## DTools 3.0 Contents

Installation  Release Notes

# Components


TAnalogClock


TBalloonHint


TBWCCCheckBox


TBWCCRadioButton


TCustomHint


TDozer


TFComboBox


TFocusPanel


TFountainFill


TLEDCheckBox


TLEDClock


TLEDLabel


TLEDRadioButton


TNeatoMeter


TODButton


TODCheckBox


TODCycler


TODRadioButton


TPieMeter


TRotaryKnob


TShadowButton


TTiledBitmap


TVisualApp

# Routines

CreateFountainFillPalette
DrawFountainFill
GetColorStep
LeftStr
LTrim
Mid
RightStr
RTrim
ScaleColor
ScaleNum
ScaleRGB
TileBitmap
Trim

## Active Property

### Applies to

TBalloonHint, TCustomHint objects

### Declaration

```
property Active: Boolean;
```

### Description

The Active property determines if balloon/custom 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. The Shape property allows you to change between plain and rounded rectangles. ShadowDepth and ShadowStyle allow you to control the appearance of the drop shadow. The Style property allows you to configure the colors and font. The Operation property allows you to tailor the way the balloons are activated and displayed.

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

**Known Problem**

Setting the Style property to hsSystem produces unexpected results.

**Properties**

- Run-time only
- Key Properties

| | | |
|---|---|---|
| Active | MaxWidth | ShadowStyle |
| BorderColor | Operation | Shape |
| Color | Position | Style |
| Font | ShadowDepth | |

## MaxWidth Property

### Applies to

TBalloonHint , TCustomHint objects

### Declaration

```
property MaxWidth: Integer;
```

### Description

The MaxWidth property determines the maximum width in pixels the hint window will occupy on the screen. If this value is less than zero, the hint window 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, TLEDCheckBox, TLEDRadioButton, TPieMeter objects

**Declaration**

TAnalogClock:

```
property Shape: TAnalogClockShape;
```

TBalloonHint:

```
property Shape: TBalloonShape;
```

TLEDCheckBox, TLEDRadioButton:

```
property Shape: TLEDShape;
```

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:

**Components**

TBalloonHint

**Types**

TShadowDepth
TBalloonBehaviors
TBalloonPosition
TBalloonShape
TBalloonShadowStyle

# 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:

**Components**

TNeatoMeter
TPieMeter

**Types**

TBevelDepth
TBevelType
TBitmapDrawStyle
TMeterDirection
TMeterStyle
TPieDirection
TPieShape

**Properties**

- Run-time only
- Key Properties

| | | | | | |
|---|---|---|---|---|---|
| | BackColor | | Completed | | ShowHint |
| | BevelDepth | | Direction | | ShowPercent |
| | BevelType | | Font | | Style |
| | Bitmap | | ForeColor | | Total |
| | BitmapDrawStyle | | ParenFont | | UseFontColor |
| | BorderStyle | | ParentShowHint | | Visible |
| | Caption | | Percent | | |

## 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, TShadowButton, TTiledBitmap objects

### Declaration

```
property Bitmap: TBitmap;
```

### Description

TNeatoMeter

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.

TShadowButton, TTiledBitmap

The Bitmap property is the bitmap to tile.

## 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

TBalloonHint, TCustomHint, TFountainFill, TNeatoMeter objects

### Declaration

TBalloonHint and TCustomHint

```
property Style: THintStyle;
```

TFountainFill

```
property Style: TFountainStyle;
```

TNeatoMeter

```
property Style: TMeterStyle;
```

### Description

TBalloonHint and TCustomHint

The Style property determines the font and colors used to display hints.

TFountainFill

The Style property determines the fill pattern.

TNeatoMeter

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

| | | |
|---|---|---|
| Align | FromColor | Steps |
| DragCursor | Hint | Style |
| DragMode | ParentShowHint | ToColor |
| DrawOnScreen | PopupMenu | Visible |
| Enabled | ShowHint | |

**Properties**

- Run-time only
- Key Properties

| | | | | | |
|---|---|---|---|---|---|
| | BackColor | | Font | | Shape |
| | BorderStyle | | ForeColor | | ShowHint |
| | Caption | | ParenFont | | ShowPercent |
| | Completed | | ParentShowHint | | Total |
| | Direction | | Percent | | Visible |

**Properties**

32 32-bit only

Key Properties

| | | |
|---|---|---|
| HelpFile | HintPause | 32 ShowMainForm |
| HintColor | 32 HintShortPause | Title |
| 32 HintHidePause | Icon | 32 UpdateFormatSettings |

# 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:

**Components**

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, TDozer, TLEDClock objects

### Declaration

```
property OnTimer: TNotifyEvent;
```

### Description

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

**Note**: The **Sender** parameter of the event will be a TAnalogClock, TDozer 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:

