



Using Ruler .Net Control

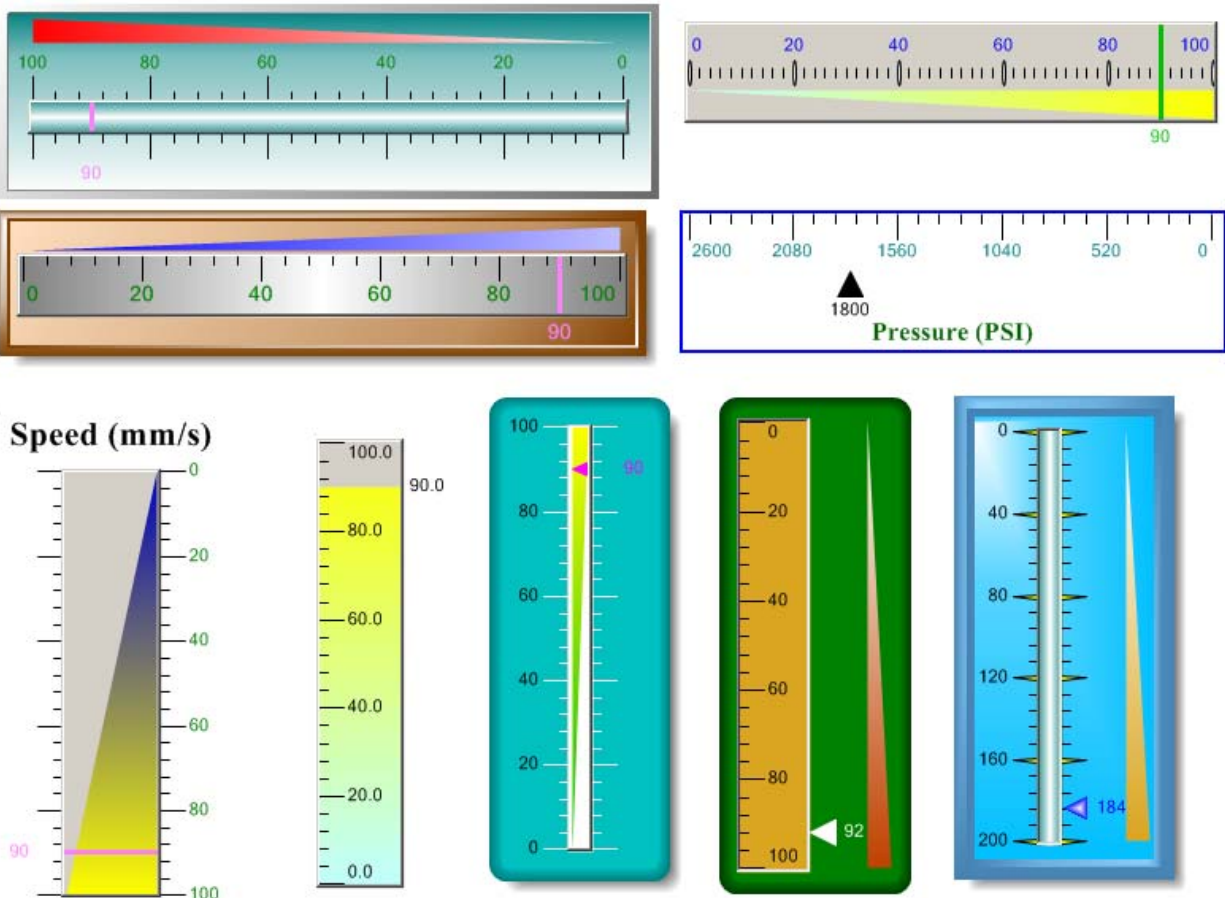
1. Overview

The DAS_Net_Ruler .Net Windows Form control allows users to visual show some dynamic values on a scaled ruler. This control can be used in many applications such as Factory Instrumentation Readouts, Process Control User Interface, Motion Control Monitor and other monitor displays.

DAS_Net_Ruler is very powerful for designers to configure different kinds of process monitor displays which present visual scale effect. There are a rich group of the properties which enable people to design many kinds of displays to their requirements. All these properties will be addressed in the following sections.

2. DAS_Net_Ruler

The designers can use the interface properties and methods to implement many kinds of displays, the following picture shows some styles which can be configured.

