JDLib Documentation

Table of Contents

[Overview 2](#_Toc157446991)

[JDLibComponents.bpl 2](#_Toc157446992)

[Contains 2](#_Toc157446993)

[Requires 2](#_Toc157446994)

[dclJDLibComponents.pbl 2](#_Toc157446995)

[Contains 2](#_Toc157446996)

[Requires 3](#_Toc157446997)

[Global TODO 3](#_Toc157446998)

[JD.Common.pas 4](#_Toc157446999)

[const WM\_JD\_COLORCHANGE 4](#_Toc157447000)

[const SC\_DRAGMOVE 4](#_Toc157447001)

[class EJDException 4](#_Toc157447002)

[class EJDOutOfRange 4](#_Toc157447003)

[type PJDPoint 4](#_Toc157447004)

[type TJDPoint 4](#_Toc157447005)

[type PJDRect 4](#_Toc157447006)

[type TJDRect 5](#_Toc157447007)

[class TJDComponent 5](#_Toc157447008)

[class TJDMessageComponent 5](#_Toc157447009)

[class TJDCustomControl 6](#_Toc157447010)

[function IntRange 6](#_Toc157447011)

[JD.Ctrls.pas 7](#_Toc157447012)

[class TJDControl 7](#_Toc157447013)

[JD.Graphics.pas 8](#_Toc157447014)

[Constants 8](#_Toc157447015)

[type TJDStandardColor 8](#_Toc157447016)

[type TJDStandardColors 8](#_Toc157447017)

[type TJDColorMode 8](#_Toc157447018)

[type TJDCHue 8](#_Toc157447019)

[type TJDCSaturation 8](#_Toc157447020)

[type TJDCBrightness 8](#_Toc157447021)

[type TJDColor 8](#_Toc157447022)

[class TJDColorRGBRef 9](#_Toc157447023)

[class TJDColorHSVRef 9](#_Toc157447024)

[class TJDColorCMYKRef 9](#_Toc157447025)

[class TJDColorRef 9](#_Toc157447026)

[class TJDColorManager 9](#_Toc157447027)

[class TJDCanvas 9](#_Toc157447028)

[function DetectColorMode 9](#_Toc157447029)

[function RGBToHSV 9](#_Toc157447030)

[function HSVToRGB 10](#_Toc157447031)

[function ColorToHtml 10](#_Toc157447032)

[function ColorToHtml2 10](#_Toc157447033)

[function HtmlToColor 10](#_Toc157447034)

[function TweakColor 10](#_Toc157447035)

[function DrawParentImage 10](#_Toc157447036)

[function PointAroundCenter 10](#_Toc157447037)

[function DrawTextJD 10](#_Toc157447038)

[function RectToGPRect 10](#_Toc157447039)

[function ColorToGPColor 11](#_Toc157447040)

[function ColorManager 11](#_Toc157447041)

[JD.SmoothMove.pas 12](#_Toc157447042)

[class TJDSmoothMove 12](#_Toc157447043)

[Usage 12](#_Toc157447044)

[TODO 12](#_Toc157447045)

[type TJDSmoothMoveEffect 12](#_Toc157447046)

[type TJDSmoothMoveEvent 12](#_Toc157447047)

[class TJDSmoothMoveThread 12](#_Toc157447048)

[JD.FontGlyphs.pas 13](#_Toc157447049)

[class TJDFontGlyphs 13](#_Toc157447050)

[Usage 13](#_Toc157447051)

[TODO 13](#_Toc157447052)

[class TJDFontGlyphRef 13](#_Toc157447053)

[class TJDFontGlyph 13](#_Toc157447054)

[class TJDImageListRef 13](#_Toc157447055)

[class TJDImageListRefs 13](#_Toc157447056)

[class TJDFontGlyphItem 13](#_Toc157447057)

[TODO 13](#_Toc157447058)

[class TJDFontGlyphList 14](#_Toc157447059)

[JD.Ctrls.FontButton.pas 15](#_Toc157447060)

[Constants 15](#_Toc157447061)

[type TJDFontButtonImgPosition 15](#_Toc157447062)

[type TJDFontButtonState 15](#_Toc157447063)

[type TJDFontButtonStyleColor 15](#_Toc157447064)

[type TJDFontButtonStyleColors 15](#_Toc157447065)

[type TJDFontButtonDrawStyle 15](#_Toc157447066)

[type TJDFontButtonKind 15](#_Toc157447067)