**Components**

TVisualApp

xxxx

# 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 | | Interval | | | ShowHint |
| BackColor | | OutlineColor | | | ShowSeconds |
| Enabled | | ParentShowHint | | | TickColor |
| FaceColor | | PopupMenu | | | Visible |
| HandsColor | | SecHandColor | | | |
| Hint | | Shape | | | |

**Events**

Key Events

| | | |
|---|---|---|
| OnClick | OnDragOver | OnMouseMove |
| OnDblClick | OnEndDrag | OnMouseUp |
| OnDragDrop | OnMouseDown | 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, TShadowButton object

**Declaration**

```
property FaceColor: TColor;
```

**Description**

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

**Events**

Key Events

OnTimer OnWakeUp

## 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, TDozer, TLEDClock objects

### Declaration

```
property Interval: Word;
```

### Description

The Interval property determines how often clocks will be updated and OnTimer events will be fired.

## 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, TCustomHint objects

### 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:

**Components**

TRotaryKnob

**Method**

GetMaxWidth
Pixels

# 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 | IndicatorColor | PopupMenu |
| Color | Max | ShowHint |
| Control | Min | Position |
| Enabled | ParentShowHint | Visible |

## Min Property

### Applies to

TODCycler, TRotaryKnob object

### Declaration

```
property Min: Integer;
```

### Description

The Min property along with the Max property determines the range of possible values a knob or cycler button can have.

## Max Property

### Applies to

TODCycler, TRotaryKnob object

### Declaration

```
property Max: Integer;
```

### Description

The Max property along with the Min property determines the range of possible values a knob or cycler button 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 event 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 | OnDragOver | OnMouseMove |
| OnDblClick | OnEndDrag | OnMouseUp |
| OnDragDrop | OnMouseDown | |

# TLEDClock Component

Properties  Events

**Unit**

LEDGadgt

**Description**

TLEDClock is a descendent of TGraphicControl.  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 | OnDragOver | OnMouseMove |
| OnDblClick | OnEndDrag | OnMouseUp |
| OnDragDrop | OnMouseDown | OnTimer |

**Properties**

- Run-time only
- Key Properties

| | | | | | |
|---|---|---|---|---|---|
| | BackColor | | Hint | | SegmentSize |
| | Caption | | Interval | | ShowHint |
| | Columns | | LitColor | | ShowSeconds |
| | DrawMode | | ParentShowHint | | UnlitColor |
| | DrawOnScreen | | PopupMenu | | Visible |
| | Enabled | | Rows | | |

**Properties**

- Run-time only
- Key Properties

| | | | | | |
|---|---|---|---|---|---|
| | BackColor | | Hint | | SegmentSize |
| | Caption | | LitColor | | ShowHint |
| | Columns | | ParentShowHint | | UnlitColor |
| | DrawMode | | PopupMenu | | Visible |
| | DrawOnScreen | | Rows | | |

## 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

TLEDLabel object

### 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.

## DTUtil Unit

The DTUtil unit contains utility routines.

The following items are declared in the DTUtil unit:

**Types**

TFountainStyle
THintStyle

**Routines**

GetColorStep
CreateFountainFillPalette
DrawFountainFill
LeftStr
LTrim
Mid
RightStr
RTrim
ScaleColor
ScaleNum
ScaleRGB
TileBitmap
Trim

## LEDGadgt Unit

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

The following items are declared in the LEDGadgt unit:

**Components**

TLEDClock

TLEDLabel

**Types**

TLEDDrawMode

TSegmentSize

## DrawOnScreen Property

### Applies to

TFountainFill, TLEDClock, TLEDLabel objects

### Declaration

```
property DrawOnScreen: Boolean;
```

### Description

The DrawOnScreen property determines whether or not the Paint method will use an off-screen bitmap.  Off-screen bitmaps result in less flicker.

## LitColor Property

**Applies to**

TLEDCheckBox, TLEDClock, TLEDLabel, TLEDRadioButton objects

**Declaration**

```
property LitColor: TColor;
```

**Description**

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

## Rows Property

### Applies to

TLEDLabel object

### 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

TLEDCheckBox, TLEDClock, TLEDLabel, TLEDRadioButton objects

### Declaration

```
property UnlitColor: TColor;
```

### Description

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

Note: For TLEDClock and TLEDLabel, if UnlitColor and BackColor are the same, the unlit segments will not be drawn to improve performance.