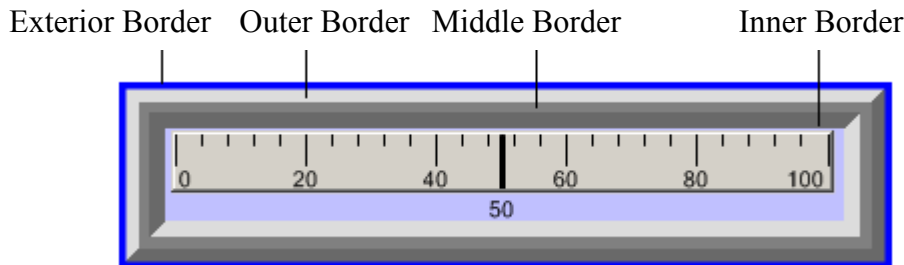




Properties:

Border Properties:

To make the control more intuitive and vivid look-and-feel, a group of border properties are provided for the designers to configure the control's border style. There are four border layers, i.e., Exterior Border layer, Outer Border layer, Middle Border layer and Inner Border layer,



We can configure different color and layer length for each layer. Especially for Outer-Border and Inner-Border, the light color and the dark color can be configured to generate different gradient 3D-Border feeling.

Property *DAS_BorderStyle BorderShape*

Specify which border shape (Rectangle or Round Rectangle) is used to draw the border of control. There are three definitions, i.e., *BS_Rect* defines Rectangle Shape, *BS_RoundRect* defines Round Rectangle Shape (The size of the round corners is defined by the property *RoundRadius*).

Property *int BorderExteriorLength*

Specify the border length of the Exterior Border, this layer is drawn using the color defined by the property *BorderExteriorColor*. The default length of this layer is zero.

Property *Color BorderExteriorColor*

Specify the color to render the exterior border layer.

Property *int OuterBorderLength*

Specify the border length of the outer border, this layer is drawn using the colors defined by the property *OuterBorderDarkColor* and *OuterBorderLightColor*. This layer can be rendered in different gradient modes which is defined by the property *BorderGradientType*.

Property *Color OuterBorderDarkColor*

Specify the dark color of the outer border that is hidden from the light.

Property *Color OuterBorderLightColor*

Specify the light color of the outer border that faces the light.

Property *int InnerBorderLength*



Specify the border length of the inner border, this layer is drawn using the colors defined by property *InnerBorderDarkColor* and *InnerBorderLightColor*. This layer can be rendered in different gradient modes which is defined by the property *BorderGradientType*.

Property Color InnerBorderDarkColor

Specify the dark color of the inner border that is hidden from the light.

Property Color InnerBorderLightColor

Specify the light color of the inner border that faces the light.

Property int MiddleBorderLength

Specify the length of the middle border layer, this layer is rendered by property *MiddleBorderColor*.

Property Color MiddleBorderColor

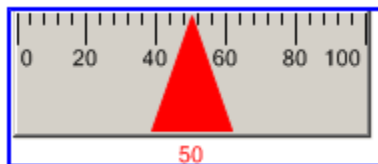
Specify the color to render the middle border layer.

Property int RoundRadius

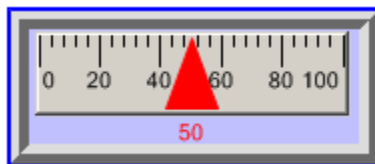
Specify the round corner size of the round-rectangle shape.

Border Gradient Properties:

There are six approaches defined in *DAS_BorderGradientStyle* to render the control border,



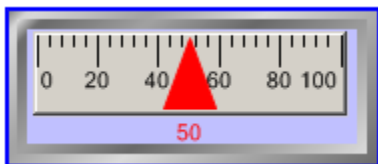
BGS_None



BGS_Flat



BGS_Ring



BGS_Linear



BGS_Linear2



BGS_Path

BGS_None ---- Ignore the Outer Layer, Middle Layer and Inner Layer, only the Exterior layer is rendered.

BGS_Flat ---- All layers are rendered. The left side and top side of the Outer Layer are rendered using *OuterBorderLightColor*, the right side and bottom side of the Outer Layer are rendered using *OuterBorderDarkColor*; The left side and top side of the Inner Layer are rendered using



InnerBorderDarkColor, the right side and bottom side of the Inner Layer are rendered using *InnerBorderLightColor*;

BGS_Ring ---- All layers are rendered. The outer layer is rendered in gradient mode at radial direction from *OuterBorderDarkColor* at the outer most periphery of the outer layer to *OuterBorderLightColor* at the inner most periphery of the outer layer. The inner layer is rendered in gradient mode at radial direction from *InnerBorderLightColor* at the outer most periphery of the inner layer to *InnerBorderDarkColor* at the inner most periphery of the inner layer.