[type TJDFontButtonSubTextStyle 15](#_Toc157447068)

[type TJDFontButtonOverlayPosition 15](#_Toc157447069)

[class TJDFontButtonImage 16](#_Toc157447070)

[class TJDFontButtonOverlay 16](#_Toc157447071)

[class TJDFontButton 16](#_Toc157447072)

[Usage 16](#_Toc157447073)

[TODO 16](#_Toc157447074)

[function DrawThemeBackground 16](#_Toc157447075)

[function OpenThemeData 16](#_Toc157447076)

[function CloseThemeData 16](#_Toc157447077)

[JD.Ctlrs.Gauges.pas 17](#_Toc157447078)

[class TJDGauge 17](#_Toc157447079)

[Usage 17](#_Toc157447080)

[TODO 17](#_Toc157447081)

[class TJDGaugeBase 17](#_Toc157447082)

[class TJDGaugeValue 17](#_Toc157447083)

[class TJDGaugeValues 18](#_Toc157447084)

[class TJDGaugePeak 18](#_Toc157447085)

[type TJDGaugeTypeClass 18](#_Toc157447086)

[type TJDGaugeBaseClass 18](#_Toc157447087)

[class TJDGaugeClassList 18](#_Toc157447088)

[type TJDGaugeTickLabelOrientation 18](#_Toc157447089)

[class TJDGaugeTickLabels 18](#_Toc157447090)

[class TJDGaugeTicks 18](#_Toc157447091)

[class TJDGaugeGradient 18](#_Toc157447092)

[JD.Ctrls.Gauges.Objects.pas 19](#_Toc157447093)

[class TJDGaugeCircle 19](#_Toc157447094)

[class TJDGaugeHorzBar 19](#_Toc157447095)

[class TJDGaugeVertBar 19](#_Toc157447096)

[class TJDGaugePie 19](#_Toc157447097)

[JD.SysMon.pas 20](#_Toc157447098)

[JD.VolumeControls.pas 20](#_Toc157447099)

[JD.AppBar.pas 20](#_Toc157447100)

[JD.Ctrls.ImageGrid.pas 20](#_Toc157447101)

[JD.Ctrls.PageMenu.pas 20](#_Toc157447102)

[JD.Ctrls.SideMenu.pas 20](#_Toc157447103)

[JD.Sockets.pas 20](#_Toc157447104)

[JD.Vector.pas 20](#_Toc157447105)

[JD.SuperObject.pas 20](#_Toc157447106)

[JD.Ctrls.ControlList.pas 20](#_Toc157447107)

[JD.Ctrls.BtnList.pas 21](#_Toc157447108)

# Overview

JDLib is a library of custom components and controls written by Jerry Dodge. It is contained within 2 packages – JDLibComponents.bpl (Runtime) and dclJDLibComponents.bpl (Designtime).

## JDLibComponents.bpl

JDLib run-time package

### Contains

1. [JD.Common.pas](#_JD.Common.pas)
2. [JD.Graphics.pas](#_JD.Graphics.pas)
3. [JD.Ctrls.pas](#_JD.Ctrls.pas)
4. [JD.Ctrls.BtnList.pas](#_JD.Ctrls.BtnList.pas)
5. [JD.Ctrls.ControlList.pas](#_JD.Ctrls.ControlList.pas)
6. [JD.Ctrls.FontButton.pas](#_JD.Ctrls.FontButton.pas)
7. [JD.Ctrls.Gauges.pas](#_JD.Ctlrs.Gauges.pas)
8. [JD.Ctrls.Gauges.Objects.pas](#_JD.Ctrls.Gauges.Objects.pas)
9. [JD.Ctrls.ImageGrid.pas](#_JD.Ctrls.ImageGrid.pas)
10. [JD.Ctrls.PageMenu.pas](#_JD.Ctrls.PageMenu.pas)
11. JD.Ctrls.PlotChart.pas
12. [JD.Ctrls.SideMenu.pas](#_JD.Ctrls.SideMenu.pas)
13. [JD.FontGlyphs.pas](#_JD.FontGlyphs.pas)
14. [JD.SmoothMove.pas](#_JD.SmoothMove.pas)
15. [JD.AppBar.pas](#_JD.AppBar.pas)
16. [JD.Sockets.pas](#_JD.Sockets.pas)
17. [JD.SuperObject.pas](#_JD.SuperObject.pas)
18. [JD.SysMon.pas](#_JD.SysMon.pas)
19. [JD.Vector.pas](#_JD.Vector.pas)
20. [JD.VolumeControls.pas](#_JD.VolumeControls.pas)