## Release Notes

**3.0**

- Added support for Delpi 2.0 32-bit!
- Added Behavior property to TBalloonHint to replace the Operation property. If you load an old project, you might receive the following message: "Error reading Object.Operation: Property does not exist. Ignore the error and continue?". Choose Ignore and everything should perform as expected. You will also need to save the form. Thanks to Nick Naimo for bringing discrepencies between the 1.0 and 2.0 balloon hint performance.
- Added HintHidePause, HintShortPause, ShowMainForm, and UpdateFormatSettings properties to TVisualApp to support new TApplication properties available in Delphi 2.0.

**2.1**

- Fixed TBWCCCheckbox and TBWCCRadioButton Font property. The Font property was not being set properly during the Paint method. Thanks to Scott Lovy for reporting this bug.
- Made sure that all DTools .DCU files were compiled without debug information. This was causing some of you a few problems - sorry!

**2.0**

- Thanks to all of you for the encouragement, ideas and support!
- Source code for DTools is now available! See ORDER.TXT for details.
- As soon as I get a copy of the 32-bit Delphi, I will update and release a new version of DTools.
- Added several new features to TBalloonHint (including Windows '95 features)
- Added Routines section to this help file. Included are: CreateFountainFillPalette, DrawFountainFill, GetColorStep, LeftStr, LTrim, Mid, RightStr, RTrim, ScaleColor, ScaleNum, ScaleRGB, TileBitmap, Trim.
- Added the following components: TCustomHintWindow, TBWCCCheckBox, TBWCCRadioButton, TDozer, TFComboBox, TFountainFill, TLEDCheckBox, TLEDRadioButton, TODButton, TODCheckBox, TODCycler, TODRadioButton, TShadowButton, TTiledBitmap.
- Fixed TLEDLabel and TLEDClock SegmentSize setting of 3. The default value in the class declaration was incorrect. Thanks to Stephen Ibbs for reporting this bug.
- A special thanks goes out to Ritchey Consulting Services for there most excellent K2B RTF File Preprocessor.

# TODCheckBox Component

Properties  Events

**Unit**

Toggler

**Description**

TODCheckBox is a descendent of TCustomControl.  TODCheckBox is an owner-draw button which behaves like a check box.

# TODCycler Component

Properties  Events

**Unit**

Toggler

**Description**

TODCycler is a descendent of TCustomControl.  TODCycler is an owner-draw button which cycles its values on each click.

# TODRadioButton Component

Properties  Events

**Unit**

Toggler

**Description**

TODRadioButton is a descendent of TCustomControl.  TODRadioButton is an owner-draw button which behaves like a radio button.

## Toggler Unit

The Toggler unit contains the classes and types used to implement custom toggle style buttons.

The following items are declared in the Toggler unit:

**Components**

TBWCCCheckBox

TBWCCRadioButton

TLEDCheckBox

TLEDRadioButton

TODCheckBox

TODCycler

TODRadioButton

**Types**

TLEDShape

xxxx

# TODButton Component

Properties Events

**Unit**

CustBtn

**Description**

TODButton is a descendent of TButton. TODButton is an owner-draw button which behaves like a standard command button.

# TCustomHint Component

Properties  Method

### Unit

DTMisc

### Description

TCustomHint is a descendent of TComponent.  TCustomHint is a replacement for the built in 'tool-tip' style hints.  TCustomHint provides control over the font, color and maximum width of the hint window.

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

### Known Problem

Setting the Style property to hsSystem produces unexpected results.

## TileBitmap Procedure

**Unit**

DTUtil

**Declaration**

```
procedure TileBitmap(Canvas: TCanvas; Bitmap: TBitmap; Bounds: TRect);
```

**Remarks**

Draws a bitmap in a tiled fashion on a canvas.

| Parameter | Description |
|---|---|
| Canvas | Canvas to draw on |
| Bitmap | Bitmap to tile |
| Bounds | Bounds rectangle to draw in |

**Properties**

Run-time only

Key Properties

| | | |
|---|---|---|
| Caption | EraseBackground | PopupMenu |
| Canvas | Font | ShowHint |
| Color | IsDown | TabOrder |
| Ctl3D | ParentColor | TabStop |
| DragCursor | ParentCtl3D | Visible |
| DragMode | ParentFont | |
| Enabled | ParentShowHint | |

## TBalloonShadowStyle Type

### Unit

Balloon

### Declaration

```
TBalloonShadowStyle = (ssShaded, ssSolid);
```

### Description

The TBalloonShadowStyle type is used by the ShadowStyle property of the TBalloonHint component to determine if balloon shadows will appear solid or shaded.

**Events**

Key Events

| | | | |
|---|---|---|---|
| OnClick | OnExit | OnMouseMove | |
| OnDragDrop | OnKeyDown | OnMouseUp | |
| OnDragOver | OnKeyPress | OnPaint | |
| OnEndDrag | OnKeyUp | | |
| OnEnter | OnMouseDown | | |

## CustBtn Unit

The CustBtn unit contains the classes and types used to implement custom and owner-draw buttons.