BGS_Linear --- All layers are rendered. The outer layer is rendered in gradient mode at the direction defined by property *BorderGradientAngle* from *OuterBorderLightColor* at the one end (defined by property *BorderGradientLightPos1* and *BorderGradientLightPos2*) to *OuterBorderDarkColor* at the other end of the outer layer. The inner layer is rendered in gradient mode at the direction defined by property *BorderGradientAngle* from *InnerBorderDarkColor* at the one end (defined by property *BorderGradientLightPos1* and *BorderGradientLightPos2*) of the inner layer to *InnerBorderLightColor* at the other end of the inner layer.

BGS_Linear2 --- Similar to BGS_Linear. The difference is the definitions of property *BorderGradientLightPos1* and *BorderGradientLightPos2* of inner layer. The inner layer is rendered in gradient mode at the direction defined by property *BorderGradientAngle* from *InnerBorderDarkColor* at the one end of the inner layer to *InnerBorderLightColor* at the other end (defined by property *BorderGradientLightPos1* and *BorderGradientLightPos2*) of the inner layer.

BGS_Path ---- All layers are rendered. The outer layer is rendered in gradient mode at the radial direction from *OuterBorderDarkColor* at the outer most periphery to *OuterBorderLightColor* at the center point that is in the outer layer at the direction defined by property *BorderGradientAngle*.. The inner layer is rendered in gradient mode at the radial direction from *InnerBorderDarkColor* at the outer most periphery to *InnerBorderLightColor* at the center point that is in the outer layer at the direction defined by (*BorderGradientAngle* + 180°).

Property DAS BorderGradientStyle BorderGradientType

Specify which border gradient type is applied to render the border of the control.

Property int BorderGradientAngle

Specify the gradient angle for BGS_Linear, BGS_Linear2 and BGS_Path.



Property float BorderGradientRate

Specify the gradient focus size. It should be within 0.0 to 1.0.

Property float BorderGradientLightPos1

Specify the gradient light position 1 at the gradient angle. It should be within 0.0 to 1.0. This property just takes effect for *BGS_Linear* and *BGS_Linear2*.

Property float BorderGradientLightPos2

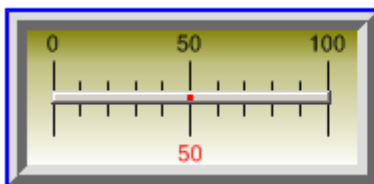
Specify the another gradient light position at the gradient angle. It should be within 0.0 to 1.0. If it is negative, the second gradient light position is ignored. This property just takes effect for *BGS_Linear* and *BGS_Linear2*.

Property float BorderLightIntermediateBrightness

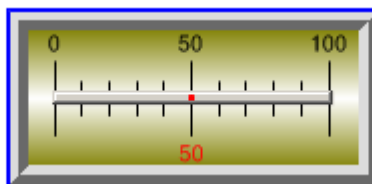
Specify the color at the middle point between the light position 1 and the light position2. It should be within 0.0 to 1.0. If it's 0, the dark color is applied at middle point, if it's 1.0, the light color is used at the middle point. This property just takes effect for *BGS_Linear* and *BGS_Linear2*.

Background Properties:

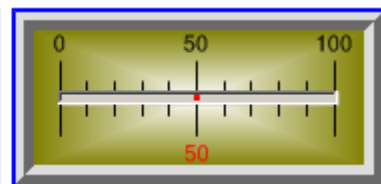
In general, the background of the control can be rendered in gradient mode or normal mode. And the background image (defined by *BackgroundImage*) can be rendered as well using the standard image rendering methods (defined by *BackgroundImageLayout*), like "Tile", "Zoom", "Stretch", "Center" and "None", please refer to the standard background image rendering in MSDN. If *BackgroundImage* is set, the image is first rendered, and then the back color with transparency (defined by *BkTransparency*) is rendered. There are five background gradient types are defined as follows (*DAS_BkGradientStyle*),



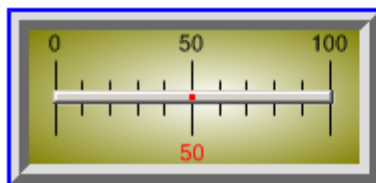
BKGS_Linear



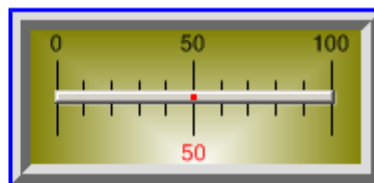
BKGS_Linear2



BKGS_Polygon



BKGS_Sphere



BKGS_Shine

The gradient direction for *BKGS_Linear*, *BKGS_Linear2* and *BKGS_Shine* is defined by the property *BkGradientAngle*.