### Requires

1. IndyCore
2. IndyProtocols
3. IndySystem
4. rtl
5. vclimg
6. VclSmp

## dclJDLibComponents.pbl

JDLib design-time package

### Contains

1. JD.ColorPropEdit.pas
2. JD.Ctrls.CharPicker.pas
3. JD.Ctrls.FontButtonEdit.pas
4. JD.Ctrls.FontButtonEditCtrl.pas
5. JD.Ctrls.JDGaugeEditors.pas
6. JD.Ctrls.PageMenuEditor.pas
7. JD.Ctrls.WinControlHelper.pas
8. JDLib.InstallComps.pas

### Requires

1. DesignIDE
2. [JDLibComponents](#_JDLibComponents.bpl)
3. rtl

## Global TODO

1. Implement .INC file to centralize conditionals and constants.
   1. Include at the top of ALL units.
   2. Define JDLib version.
   3. Detect Delphi version.
2. Fully implement optional GDI+ via USE\_GDIP conditional
   1. If not enabled, fallback to basic GDI.
3. Figure out why trying to compile after already installed crashes the IDE.
   1. Most likely related to [TJDGauge](#_class_TJDGauge) control and its peak decay thread.
4. Make use of [TJDRect](#_type_TJDRect) and [TJDPoint](#_type_TJDPoint) ([JD.Common.pas](#_JD.Common.pas)) across the entire library.
5. Make use of [DrawTextJD](#_function_DrawTextJD) ([JD.Graphics.pas](#_JD.Graphics.pas)) across the entire library.
6. Add more units / components / stuff of mine from the past.

# JD.Common.pas

## const WM\_JD\_COLORCHANGE

WM\_JD\_COLORCHANGE = WM\_USER + 42;

Windows message for when color themes change.

## const SC\_DRAGMOVE

SC\_DRAGMOVE = $F012;

Windows message when user drags window / control.

## class EJDException

EJDException = Exception;

Base exception type for us across JDLib.

## class EJDOutOfRange

EJDOutOfRange = EJDException;

Exception for when a value is out of a given range.

## type PJDPoint

PJDPoint = ^[TJDPoint](#_type_TJDPoint);

Pointer to type [TJDPoint](#_type_TJDPoint)

## type TJDPoint

TJDPoint = record

Defines a standardized point with floating point values at its root.

1. class operator Implicit(Value: TJDPoint): TPoint;
2. class operator Implicit(Value: TPoint): TJDPoint;
3. class operator Implicit(Value: TJDPoint): TGPPointF;
4. class operator Implicit(Value: TGPPointF): TJDPoint;
5. property X: Single read FX write SetX;
6. property Y: Single read FY write SetY;
7. procedure Move(const AmtX, AmtY: Single);

## type PJDRect

PJDRect = ^[TJDRect](#_type_TJDRect);

Pointer to type [TJDRect](#_type_TJDRect)

## type TJDRect

TJDRect = record

Defines a standardized rectangle with floating point values at its root.

1. class operator Implicit(Value: TRect): TJDRect;
2. class operator Implicit(Value: TJDRect): TRect;
3. class operator Implicit(Value: TRectF): TJDRect;
4. class operator Implicit(Value: TJDRect): TRectF;
5. class operator Implicit(Value: TGPRectF): TJDRect;
6. class operator Implicit(Value: TJDRect): TGPRectF;
7. property X: Single read GetX write SetX;
8. property Y: Single read GetY write SetY;
9. property Width: Single read GetWidth write SetWidth;
10. property Height: Single read GetHeight write SetHeight;
11. property Right: Single read GetRight write SetRight;
12. property Bottom: Single read GetBottom write SetBottom;
13. property Left: Single read GetLeft write SetLeft;
14. property Top: Single read GetTop write SetTop;
15. procedure Inflate(const AmtX, AmtY: Single);
16. procedure Deflate(const AmtX, AmtY: Single);
17. procedure Move(const AmtX, AmtY: Single);
18. function TopLeft: TJDPoint;
19. function TopRight: TJDPoint;
20. function BottomLeft: TJDPoint;
21. function BottomRight: TJDPoint;
22. function Center: TJDPoint;
23. function TopCenter: TJDPoint;
24. function BottomCenter: TJDPoint;
25. function LeftCenter: TJDPoint;
26. function RightCenter: TJDPoint;

## class TJDComponent

TJDComponent = class(TComponent)

Very core base for all custom JD components. This has the sole purpose of forcing these units to be included in the uses clause automatically.