The following items are declared in the CustBtn unit:

**Components**

TODButton

TShadowButton

**Events**

Key Events

| | | |
|---|---|---|
| OnClick | OnExit | OnMouseMove |
| OnDragDrop | OnKeyDown | OnMouseUp |
| OnDragOver | OnKeyPress | |
| OnEndDrag | OnKeyUp | |
| OnEnter | OnMouseDown | |

## ScaleNum Function

**Unit**

DTUtil

**Declaration**

```
function ScaleNum(value, limit, percent: Integer): Integer;
```

**Remarks**

Returns a number scaled by *percent*.

| Parameter | Description |
|---|---|
| value | value to scale |
| limit | upper/lower bound of value |
| percent | percentage to scale by - if limit is less than value then the result will also be less than value, otherwise the result will be greater than value. |

## ScaleColor Function

### Unit

DTUtil

### Declaration

```
function ScaleColor(color: TColor; HowMuch: Integer): TColor;
```

### Remarks

Returns a lightened/darkened RGB color.  This function can accept system color values as well as RGB colors.

| Parameter | Description |
|---|---|
| color | color to scale |
| HowMuch | percentage to scale by - positive values produce lighter colors, negative values produce darker colors |

## ScaleRGB Function

### Unit

DTUtil

### Declaration

```
function ScaleRGB(color: TColor; HowMuch: Integer): TColor;
```

### Remarks

Returns a lightened/darkened RGB color. This function will not work properly with system color values - use ScaleColor instead.

| Parameter | Description |
|---|---|
| color | color to scale |
| HowMuch | percentage to scale by - positive values produce lighter colors, negative values produce darker colors |

## DTMisc Unit

The DTMisc unit contains the classes and types used to implement miscelaneous components.

The following items are declared in the DTMisc unit:

**Components**

TCustomHint

TDozer

TFountainFill

TFComboBox

TFocusPanel

TTiledBitmap

# TBWCCRadioButton Component

Properties  Events

**Unit**

Toggler

**Description**

TBWCCRadioButton is a descendent of TCustomControl.  TBWCCRadioButton functions as a radio button which looks like the ones found in BWCC.DLL.

# TLEDCheckBox Component

Properties Events

**Unit**

Toggler

**Description**

TLEDCheckBox is a descendent of TCustomControl. TLEDCheckBox functions as a check box which displays an LED style 'button'.

# TLEDRadioButton Component

Properties  Events

**Unit**

Toggler

**Description**

TLEDRadioButton is a descendent of TCustomControl.  TLEDRadioButton functions as a radio button which displays an LED style 'button'.

# TBWCCCheckBox Component

Properties  Events

**Unit**

Toggler

**Description**

TBWCCCheckBox is a descendent of TCustomControl.  TBWCCCheckBox functions as a check box which looks like the ones found in BWCC.DLL.

## TFountainStyle Type

### Unit

DTUtil

### Declaration

```
TFountainStyle = (fsHorizontal, fsVertical, fsCircular);
```

### Description

The TFountainStyle type is used by to determine the drawing method for the DrawFountainFill procedure and the **Style** property of a TFountainFill object. The following table describes the meaning of each value:

| Value | Meaning |
|---|---|
| fsHorizontal | The gradient will be draw using horizontal rectangles. |
| fsVertical | The gradient will be draw using vertical rectangles. |
| fsCircular | The gradient will be draw using rings. |

# TFountainFill Component

Properties Events

**Unit**

DTMisc

**Description**

TFountainFill is a descendent of TGraphicControl. TFountainFill is a component to display color gradients similar to those found in illustration packages.

# TFocusPanel Component

TPanel reference

### Unit

DTMisc

### Description

TFocusPanel is a descendent of TPanel.  Unlike TPanel, TFocusPanel will accept the input focus and publishes the OnKeyDown, OnKeyPress and OnKeyUp events.

**Properties**

- Run-time only
- Key Properties

| | | |
|---|---|---|
| Active | Font | Style |
| Color | MaxWidth | |

## OnPaint Event

**Applies to**

TODButton, TODCheckBox, TODCycler, TODRadioButton objects

**Declaration**

```
property OnPaint: TNotifyEvent;
```

**Description**

The OnPaint event is triggered whenever the control needs to be repainted.

## Operation Property

This property has been replaced by the Behavior property.

**Note**: If you load an old project, you might receive the following message: "Error reading Object.Operation: Property does not exist. Ignore the error and continue?". Choose Ignore and everything should perform as expected. You will also need to save the form.

## Behavior Property

### Applies to

TBalloonHint object

### Declaration

```
property Behavior: TBalloonBehaviors;
```

### Description

The Behavior property determines how and when the balloon will be displayed. Behavior replaces the Operation property.

