

## DTools Contents



[TAnalogClock](#)



[TBalloonHint](#)



[TLEDClock](#)



[TLEDLabel](#)



[TNeatoMeter](#)



[TPieMeter](#)



[TRotaryKnob](#)



[TVisualApp](#)

## Active Property

### Applies to

TBalloonHint object

### Declaration

**property** Active: Boolean;

### Description

The Active property determines if balloon hints will be displayed. If Active is False, then standard Delphi hints will be displayed.



## TBalloonHint Component

[Properties](#)

[Method](#)

### Unit

[Balloon](#)

### Description

TBalloonHint is a descendent of TComponent. TBalloonHint is a component to display hint strings in a cartoon style balloon. Several properties are provided to customize the appearance of the balloon.

To add balloon hints to your application, simply place a TBalloonHint control on your main form and set the Active property to True.

### Color

The color of the balloons is determined by the Application.HintColor property. The outline of the balloon is drawn as clWindowFrame.

### Font

Balloons use the standard THintWindow font (if you have not modified the VCL source code, this will be MS Sans Serif 8 point regular).

## Properties

▶ Run-time only

🔑 Key Properties

🔑 Active

🔑 MaxWidth

🔑 Position

🔑 ShadowDepth

🔑 Shape

## MaxWidth Property

### Applies to

TBalloonHint object

### Declaration

**property** MaxWidth: Integer;

### Description

The MaxWidth property determines the maximum width in pixels the balloon will occupy on the screen. If this value is less than zero, TBalloonHint will use the absolute value of the number as a divisor to the width of the screen. For example: If MaxWidth = -4, the maximum width in pixels would be Screen.Width div 4. To get the actual maximum width in pixels no matter what MaxWidth is set to, use the GetMaxWidthPixels method.

## Position Property

### Applies to

TBalloonHint, TRotaryKnob objects

### Declaration

TBalloonHint:

**property** Position: TBalloonPosition;

TRotaryKnob:

**property** Position: Integer;

### Description

TBalloonHint:

The Position property determines the preferred location to display the balloon hint.

TRotaryKnob:

The Position property determines position of the indicator on the knob.

## ShadowDepth Property

### Applies to

TBalloonHint object

### Declaration

**property** ShadowDepth: TShadowDepth;

### Description

The ShadowDepth property determines the number of pixels to offset the balloon shadow.

## Shape Property

### Applies to

TAnalogClock, TBalloonHint, TPieMeter objects

### Declaration

TAnalogClock:

**property** Shape: TAnalogClockShape;

TBalloonHint:

**property** Shape: TBalloonShape;

TPieMeter

**property** Shape: TPieShape;

### Description

The Shape property determines the basic shape or outline of the object.



## TBalloonShape Type

### Unit

Balloon

### Declaration

```
TBalloonShape = (bsRoundRect, bsRectangle);
```

### Description

The TBalloonShape type is used by the Shape property to determine the shape of a TBalloonHint component.

## TBalloonPosition Type

### Unit

Balloon

### Declaration

```
TBalloonPosition = (bpAboveLeft, bpAboveRight, bpBelowLeft, bpBelowRight);
```

### Description

The TBalloonPosition type is used by the Position property of the TBalloonHint component to determine the default positioning of the balloon.

## TShadowDepth Type

### Unit

Balloon

### Declaration

```
TShadowDepth = 0..16;
```

### Description

The TShadowDepth type is used by the ShadowDepth property to determine the pixel offset of the balloon shadow of a TBalloonHint object.

## Balloon Unit

The Balloon unit contains the classes and types used to implement balloon hints.

The following items are declared in the Balloon unit:

### Objects

TBalloonHint

### Types

TBalloonShape

TBalloonPosition

TShadowDepth



## TNeatoMeter Component

Properties

**Unit**

Feedback

### Description

TNeatoMeter is a descendent of TGraphicControl. TNeatoMeter is a component to give user feedback for lengthy operations.

## Feedback Unit

The Feedback unit contains the classes and types used to implement progress meters.