**NOTE**: Use [TJDMessageComponent](#_class_TJDMessageComponent) instead for Windows message handling.

## class TJDMessageComponent

TJDMessageComponent = class([TJDComponent](#_class_TJDComponent))

Base component to efficiently receive Windows messages.

1. constructor Create(AOwner: TComponent); override;
2. destructor Destroy; override;
3. procedure Invalidate; virtual;
4. property Handle: HWND read FHandle;

## class TJDCustomControl

TJDCustomControl = class(TCustomControl)

Very core base for all custom JD controls. Gets inherited by TJDControl in JD.Ctrls for the purpose of forcing these units to be included in the uses clause automatically.

1. constructor Create(AOwner: TComponent); override;
2. destructor Destroy; override;

## function IntRange

function IntRange(const Value, Min, Max: Integer): Integer;

Ensures an integer falls within a given min/max range.

1. Value
2. Min
3. Max

# JD.Ctrls.pas

## class TJDControl

TJDControl = class([TJDCustomControl](#_class_TJDCustomControl))

Base class for all JD custom controls.

Currently for the sole purpose of automatically adding this unit to the uses clause of any form which uses a JD custom control.

Also applies to [TJDCustomControl](#_class_TJDCustomControl) as defined in [JD.Common](#_JD.Common.pas). This way, whenver any JD custom control is implemented on any form, both [JD.Ctrls](#_JD.Ctrls.pas) and [JD.Common](#_JD.Common.pas) are automatically included in the uses clause.

# JD.Graphics.pas

JD Graphics is a collection of graphic related tools to use throughout JDLib. This includes especially standardized colors for light and dark modes, and flexible color references which can convert between RGB, HSV, and CMYK.

## Constants

(TODO)

## type TJDStandardColor

TJDStandardColor = (fcNeutral, fcGray, fcBlue, fcGreen,

fcRed, fcYellow, fcOrange, fcPurple);

JD standard colors, to avoid having to decide on specific colors.

Automatically differs whether using light or dark modes.

Customizable via [ColorManager](#_function_ColorManager) ([TJDColorManager](#_class_TJDColorManager)).

## type TJDStandardColors

TJDStandardColors = array[[TJDStandardColor](#_type_TJDStandardColor)] of TColor;

An array of TColor values per [TJDStandardColor](#_type_TJDStandardColor) value.

## type TJDColorMode

TJDColorMode = (cmLight, cmMedium, cmDark);

Defines whether in light or dark mode.

Middle mode not yet supported.

## type TJDCHue

TJDCHue = record

Hue value associated with HSV in [TJDColor](#_type_TJDColor).

## type TJDCSaturation

TJDCSaturation = record

Saturation value associated with HSV in [TJDColor](#_type_TJDColor).

## type TJDCBrightness

TJDCBrightness = record

Brightness value associated with HSV in [TJDColor](#_type_TJDColor).

## type TJDColor

TJDColor = record

A universal color reference, and implicitly casts with TColor.

1. Directly read and write different color profiles, such as RGB, HSV, and CMYK.

## class TJDColorRGBRef

TJDColorRGBRef = class(TPersistent)

## class TJDColorHSVRef

TJDColorHSVRef = class(TPersistent)

## class TJDColorCMYKRef

TJDColorCMYKRef = class(TPersistent)

## class TJDColorRef

TJDColorRef = class(TPersistent)

A published property in custom components / controls to define a standardized color property.

## class TJDColorManager

TJDColorManager = class(TObject)

Used to access system-wide color options.

1. Do not create directly – instead use [ColorManager](#_function_ColorManager) function to access central object.

## class TJDCanvas

TJDCanvas = class(TPersistent)

## function DetectColorMode

function DetectColorMode(const AColor: TColor): [TJDColorMode](#_type_TJDColorMode);

## function RGBToHSV

function RGBToHSV(R, G, B: Byte; var H, S, V: Double): Boolean;

## function HSVToRGB

function HSVToRGB(H, S, V: Double; var R, G, B: Byte): Boolean;

## function ColorToHtml

function ColorToHtml(Color: TColor): string;

## function ColorToHtml2

function ColorToHtml2(Clr: TColor): string;

## function HtmlToColor

function HtmlToColor(Color: string): TColor;

## function TweakColor

function TweakColor(const AColor: TColor; const Diff: Integer): TColor;