Property Color BackColor

Specify the background color of the control.

Property Image BackGroundImage

Specify the background image of the control. For details, refer to MSDN.

Property ImageLayout BackGroundImageLayout

Specify the background image layout style. For details, refer to MSDN.

Property float BkTransparency

Specify the back color rendering transparency rate. If it's 0.0, there should be no transparency, so if BackGroundImage is set, but BkTransparency is zero, the image is invisible. If it's 1.0, the image is fully visible, and the gradient back color rendering is not invisible.

Property float BkGradientRate

Specify the gradient focus size, it should be between 0.0 and 1.0.

Property bool BkGradient

Specify whether the background is rendered in gradient mode or normal mode.

Property int BkGradientAngle

Specify the background gradient direction angle for *BKGS_Linear*, *BKGS_Linear2* and *BKGS_Shine*)

Property Color BkGradientColor

Specify the background gradient color.

Property DAS BkGradientStyle BkGradientType

Specify the background gradient type.

Property float BkShinePosition

Specify the background gradient shine (focus) position (for *BKGS_Shine*), it should be between 0.0 and 1.0. In conjunction with *BkGradientAngle*, the gradient shine position can be fully adjusted in the background rendering area.

Shadow Properties:

Shadow display is a basic feature of the Dragonfly .Net controls.

Property bool ControlShadow

Specify whether to show the shadow or not.

Property int ShadowDepth

Specify the depth of the shadow.

Property Color ShadowColor

Specify the color of the shadow.



Property float ShadowRate

Specify the emission rate of the shadow.

General Properties:

Property Color ForeColor

Specify the color to draw the ruler name and unit.

Property bool Vertical

Specify whether the ruler is drawn vertically or horizontally.

Property int Precision

Specify the decimal digit number for the scale texts and actual value display. Default is zero.

Property double Value

Specify the value of the ruler.

Property string RulerName

Specify the name of the ruler.

Property string Unit

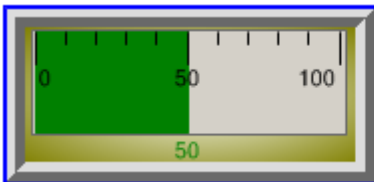
Specify the unit of the ruler value.

Property Color ValueColor

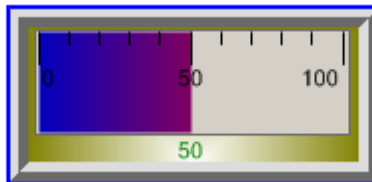
Specify the color to draw the value indicator and the actual value of the ruler.

Property DAS ValueIndicatorStyle ValueIndicator

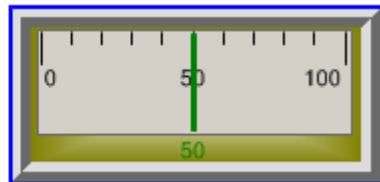
Specify the value indicator type. There are six definitions in `DAS_ValueIndicatorStyle`, i.e.,



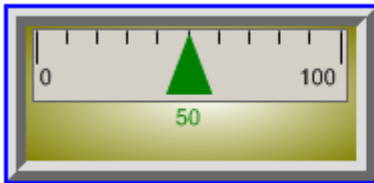
VIS_Progress_Bar



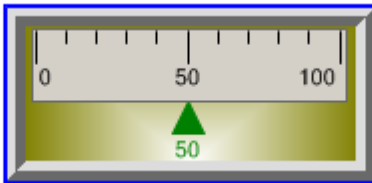
VIS_Gradient_Bar



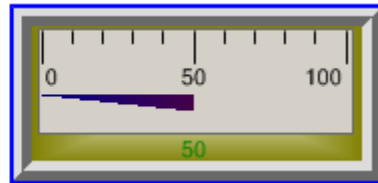
VIS_Line



VIS_Inner_Triangle



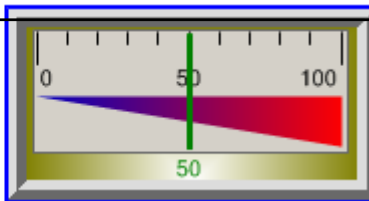
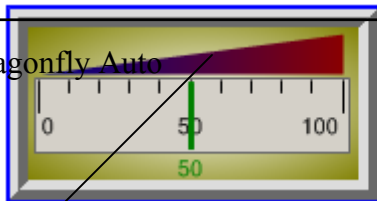
VIS_Outer_Triangle



VIS_Progress_Triangle

Property bool ProgressTriangleVisible

Specify the visibility of the progress triangle.