The following items are declared in the Feedback unit:

### Objects

TNeatoMeter

TPieMeter

### Types

TBevelDepth

TBevelType

TBitmapDrawStyle


TMeterDirection

TMeterStyle












TPieDirection

TPieShape

## Properties

 Run-time only

 Key Properties

 <u>BackColor</u>	 <u>Completed</u>	<u>ShowHint</u>
 <u>BevelDepth</u>	 <u>Direction</u>	 <u>ShowPercent</u>
 <u>BevelType</u>	<u>Font</u>	<u>Style</u>
 <u>Bitmap</u>	<u>ForeColor</u>	 <u>Total</u>
 <u>BitmapDrawStyle</u>	<u>ParenFont</u>	 <u>UseFontColor</u>
<u>BorderStyle</u>	<u>ParentShowHint</u>	<u>Visible</u>
<u>Caption</u>	 <u>Percent</u>	

## TBevelType Type

### Unit

### Feedback

### Declaration

```
TBevelType = (btNone, btInset, btRaised);
```

### Description

The TBevelType type is used by the BevelType property to give a TNeatoMeter component 3-D appearance.



## BackColor Property

### Applies to

TAnalogClock, TLEDClock, TLEDLabel, TNeatoMeter, TPieMeter objects

### Declaration

**property** BackColor: TColor

### Description

TAnalogClock:

The BackColor property determines the color of the area around the clock.

TLEDClock and TLEDLabel

The BackColor property determines the color of the area around the segments.

TNeatoMeter and TPieMeter:

The BackColor property determines the color of the incomplete area of the meter.

## BevelDepth Property

### Applies to

TNeatoMeter object

### Declaration

**property** BevelDepth: TBevelDepth;

### Description

The BevelDepth property is used to set the 3-D depth of the meter.

## BevelType Property

### Applies to

TNeatoMeter object

### Declaration

**property** BevelType: TBevelType;

### Description

The BevelType property is used to give a meter a 3-D appearance.

## Bitmap Property

### Applies to

TNeatoMeter object

### Declaration

**property** Bitmap: TBitmap;

### Description

The Bitmap property is used to show progress with a graphic instead of simple filled rectangles. The BitmapDrawStyle property determines the appearance of the bitmap.

## BitmapDrawStyle Property

### Applies to

TNeatoMeter object

### Declaration

**property** BitmapDrawStyle: TBitmapDrawStyle;

### Description

The BitmapDrawStyle property is used to determine how the bitmap will be displayed in a meter.

## Percent Property

### Applies to

TNeatoMeter, TPieMeter objects

### Declaration

**property** Percent: Integer;

### Description

The Percent property indicates the amount completed.

## Caption Property

### Applies to

TNeatoMeter, TPieMeter objects

### Declaration

**property** Caption: **string**;

### Description

The Caption property contains the text that will be displayed on the meter. If Caption is an empty string and ShowPercent is True, the percent complete will be displayed.

## Completed Property

### Applies to

TNeatoMeter, TPieMeter objects

### Declaration

**property** Completed: **Longint**;

### Description

The Completed property determines how many items out of a possible Total have been completed.



## Direction Property

### Applies to

TNeatoMeter, TPieMeter objects

### Declaration

TNeatoMeter

**property** Direction: TMeterDirection;

TPieMeter

**property** Direction: TPieDirection;

### Description

The Direction property determines the way a meter will indicate progress.

## ForeColor Property

### Applies to

TNeatoMeter, TPieMeter objects

### Declaration

**property** ForeColor: TColor;

### Description

The ForeColor property determines the color of the complete area of the meter.



## **TPieMeter Component**

Properties

**Unit**

Feedback

### **Description**

TPieMeter component is a descendent of TGraphicControl. TPieMeter is a component to give user feedback for lengthy operations.

## ShowPercent Property

### Applies to

TNeatoMeter, TPieMeter objects

### Declaration

**property** ShowPercent: Boolean;

### Description

The ShowPercent property determines whether or not the percent complete will be displayed when Caption is an empty string.

## Style Property

### Applies to

TNeatoMeter object

### Declaration

**property** Style: TMeterStyle;

### Description

The Style property determines the look of the meter.

## Total Property

### Applies to

TNeatoMeter, TPieMeter objects

### Declaration

**property** Total: Longint;

### Description

The Total property determines the number of Completed items required to reach 100 percent.