## function DrawParentImage

procedure DrawParentImage(Control: TControl; Dest: TCanvas);

## function PointAroundCenter

function PointAroundCenter(Center: [TJDPoint](#_type_TJDPoint); Distance: Integer; Degrees: Single;

OvalOffset: Single = 1): [TJDPoint](#_type_TJDPoint);

## function DrawTextJD

function DrawTextJD(hDC: HDC; Str: String;

var lpRect: [TJDRect](#_type_TJDRect); uFormat: UINT): Integer;

## function RectToGPRect

function RectToGPRect(R: TRect): TGPRectF;

## function ColorToGPColor

function ColorToGPColor(C: TColor): Cardinal;

## function ColorManager

function ColorManager: [TJDColorManager](#_class_TJDColorManager);

1. Accesses the central instance of [TJDColorManager](#_class_TJDColorManager).
2. Use BaseColor property to assign the base color of your theme. For example, clBlack if you’re using dark mode with black backgrounds.
3. Modify the colors used for [TJDStandardColor](#_type_TJDStandardColor) in light and dark modes by using “ColorNew” property.

# JD.SmoothMove.pas

[TJDSmoothMove](#_class_TJDSmoothMove) is a component which accepts a float value and triggers events to modify an external value in a “smooth” motion. Similar to “Float Animations”, but in its own standalone component. Value changes are triggered by an internal thread. This eliminates the need for a TTimer which relies on Windows messages to animate float values.

## class TJDSmoothMove

TJDSmoothMove = class(TComponent)

Main component which encapsulates Smooth Move animations.

### Usage

1. Drop a TJDSmoothMove component into your form.
2. Use the Delay property to control time interval between calculations / animations.
3. Use the Step property to control the amount the float value changes between each step.
4. Assign a value the Value property in runtime to trigger float value animation.
5. Create a handler for OnValue event which uses the Position parameter to set an external value of your choice.

### TODO

1. Finish implementing “seSmooth” type, which slows down when it’s approaching its target value.

## type TJDSmoothMoveEffect

TJDSmoothMoveEffect = (seNone, seNormal, seSmooth);

Defines the effect of the animation.

1. seNone – No animation – input value immediately changes output value.
2. seNormal – Normal animation – input value evenly changes to output value.
3. seSmooth – Smooth animation – input value slows down when approaching output value.

## type TJDSmoothMoveEvent

TJDSmoothMoveEvent = procedure(Sender: TObject; const Position: Double) of object;

Event triggered upon value changes depending on effect type.

## class TJDSmoothMoveThread

TJDSmoothMoveThread = class(TThread)

Dedicated thread used internally to animate value changes.

# JD.FontGlyphs.pas

[TJDFontGlyphs](#_class_TJDFontGlyphs) is a component which attaches to one or more TImageList components to automatically populate them with selected font glyphs.

## class TJDFontGlyphs

Main component encapsulating a list of image lists and a list of font glyphs to be populated in those image lists.

### Usage

1. Drop as many TImageList components as you need and assign different sizes to each.
2. Drop a TJDImageGlyphs component into your form.
3. Use the ImageLists property to add a reference to each of your TImageList components.
   1. Add a new item to the collection.
   2. Assign corresponding TImageList component to ImageList property.
4. Use the Glyphs property to add references to each possible glyph you want in your TImageList components.
   1. Add a new item to the collection.
   2. Use the Ref property and its property editor to choose a font (such as FontAwesome) and then a glyph within that font.
   3. Use the Color property to choose a color for this glyph.
5. All the TImageList components will automatically get populated with the glyphs you created in the Glyphs property. Any changes made to those glyphs will regenerate those images in all referenced image lists.

### TODO

## class TJDFontGlyphRef

Represents a reference to a particular font glyph with options for font name, character, scale, and color.

Also provides property editor to pick glyph and options.

## class TJDFontGlyph

Represents a particular font glyph with options for font, character, scale, and color.

Also provides property editor to pick glyph and options.

For use on any component which wishes to implement font glyph properties.

## class TJDImageListRef

## class TJDImageListRefs

## class TJDFontGlyphItem

### TODO

1. Rename Ref property to Glyph and remove Glyph string property.
   1. NOTE: Gracefully convert Glyph property from String to [TJDFontGlyphRef](#_class_TJDFontGlyphRef)?