Property Color ProgressLowColor

Specify the color to represent the lower part of the progress triangle or the lower part of the value indicator when its type is VIS_Gradient_Bar or VIS_Progress_Triangle.

Property Color OuterProgressTriangleColor

Specify the color to represent the higher part of the progress triangle or the lower part of the value indicator when its type is VIS_Gradient_Bar or VIS_Progress_Triangle.

Inner Progress Triangle

Property bool InnerProgressTriangle

Specify whether the inner progress triangle is shown or the outer progress triangle is shown.

Property int ProgressTriangleWidth

Specify the size of the progress triangle. It only takes effect for the outer progress triangle.

Property Font NameFont

Specify the standard font to draw the name and the unit.

Property int NameUnitXOffset

Specify the X offset to draw the name and unit. Which is used to adjust the horizontal position of the name and unit.

Property int NameUnitYOffset

Specify the Y offset to draw the name and unit. Which is used to adjust the vertical position of the name and unit.

Property bool TriangleIndicatorGradient

Specify whether the triangle value indicator (inner or outer) is drawn in gradient mode or not.

Ruler Face Properties:

The following properties are about the face of the ruler.

Property Color RulerFaceColor

Specify the color to render the ruler face.

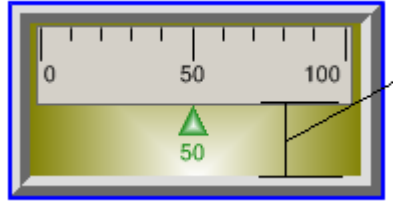
Property Color RulerFaceGradientColor

Specify the gradient color to render the ruler face.

Property int BorderRulerGap

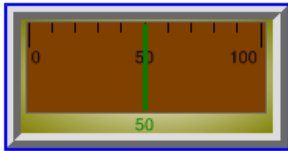
Specify the gap distance between the border of the control and the ruler face.

BorderRulerGap

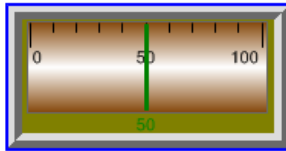


Property DAS RulerFaceGradientStyle RulerFaceGradientType

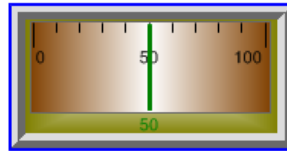
Specify the gradient type to render the ruler face. There are four definitions in DAS_RulerFaceGradientStyle, i.e.,



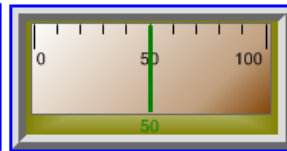
RFGS_None



RFGS_Horizontal



RFGS_Vertical



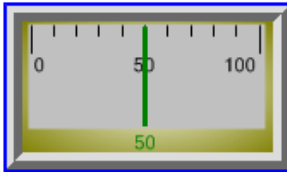
RFGS_Linear

Property float RulerFaceGradientRate

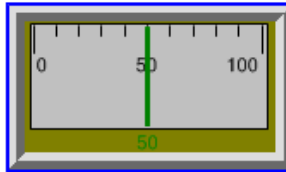
Specify the gradient focus size in gradient mode. It should be between 0.0 and 1.0.

Property DAS FaceBorderStyle RulerFaceBorderStyle

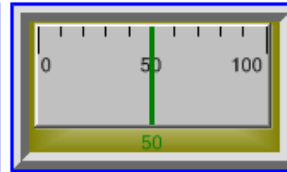
Specify the border type of the ruler face. There are four definitions in DAS_FaceBorderStyle, i.e.,



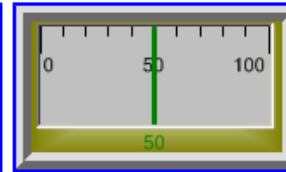
FBS_None



FBS_Flat



FBS_Raised



FBS_Sunken

Property Color RulerFaceBorderColor

Specify the color to draw the border of the ruler face.

Scale Properties:

The following properties are used to define the scale.

Property double Max

Specify the maximum value of the scale.

Property double Min

Specify the minimum value of the scale.