## UseFontColor Property

### Applies to

TNeatoMeter object

### Declaration

**property** UseFontColor: Boolean;

### Description

The UseFontColor property determines whether text displayed on the meter will be displayed using the color of the font or using the inverse color of the meter sections.

**Note:** When a bitmap has been assigned, the meter will always use the font color.

## TBevelDepth Type

### Unit

[Feedback](#)

### Declaration

```
TBevelDepth = 0..10;
```

### Description

The TBevelDepth type is used by the [BevelDepth](#) property to set the 3-D depth of a [TNeatoMeter](#) component.



## TBitmapDrawStyle Type

### Unit

[Feedback](#)

### Declaration

```
TBitmapDrawStyle = (dsStretch, dsTile, dsTileInvert);
```

### Description

The TBitmapDrawStyle type is used by the [BitmapDrawStyle](#) property to determine how the [bitmap](#) will be displayed in a [TNeatoMeter](#) object. The following table describes the meaning of each value:

Value	Meaning
dsStretch	The bitmap will be stretched in the completed section of the meter. The remainder of the meter will be filled with the background color.
dsTile	The bitmap will be tiled in the completed section of the meter. The remainder of the meter will be filled with the background color.
dsTileInvert	The bitmap will be tiled in the completed section of the meter. The remainder of the meter will be tiled with the inverted image of the bitmap.

## TMeterDirection Type

### Unit

### Feedback

### Declaration

```
TMeterDirection = (mdLeftToRight, mdRightToLeft, mdTopToBottom,  
    mdBottomToTop);
```

### Description

The TMeterDirection type is used by the Direction property to determine which way a TNeatoMeter object will indicate progress.

## TMeterStyle Type

### Unit

[Feedback](#)

### Declaration


```
TMeterStyle = (msStandard);
```

### Description

The TMeterStyle type is used by the [Style](#) property to determine the look of a [TNeatoMeter](#) object.

**Note:** Future versions will hopefully support more styles (segments, etc.).


## Properties

 Run-time only

 Key Properties

 BackColor

 BorderStyle

 Caption


 Completed

 Direction

 Font

 ForeColor


 ParentFont


 ParentShowHint


t

 Percent

 Shape

 ShowHint

 ShowPercent

 Total

 Visible

## Properties


 Key Properties

 HelpFile

 HintColor

 HintPause

 Icon

 Title



## TVisualApp Component

[Properties](#)

[Events](#)

### Unit

[VisApp](#)

### Description

TVisualApp is a descendent of TComponent. TVisualApp is a component to allow you to easily manipulate the global Application objects properties and attach event handlers.

## AnaClock Unit

The AnaClock unit contains the classes and types used to implement an analog clock component. The following items are declared in the AnaClock unit:

### Objects

TAnalogClock

### Types

TAnalogClockShape

## TPieShape Type

**Unit**

[Feedback](#)

**Declaration**

```
TPieShape = (psCircle, psEllipse);
```

**Description**

The TPieShape type is used by the [Shape](#) property to determine the shape of a [TPieMeter](#) object.



## TPieDirection Type

### Unit

### Feedback


### Declaration

```
TPieDirection = (pdClockwise, pdCounterClockwise);
```

### Description

The TPieDirection type is used by the Direction property to determine the direction a TPieMeter will indicate progress.

## Events

 Key Events

 OnActivate

 OnDeactivate


 OnException

 OnHelp

 OnHint

 OnIdle

 OnMessage

 OnMinimize

 OnRestore

 OnShowHint

## OnTimer Event

### Applies to

TAnalogClock, TLEDClock objects

### Declaration

**property** OnTimer: TNotifyEvent;

### Description

The OnTimer event is used to execute code at regular intervals. The Interval property of a TAnalogClock object determines how often this event occurs.

Note: The **Sender** parameter of the event will be a TAnalogClock or TLEDClock object not a TTimer.

## VisApp Unit

The VisApp unit contains the classes and types used to implement the visual application component.