**Note**: If you load an old project, you might receive the following message: "Error reading Object.Operation: Property does not exist. Ignore the error and continue?". Choose Ignore and everything should perform as expected. You will also need to save the form.

**Properties**

- Run-time only
- Key Properties

| | | |
|---|---|---|
| Caption | GroupIndex | Shape |
| Checked | IgnoreEnabled | ShowHint |
| Color | LitColor | TabOrder |
| DragCursor | ParentColor | TabStop |
| DragMode | ParentFont | UnlitColor |
| Enabled | ParentShowHint | Visible |
| Font | PopupMenu | |

**Properties**

- Run-time only
- Key Properties

| | | | | | |
|---|---|---|---|---|---|
| | Caption | | IgnoreEnabled | | ShowHint |
| | Checked | | LitColor | | State |
| | Color | | ParentColor | | TabOrder |
| | DragCursor | | ParentFont | | TabStop |
| | DragMode | | ParentShowHint | | UnlitColor |
| | Enabled | | PopupMenu | | Visible |
| | Font | | Shape | | |

## ShadowStyle Property

### Applies to

TBalloonHint object

### Declaration

```
property ShadowStyle: TBalloonShadowStyle;
```

### Description

The ShadowStyle property determines the type of shading used under the balloon.

**Note**: Setting ShadowStyle to ssSolid and setting Operation to boUseRegions is nice for Windows '95.

## GroupIndex Property

**Applies to**

TBWCCRadioButton, TLEDRadioButton, TODRadioButton objects

**Declaration**

```
property GroupIndex: Integer;
```

**Description**

The GroupIndex property determines which radio buttons work together as a group.

## EraseBackground Property

### Applies to

TODButton, TODCheckBox, TODCycler, TODRadioButton objects

### Declaration

```
property EraseBackground: Boolean;
```

### Description

The EraseBackground property determines whether the background of the control will be erased prior to painting.  If your control paints the entire client area then EraseBackground should be False, otherwise EraseBackground should be True.

## IsDown Property

**Applies to**

TODButton, TODCheckBox, TODCycler, TODRadioButton objects

**Declaration**

```
property IsDown: Boolean;
```

**Description**

The IsDown property determines whether the control should be drawn as pressed.

## Step Property

### Applies to

TODCycler object

### Declaration

```
property Step: Integer;
```

### Description

The Step property determines how much the Value property is incremented for each click. The range of possible values can be limited by the Min and Max properties.

## TLEDShape Type

### Unit

Feedback

### Declaration

```
TLEDShape = (shCircle, shSquare);
```

### Description

The TLEDShape type is used by the Shape property to determine the shape of TLEDCheckBox and TLEDRadioButton indicators.

## Value Property

### Applies to

TODCycler object

### Declaration

```
property Value: Integer;
```

### Description

The Value property holds the current value of a TODCycler object. The range of possible values can be limited by the Min and Max properties.

# TDozer Component

Properties Methods Events

**Unit**

DTMisc

**Description**

TDozer is a descendent of TComponent. TDozer is used to put your program to 'sleep' for the time specified by DozeLength. TDozer will also generate OnTimer events (depending on the Interval setting) for the duration of the nap.

## DozeFor Method

### Applies to

TDozer object

### Declaration

```
procedure DozeFor(ADozeLength: Word);
```

### Description

The DozeFor method sets DozeLength to ADozeLength. and then calls Doze.

## Doze Method

**Applies to**

TDozer object

**Declaration**

```
procedure Doze;
```

**Description**

The Doze method puts the program to 'sleep' for the time specified by DozeLength.

**See Also**

DozeFor

**Methods**

Key Methods

Doze DozeFor

## DozeLength Property

### Applies to

TDozer object

### Declaration

```
property DozeLength: Word;
```

### Description

The DozeLength property determines how long the program should 'sleep'. When this time has expired the OnWakeUp event will be fired.

**Properties**

- Run-time only
- Key Properties
  - DozeLength
  - Interval

## OnWakeUp Event

### Applies to

TDozer object

### Declaration

```
property OnWakeUp: TNotifyEvent;
```

### Description

The OnWakeUp event is used to execute code when the time specified by DozeLength has expired.