## class TJDFontGlyphList

# JD.Ctrls.FontButton.pas

[TJDFontButton](#_class_TJDFontButton) is a button control which accepts a font glyph for an image instead of VCL’s standard button glyphs. Custom property editors allow you to scroll and choose a glyph icon within a given font, and apply color, size, and other properties.

**IMPORTANT NOTE: Currently, this is still named “TFontButton” from its original version, but will soon be renamed to “TJDFontButton”.**

## Constants

TODO

## type TJDFontButtonImgPosition

## type TJDFontButtonState

## type TJDFontButtonStyleColor

## type TJDFontButtonStyleColors

## type TJDFontButtonDrawStyle

## type TJDFontButtonKind

## type TJDFontButtonSubTextStyle

## type TJDFontButtonOverlayPosition

## class TJDFontButtonImage

## class TJDFontButtonOverlay

## class TJDFontButton

### Usage

1. Drop a TJDFontButton control onto your form.

### TODO

1. Rename / refactor all identifiers to include JD prefix.
   1. Mainly TJDFontButton itself, and deal with broken references in other projects.
2. Change “Image” property to “Glyph” using the new [TJDFontGlyphRef](#_class_TJDFontGlyphRef) persistent.

## function DrawThemeBackground

## function OpenThemeData

## function CloseThemeData

# JD.Ctrls.Gauges.pas

## class TJDGauge

TJDGauge = class(TJDControl)

TJDGauge is a control to display a variety of different gauge types, ranging from circle, pie, arc, horizontal bar, vertical bar, and has the ability to create custom gauge types to attach. This allows you to simply switch between these different gauge types on the fly by changing a single property.

### Usage

1. Drop a TJDGauge control onto your form.
2. Assign its GaugeType property to the gauge type of your choice.
3. Use the MainValue property to access the primary gauge value.
4. Use the Values property to create more values than just the main.

### TODO

1. Implement better scaling, Automatic or manual.
   1. Need to base everything off a center point, instead of current ClientRect method.
2. Finish implementing “Min” aspect of values (currently assumes 0 and ignores “Min”).
3. Implement “PctBegin” and “PctEnd” properties on values to specify the range on the base for the gauge.
4. Implement “Grouping” property to enable stacking or merging of values instead of overlaying them.
5. Implement “OffsetPos” property on values to specify distance from main point for a gauge value.
6. Implement “Split” gauges, which assume two from the center.
7. Change peak decay thread to a central thread that manages all registered gauges.
   1. If hundreds of gauges are implemented, then hundreds of threads would be created currently…
8. Finish implementing drawing glyphs.
9. Implement auto-scaling of glyphs.
10. Finish implementing drawing captions.
11. Implement auto-scaling of captions.
12. Implement drawing sub captions.
13. Implement auto-scaling of sub captions.
14. Implement tick marks, both major and minor.
15. Implement hints per value.
16. Implement more gauge types…
    1. Needle gauges…
    2. Segmented gauges…
17. Implement color changing depending on value…
    1. Color intensity based on value…
    2. Collection of ranges as defined by dev…
18. Implement gradients
    1. Will require redesigning line-based gauges…
19. Implement clicking values

## class TJDGaugeBase

TJDGaugeBase = class(TObject)

Base abstract class for any given implementation of TJDGauge. Each possible gauge type inherits from this base and overrides several methods to implement a standardized gauge. You can also inherit this into your own class for a custom gauge implementation which you can switch to.

## class TJDGaugeValue

## class TJDGaugeValues

## class TJDGaugePeak

## type TJDGaugeTypeClass

## type TJDGaugeBaseClass

## class TJDGaugeClassList

## type TJDGaugeTickLabelOrientation

## class TJDGaugeTickLabels

## class TJDGaugeTicks

## class TJDGaugeGradient

# JD.Ctrls.Gauges.Objects.pas

## class TJDGaugeCircle

inherits from TJDGaugeBase

## class TJDGaugeHorzBar

inherits from TJDGaugeBase

## class TJDGaugeVertBar

inherits from TJDGaugeBase

## class TJDGaugePie

inherits from TJDGaugeBase

# JD.SysMon.pas

# JD.VolumeControls.pas

# JD.AppBar.pas

# JD.Ctrls.ImageGrid.pas

# JD.Ctrls.PageMenu.pas

# JD.Ctrls.SideMenu.pas

# JD.Sockets.pas

# JD.Vector.pas

# JD.SuperObject.pas

# JD.Ctrls.ControlList.pas

# JD.Ctrls.BtnList.pas

# JD.CtrlsPlotChart.pas