The following items are declared in the VisApp unit:

### Objects

TVisualApp



## TAnalogClock Component

[Properties](#)

[Events](#)


### Unit

[AnaClock](#)

### Description

TAnalogClock is a descendent of TCustomControl. TAnalogClock is a component to display a standard analog clock. TAnalogClock can also be used as a timer by setting the [Interval](#) property and writing a handler for the [OnTimer](#) event.

## Properties

 Run-time only

 Key Properties

Align

BackColor

Enabled

FaceColor

HandsColor

Hint



Interval

OutlineColor

ParentShowHint

PopupMenu

SecHandColor



Shape

ShowHint



ShowSeconds

TickColor

Visible

## Events

### Key Events

OnClick

OnDbClick

OnDragDrop

OnDragOver

OnEndDrag

OnMouseDown

OnMouseMove

OnMouseUp

OnTimer



## TAnalogClockShape Type

### Unit

AnaClock

### Declaration

```
TAnalogClockShape = (csCircle, csSquare);
```

### Description

The TAnalogClockShape type is used by the Shape property to determine the displayed shape of a TAnalogClock component.



## FaceColor Property

### Applies to

TAnalogClock object

### Declaration

**property** FaceColor: TColor;

### Description

The FaceColor property determines the color of the face of the clock.

## HandsColor Property

### Applies to

TAnalogClock object

### Declaration

**property** HandsColor: TColor;

### Description

The HandsColor property determines the color of the minute and hour hands of the clock.

## Interval Property

### Applies to

TAnalogClock, TLEDClock objects

### Declaration

**property** Interval: Word;

### Description

The Interval property determines how often the clock will update the time display and generate OnTimer events.

## OutlineColor Property

### Applies to

TAnalogClock object

### Declaration

**property** OutlineColor: TColor;

### Description

The OutlineColor property determines the color of the clock border.

## SecHandColor Property

### Applies to

TAnalogClock object

### Declaration

**property** SecHandColor: TColor;

### Description

The SecHandColor property determines the color of the second hand of the clock.

## ShowSeconds Property

### Applies to

TAnalogClock, TLEDClock objects

### Declaration

**property** ShowSeconds: Boolean;

### Description

The ShowSeconds property determines the whether or not seconds will be displayed.

## TickColor Property

### Applies to

TAnalogClock object

### Declaration

**property** TickColor: TColor;

### Description

The TickColor property determines the color of the markers around the clock.



## TLEDLabel Component

[Properties](#)

[Events](#)

### Unit

[LEDGadgt](#)

### Description

TLEDClock is a descendent of TGraphicControl. TLEDLabel is a component to display a standard segmented LED readout.



## GetMaxWidthPixels Method

### Applies to

TBalloonHint object

### Declaration

```
function GetMaxWidthPixels: Integer;
```

### Description

The GetMaxWidthPixels method returns the actual maximum width in pixels of the balloon no matter what MaxWidth is set to.

## IndicatorColor Property

### Applies to

TRotaryKnob object

### Declaration

**property** IndicatorColor: TColor;

### Description

The IndicatorColor property determines the color used to paint the position indicator.

Note: The area around the knob is painted using the value of the Color property. The rest of the knob colors are determined by the current system colors.

## MMGadget Unit

The MMGadget unit contains the classes and types used to implement stereo-style rotary knobs.  
The following items are declared in the MMGadget unit:

### Objects

TRotaryKnob

## Method

GetMaxWidthPi  
xels



## TRotaryKnob Component

[Properties](#)

[Events](#)


### Unit


[MMGadget](#)

### Description


TRotaryKnob is a descendent of TCustomControl. TRotaryKnob is a component to allow users to select a value within a range using a familiar stereo-style rotary knob. TRotaryKnob can update the Caption or Text property of another control automatically using the Control property.

## Events

 Key Events

 OnChange

## Properties

 Run-time only

 Key Properties


Align


Color

 Control

Enabled

IndicatorColor

 Max

 Min

ParentShowHint  
t

PopupMenu

ShowHint

 Position

Visible

## Min Property

### Applies to

TRotaryKnob object

### Declaration

**property** Min: Integer;

### Description

The Min property along with the Max property determines the number of possible positions a knob can have.



## Max Property

### Applies to

TRotaryKnob object

### Declaration

**property** Max: Integer;

### Description

The Max property along with the Min property determines the number of possible positions a knob can have.

