked through a class reference or an object reference. When invoked through an object reference, the class of the given object reference is passed as the **Self** parameter.

Name Property

See Also

Declaration

property Name: THelpCloud;

Description

The **Name** property contains the name of the component as referenced by other components. By default, Delphi assigns sequential names based on the type of the component, such as 'HelpCloud1', 'HelpCloud2' and so on. You may change these to suit your needs.

Note: Change component names only at design time.

Overriding Methods

NOTE: For more information regarding overriding methods please refer to the Delphi help.

Overriding a method means extending or refining it, rather than replacing it. That is, a descendant object type can redeclare and reimplement any of the methods declared in its ancestors. One cannot override static methods, because declaring a static method with the same name as an inherited static method replaces the inherited method completely.

To override a method in a descendant object type, add the directive **override** to the end of the method declaration.

Using **override** will cause a compile-time error if:

The method does not exist in the ancestor object

The ancestor's method of that name is static

The declarations are not otherwise identical (names and types of parameters, procedure vs. function, and so on)

Properties

About
Alignment
Bitmaps
CaptionWidth
Color
Corner
Font
Name
Style
Tag

Read Only

Properties with this symbol are read only and cannot be changed.   Usually provided so that application code can inspect certain characteristics of a component.

Runtime Only

Properties with this symbol are only available at runtime. In design mode these properties will not be shown in the object inspector. Most runtime only properties are read only as well.

See Also

About
Alignment
Bitmaps
CaptionWidth
Color
Corner
Font
Name
Style
Tag

Style Property

See Also

Declaration

property Style:

Description

The value of the **Style** property determines the appearance of the help cloud. These are the possible values:

Value	**Meaning**
standard	The hint appears connected to the component by a solid cloud.
bubble	The hint appears connected to the component by a "bubble" cloud.

TTextAlignment Type

Declaration

TTextAlignment = (taLeft, taRight, taCenter);

Description

TAlignment is the type of the alignment property.

TColor Type

Declaration

```
type
  TColor = -(COLOR_ENDCOLORS + 1)..$02FFFFFF;
```

Description

The **TColor** type is used to specify the color of an object.

The Graphics unit contains definitions of useful constants for TColor. These constants map either directly to the closest matching color in the system palette (for example, clBlue maps to blue), or to the corresponding system screen element color, defined in the Color section of the Windows Control panel (for example, clBtnFace maps to the system color for button faces).

The constants that map to the closest matching system colors are: clAqua, clBlack, clBlue, clDkGray, clFuchsia, clGray, clGreen, clLime, clLtGray, clMaroon, clNavy, clOlive, clPurple, clRed, clSilver, clTeal, clWhite and clYellow.

The constants that map to the system screen element colors are: clActiveBorder, clActiveCaption, clAppWorkSpace, clBackground, clBtnFace, clBtnHighlight, clBtnShadow, clBtnText, clCaptionText, clGrayText, clHighlight, clHighlightText, clInactiveBorder, clInactiveCaption, clInactiveCaptionText, clMenu, clMenuText, clScrollBar, clWindow, clWindowFrame and clWindowText.

If you specify TColor as a specific 4-byte hexadecimal number instead of using the constants defined in the graphics unit, the low three bytes represent RGB color intensities for blue, green, and red, respectively. The value $00FF0000 represents full-intensity, pure blue, $0000FF00 is pure green, and $000000FF is pure red. $00000000 is black and $00FFFFFF is white.

If the highest-order byte is zero ($00), then the color obtained is the closest matching color in the system palette. If the highest-order byte is one ($01), the color obtained is the closest matching color in the currently realized palette. If the highest-order byte is two ($02), the value is matched with the nearest color in the logical palette of the current device context.

To work with logical palettes, you must select the palette with the Windows API function SelectPalette. To realize a palette, you must use the Windows API function RealizePalette.

TFont Object

NOTE: This information provided for reference only. For more information refer to the Delphi help.

Description

A **TFont** object defines the appearance of text. TFont encapsulates a Windows HFONT.

A TFont object defines a set of characters by specifying their height, font family (typeface), name and so on. The height is specified by the Height property; The typeface is specified by the Name property; The size in points is specified by the Size property; The color is specified by the Color property; The attributes of the font (bold, italic, and so on) are specified by the Style property.

When a font is modified, an OnChange event occurs.

THelpCloudCorner

Declaration
THelpCloudCorner = (LowerLeft, LowerRight, UpperLeft, UpperRight);

Description
THelpCloudCorner is the data type for the Corner property of the HelpCloud Component.

THelpCloud Component

Properties   Methods

Version 1.0 HelpCloud Component Help, © 1994, 1995 SHORELINE SOFTWARE
VisualPROS is a trademark of SHORELINE SOFTWARE

SHORELINE SOFTWARE   **35-31 Talcottville Rd.   #123, Vernon, CT 06066-4030**
Technical Support: **Phone: (203) 870-5707 24-Hour Fax: (203) 870-5727**
CompuServe 70541,2436

Description
HelpCloud provides an enhanced form of Delphi hint functionality with a stylized appearance. HelpCloud also allows manipulation of shape plus characteristics of help view.

Feature List

User Defined background bitmapping.
Four possible locations of the help cloud
Two possible styles of help cloud.

Key Properties

Name
BitMaps
Corner
Style
Font

About SHORELINE SOFTWARE

Tag Property
See Also
Declaration
property Tag: **Longint**;

Description
The **Tag** property is available to store an integer value as part of a component. While the Tag property has no meaning to Delphi, your application can use the property to store a value for its special needs.

Where to send your resume

Please send your resume and other CV related materials to:

SHORELINE SOFTWARE
35-31 Talcottville Road, #123
Vernon, CT 06066-4030

ATTN: Glenn A. Field

Phone:	800-261-9198
Fax:	203-870-5727

We will contact you after receiving your resume.