**Properties**

- Run-time only
- Key Properties

| | | | |
|---|---|---|---|
| BorderColor | Font | ShowHint |
| Caption | ParentColor | TabOrder |
| Color | ParentFont | TabStop |
| DragCursor | ParentShowHint | TextAlignment |
| DragMode | PopupMenu | Visible |
| Enabled | ShadowColor | |
| FaceColor | ShadowOffset | |

## LTrim Function

**Unit**

DTUtil

**Declaration**

```
function LTrim(s: string): string;
```

**Remarks**

Returns a string with the leading spaces removed.

| Parameter | Description |
|---|---|
| s | string from which the leading spaces will be removed |

**Note**: Under Delphi 2.0, this function is implemented as a simple wrapper for the Delphi TrimLeft function.

# TShadowButton Component

Properties Events

**Unit**

CustBtn

**Description**

TShadowButton is a descendent of TButton. TShadowButton is a shadowed button which behaves like a standard command button.

## DrawFountainFill Procedure

### Unit

DTUtil

### Declaration

```
procedure DrawFountainFill(Canvas: TCanvas; FromColor, ToColor: TColor;
  Steps: Integer; Style: TFountainStyle; Height, Width: Integer; Palette:
  HPalette; DrawOnCanvas: Boolean);
```

### Remarks

This procedure draws a gradient on the specified canvas.

| Parameter | Description |
|---|---|
| Canvas | Canvas to draw on |
| FromColor | Starting color for gradient |
| ToColor | Ending color for gradient |
| Steps | The number of colors from FromColor to ToColor |
| Style | Determines the appearance of the gradient - TFountainStyle |
| Height | Height of the area to fill |
| Width | Width of the area to fill |
| Palette | Palette to use - CreateFountainFillPalette can be used to create a suitable palette |
| DrawOnCanvas | If this parameter is True then drawing will occur directly on Canvas, otherwise drawing will occur on a temporary bitmap which will be transferred to Canvas when the gradient is complete. |

## LeftStr Function

### Unit

DTUtil

### Declaration

```
function LeftStr(s: string; cnt: Integer): string;
```

### Remarks

Returns the leftmost *cnt* characters of *s*.

| Parameter | Description |
|---|---|
| s | string from which the leftmost characters are returned |
| cnt | number of characters to return |

## BorderColor Property

### Applies to

TBalloonHint, TShadowButton objects

### Declaration

```
property BorderColor: TColor
```

### Description

The BorderColor property determines the color of the border around the object.

## ShadowOffset Property

**Applies to**

TShadowButton object

**Declaration**

```
property ShadowOffset: Integer;
```

**Description**

The ShadowOffset property determines the number of pixels to offset the button shadow.

## ShadowColor Property

### Applies to

TShadowButton object

### Declaration

```
property ShadowColor: TColor
```

### Description

The ShadowColor property determines the color of the button shadow.

## TBalloonBehaviors Type

### Unit

Balloon

### Declaration

```
TBalloonBehavior = (bbNoShowMouseDown, bbUseRegions, bbHideOnPaint);
TBalloonBehaviors = set of TBalloonBehavior;
```

### Description

The TBalloonBehaviors type is used by the Behavior property of the TBalloonHint component to determine the behavior of the balloon. The following table describes the meaning of each value:

| Value | Meaning |
|---|---|
| bbNoShowMouseDown | The balloon will not display if any mouse button is down. This helps to reduce the amount of screen flicker that occurs. |
| bbUseRegions | The balloon will take advantage of region windows when running under Win32. The balloon will behave just like the rectangular tooltip style hints. When using this setting it is best to set ShadowStyle to ssSolid. *Since Window NT does not currently export a 16-bit entry point for SetWindowRgn this value will be ignored in 16-bit applications running under Windows NT.* |
| bbHideOnPaint | The balloon will hide when a WM_PAINT message is sent to the application which includes an update rectangle which is under the balloon window. *This value is ignored when bbUseRegions is set.* |

## CreateFountainFillPalette Function

### Unit

DTUtil

### Declaration

```
function CreateFountainFillPalette(FromColor, ToColor: TColor; Steps:
  Integer): HPalette;
```

### Remarks

This function will return a Windows palette handle (HPALETTE) filled with the range of colors specified. Use this function in conjunction with DrawFountainFill to create nice gradient backgrounds. If the palette could not be created the return value will be zero.

| Parameter | Description |
|---|---|
| FromColor | Starting color for gradient |
| ToColor | Ending color for gradient |
| Steps | The number of colors from FromColor to ToColor |

## GetColorStep Function

### Unit

DTUtil

### Declaration

```
function GetColorStep(FromColor, ToColor: TColor; Steps, Step: Integer):
  TColor;
```

### Remarks

This function will return an RGB color value for the specified step within the range of colors.

| Parameter | Description |
|---|---|
| FromColor | Starting color for gradient |
| ToColor | Ending color for gradient |
| Steps | The number of colors from FromColor to ToColor |
| Step | Item within the range to return |

## Steps Property

### Applies to

TFountainFill object

### Declaration

```
property Steps: Integer;
```

### Description

The Steps property determines how many colors will be drawn between FromColor and ToColor.