## Control Property

### Applies to

TRotaryKnob object

### Declaration

**property** Control: TControl;

### Description

When the Position changes, the knob control will automatically update the value of the Caption or Text property of the assigned control.

## OnChange Event

### Applies to

TRotaryKnob object

### Declaration

**property** OnChange: TNotifyEvent;

### Description

The OnChange is triggered whenever the Position of the knob changes.

## TSegmentSize Type

### Unit

LEDGadgt

### Declaration

```
TSegmentSize = 1..16;
```

### Description

The TSegmentSize type is used by the SegmentSize property to determine the thickness of LED segments.

## Events

### Key Events

OnClick

OnDbClick

OnDragDrop

OnDragOver

OnEndDrag

OnMouseDown

OnMouseMove

OnMouseUp



## TLEDClock Component

[Properties](#)

[Events](#)

### Unit

[LEDGadgt](#)

### Description

TLEDClock is a descendent of [TLEDLabel](#). TLEDClock is a component to display a standard LED clock. TLEDClock can also be used as a timer by setting the [Interval](#) property and writing a handler for the [OnTimer](#) event.

## Events

### Key Events

OnClick

OnDbClick

OnDragDrop

OnDragOver

OnEndDrag

OnMouseDown


OnMouseMove

OnMouseUp

OnTimer



## Properties

 Run-time only

 Key Properties

BackColor

Caption


Columns

 DrawMode

 DrawOnScreen

Enabled

Hint


 Interval

LitColor


ParentShowHint

PopupMenu

Rows

 SegmentSize

ShowHint


 ShowSeconds

UnlitColor

Visible




## Properties

 Run-time only


 Key Properties

 BackColor

Caption

 Columns

 DrawMode

 DrawOnScreen

Hint


LitColor

ParentShowHint

t

PopupMenu

 Rows

 SegmentSize

ShowHint

UnlitColor

Visible

## DrawMode Property

### Applies to

TLEDClock, TLEDLabel objects

### Declaration

**property** DrawMode: TLEDDrawMode;

### Description

The DrawMode property determines the method used to draw the LED segments.

Note: When SegmentSize is 1, the drawing mode will be dmLine regardless of the DrawMode setting.

## Columns Property

### Applies to

TLEDClock, TLEDLabel objects

### Declaration

**property** Columns: **Integer**;

### Description

The Columns property combined with the Rows property determine the number of LED characters in a LED display.

## TLEDDrawMode Type

**Unit**

LEDGadgt

**Declaration**

```
TLEDDrawMode = (dmPolygon, dmLine);
```

**Description**

The TLEDDrawMode type is used by the DrawMode property to determine what method will be used to draw LED segments.

## LEDGadgt Unit

The LEDGadgt unit contains the classes and types used to implement an LED label and clock components.

The following items are declared in the LEDGadgt unit:

### Objects

TLEDClock

TLEDLabel

### Types

## DrawOnScreen Property

### Applies to

TLEDClock, TLEDLabel objects

### Declaration

**property** DrawOnScreen: Boolean;

### Description

The DrawOnScreen property determines whether or not the LED Paint method will use an off-screen bitmap. Off-screen bitmaps result in less flicker - direct to screen may result in a more realistic LED.

## LitColor Property

### Applies to

TLEDClock, TLEDLabel objects

### Declaration

**property** LitColor: TColor;

### Description

The LitColor property determines the color of LED segments which should be "lit".

## Rows Property

### Applies to

TLEDClock, TLEDLabel objects

### Declaration

**property** Rows: **Integer**;

### Description

The Rows property combined with the Columns property determine the number of LED characters in an LED display.



## SegmentSize Property

### Applies to

TLEDClock, TLEDLabel objects

### Declaration

**property** SegmentSize: TSegmentSize;

### Description

The SegmentSize type is used to determine the thickness of LED segments.

## UnlitColor Property

### Applies to

TLEDClock, TLEDLabel objects

### Declaration

**property** UnlitColor: TColor;

### Description

The UnlitColor property determines the color of LED segments which should not be "lit".

Note: If UnlitColor and BackColor are the same, the unlit segments will not be drawn to improve performance.