## FromColor Property

### Applies to

TFountainFill object

### Declaration

```
property FromColor: TColor;
```

### Description

The FromColor property determines the starting color of a fountain fill.

## ToColor Property

### Applies to

TFountainFill object

### Declaration

```
property ToColor: TColor;
```

### Description

The ToColor property determines the ending color of a fountain fill.

## UsePalette Property

### Applies to

TFountainFill object

### Declaration

```
property UsePalette: Boolean;
```

### Description

The UsePalette property determines if a fountain fill will use an optimized palette.

## TextAlignment Property

**Applies to**

TShadowButton object

**Declaration**

```
property TextAlignment: TAlignment;
```

**Description**

The TextAlignment property determines the justification of text on the button.

# TFComboBox Component

TComboBox reference

### Unit

DTMisc

### Description

TFComboBox is a descendent of TComboBox. Under Windows '95, the menu you specify for the PopupMenu property will not display if your combo box has an edit box while the mouse is over the edit box. TFComboBox fixes this problem - it does not add other properties or events to TComboBox.

## Mid Function

### Unit

DTUtil

### Declaration

```
function Mid(s: string; idx, cnt: Integer): string;
```

### Remarks

Returns a substring of a string. This function is identical in function to the built in Copy function.

| Parameter | Description |
|---|---|
| s | string from which the substring will be extracted |
| idx | starting point |
| cnt | number of characters to return |

## RightStr Function

**Unit**

DTUtil

**Declaration**

```
function RightStr(s: string; cnt: Integer): string;
```

**Remarks**

Returns the rightmost *cnt* characters of *s*.

| Parameter | Description |
|---|---|
| s | string from which the rightmost characters are returned |
| cnt | number of characters to return |

## RTrim Function

### Unit

DTUtil

### Declaration

```
function RTrim(s: string): string;
```

### Remarks

Returns a string with the trailing spaces removed.

| Parameter | Description |
|---|---|
| s | string from which the trailing spaces will be removed |

**Note**: Under Delphi 2.0, this function is implemented as a simple wrapper for the Delphi TrimRight function.

## Trim Function

**Unit**

DTUtil

**Declaration**

```
function Trim(s: string): string;
```

**Remarks**

Returns a string with the leading and trailing spaces removed.

| Parameter | Description |
|---|---|
| s | string from which the leading and trailing spaces will be removed |

**Note**: Since Delphi 2.0 contains a Trim function, this function is not included in the 32-bit version of DTools.

## IgnoreEnabled Property

### Applies to

TLEDCheckBox, TLEDRadioButton objects

### Declaration

```
property IgnoreEnabled: Boolean;
```

### Description

The IgnoreEnabled property determines whether or not the component will take the Enabled property into account when drawing the control. Normally the control would 'dim' the text color to indicate to the user that the component is disabled. If IgnoreEnabled is True, the control will be drawn as if it were enabled (text will be the normal color). The IgnoreEnabled property allows the component to be used as a status indicator without allowing the user to change values using the mouse or keyboard.

# TTiledBitmap Component

Properties  Events

**Unit**

DTMisc

**Description**

TTiledBitmap is a descendent of TGraphicControl.  TTiledBitmap is a component to display tiled bitmaps.

**Properties**

Run-time only

Key Properties

| | | |
|---|---|---|
| Align | Enabled | ShowHint |
| DragCursor | Hint | Visible |
| DragMode | ParentShowHint | |
| Bitmap | PopupMenu | |

## RedrawOnUpDown Property

### Applies to

TODCheckBox, TODCycler, TODRadioButton objects

### Declaration

```
property RedrawOnUpDown: Boolean;
```

### Description

The RedrawOnUpDown property determines whether the control will generate OnPaint events on an 'up/down' state change. The 'up/down' state change occurs when the user is moving the mouse on and off the control with the primary mouse button down.

## THintStyle Type

### Unit

DTUtil

### Declaration

```
THintStyle = (hsDefault, hsCustom, hsSystem);
```

### Description

The THintStyle type is used by to determine which font and colors to use when displaying hints. The following table describes the meaning of each value:

| Value | Meaning |
|---|---|
| hsDefault | The hint will use the default Delphi settings of Black 8 point MS Sans Serif font and Application.HintColor for the hint background. |
| hsCustom | The hint will use the values specified in the Color and Font properties. |
| hsSystem | The hint will use the values stored in the system registry for Windows '95 (hsDefault when running under other than Windows '95). |

## DTools Installation

### Component Installation

- Delphi 2.0: Choose Install from the Components menu
  Delphi 1.0: Choose Install Components from the Options menu
- Choose Add
- Choose Browse
- Select DTOOLS.PAS from the directory you put DTOOLS in
- Choose OK and wait
- There will now be a DTools page on your component palette

### Help File Installation

**Note:** There are two help files referenced in this section - DTOOLS32 and DTOOLS. The only difference between the files are the links to Delphi help topics. Delphi 1.0 has VCL references in DELPHI.HLP while Delphi 2.0 has VCL references in VCL.HLP. There is no difference in the content of these files.

- Make sure Delphi is NOT running
- Put the DTools help files where Delphi can find them. This can be anywhere along your PATH or:

  Delphi 2.0: c:\Program Files\Borland\Delphi 2.0\Bin

  Copy DTOOLS32.HLP and DTOOLS32.CNT

  Delphi 1.0: c:\delphi\bin

  Copy DTOOLS.HLP and DTOOLS.CNT
- Run the HelpInst application that comes with Delphi

  Delphi 2.0: c:\Program Files\Borland\Delphi 2.0\Help\Tools\Helpinst.exe

  Delphi 1.0: c:\delphi\help\Helpinst.exe
- Open DELPHI.HDX

  Delphi 2.0: C:\Program Files\Borland\Delphi 2.0\BIN\delphi.hdx

  Since HelpInst doesn't support long filenames you will probably need to select something like:

  c:\progra~1\borland\delphi~1.0\bin\delphi.hdx

  Delphi 1.0: c:\delphi\bin\delphi.hdx
- Set the search paths

  Delphi 2.0: Select Search paths from the Options menu. You need to specify the directory where the .KWF files can be found. This is usually:

  C:\Program Files\Borland\Delphi 2.0\Help

  For a default installation you can just type in ..\ (dot dot backslash) for the path.

  Otherwise, since HelpInst doesn't support long filenames you will probably need to enter something like:

  c:\progra~1\borland\delphi~1.0\help

  **Note:** To avoid this step in the future, you can simply move or copy Helpinst.exe into the Delphi HELP directory where the .KWF files

reside.

Delphi 1.0: This step should be unnecessary for a normal Delphi 1.0 installation.

- Choose Add Keyword File from the Keywords menu
- Select the DTools keyword file:

  Delphi 2.0: DTOOLS32.KWF

  Delphi 1.0: DTOOLS.KWF

  Putting the keyword file in the Delphi HELP directory will make your life easier if you need to run HelpInst in the future.
- Choose Save from the File menu and wait

**Note:** All directory references assume you installed Delphi using the setup defaults. If you installed Delphi using different directories, you will need to adjust these instructions to match your installation.

**Properties**

 Run-time only

 Key Properties

| | | |
|---|---|---|
| Caption | Font | TabOrder |
| CheckColor | ParentColor | TabStop |
| Checked | ParentFont | Visible |
| Color | ParentShowHint | |
| DragCursor | PopupMenu | |
| DragMode | ShowHint | |
| Enabled | State | |

**Properties**

 Run-time only

 Key Properties

| | | | |
|---|---|---|---|
| | Caption | Font | TabOrder |
|  | CheckColor |  GroupIndex | TabStop |
|  | Checked | ParentColor | Visible |
| | Color | ParentFont | |
| | DragCursor | ParentShowHint | |
| | DragMode | PopupMenu | |
| | Enabled | ShowHint | |

**Properties**

 Run-time only

 Key Properties

| | | |
|---|---|---|
| Caption | EraseBackground | RedrawOnUpDown |
| Canvas | Font | ShowHint |
| Checked | IsDown | State |
| Color | ParentColor | TabOrder |
| Ctl3D | ParentCtl3D | TabStop |
| DragCursor | ParentFont | Visible |
| DragMode | ParentShowHint | |
| Enabled | PopupMenu | |

## CheckColor Property

### Applies to

TBWCCCheckBox, TBWCCRadioButton objects

### Declaration

```
property CheckColor: TColor;
```

### Description

The CheckColor property determines the color of the 'checked' indicator for BWCC-style radio buttons and checkboxes.

**Properties**


Run-time only


Key Properties

Caption


EraseBackground

PopupMenu


Canvas

Font


RedrawOnUpDown


Checked


GroupIndex

ShowHint

Color


IsDown

TabOrder

Ctl3D

ParentColor

TabStop

DragCursor

ParentCtl3D

Visible

DragMode

ParentFont

Enabled

ParentShowHint

**Properties**

 Run-time only

 Key Properties

Caption

 Canvas

Color

Ctl3D

DragCursor

DragMode

Enabled

 EraseBackground

 Font

 IsDown

 Max

 Min

ParentColor

ParentCtl3D

ParentFont

ParentShowHint

PopupMenu

 RedrawOnUpDown

ShowHint

 Step

TabOrder

TabStop

 Value

Visible