Property bool InverseScaleDirection

Specify whether to invert the scale direction or not. Default direction is Left_To_Right for Min_To_Max (Horizontal) or Bottom_To_Top for Min_To_Max (Vertical).

Property int TickerNumber

Specify how many tickers are drawn for the scale. The minimum is 2.

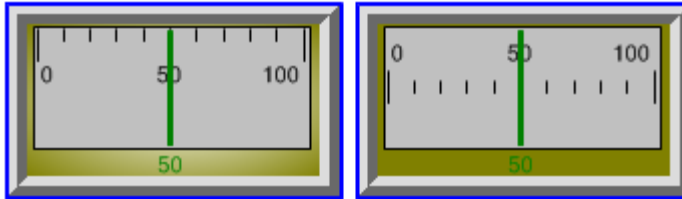


Property int SubTickerViewNumber

Specify how many sub tickers are drawn between two scale tickers.

Property DAS TickerAlignmentStyle TickerAlignment

Specify the scale ticker alignment type. There are two definitions, i.e.,



TAS_Border

TAS_Center

This property only takes effect for InnerScale = True.

Property Color TickerLineColor

Specify the color to draw the border of tickers and sub tickers.

Property Color ScaleTextColor

Specify the color to draw scale texts.

Property int TickerWidth

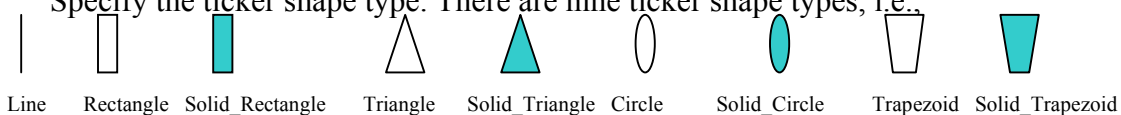
Specify the ticker width if the ticker shape is not “Line”.

Property Color TickerFillColor

Specify the color to fill the ticker if TickerShape is Solid_Rectangle, Solid_Triangle, Solid_Circle or Solid_Trapezoid.

Property DAS TickerShapeStyle TickerShape

Specify the ticker shape type. There are nine ticker shape types, i.e.,



Line

Rectangle

Solid_Rectangle

Triangle

Solid_Triangle

Circle

Solid_Circle

Trapezoid

Solid_Trapezoid

Property int SubTickerViewWidth

Specify the sub ticker width if the sub ticker shape is not “Line”.

Property Color SubTickerViewFillColor

Specify the color to fill the sub ticker if SubTickerViewShape is Solid_Rectangle, Solid_Triangle, Solid_Circle or Solid_Trapezoid.

Property DAS TickerShapeStyle SubTickerViewShape

Specify the sub ticker shape type.

Property int TickerViewLength



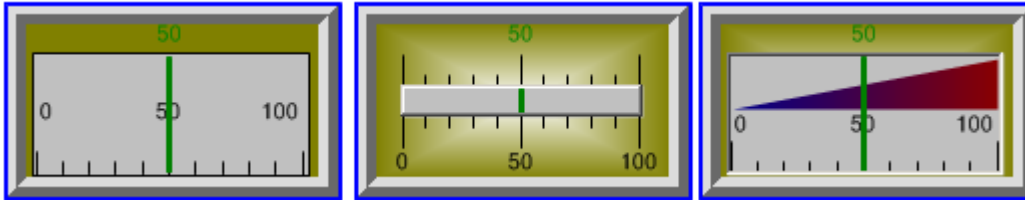
Specify the length of the tickers.

Property *int SubTickerLength*

Specify the length of the sub tickers.

Property *bool ScaleSideChanged*

Specify whether the scale is changed to another side of the ruler face. The followings are some samples that `ScaleSideChanged = True`.



Property *bool InnerScale*

Specify whether the scale is shown inside the ruler face.

Functions & Methods:

SetupMultiScaleRange(*int iIndex, MultiRange stRange*)

SetScaleRangeColor is used to configure the multi-color scale ring, where index defines a scale range identifier, *stRange* defines a scale range which is a structure as follows,

MultiRange

```
{  
    int iIndex;        // index of the scale range  
    bool bEnable;      // the scale range is enabled or not  
    Color cRangeColor; // the scale range color  
    double dblMax;     // the max value of the scale rang  
    double dblMin;     // the min value of the scale range  
}
```

The index can be 0 to 9 which means that the control can support up to 10 scale ranges.

DisableMultiScaleRange(*int iIndex*)

Disable a scale range. The disabled scale range is not invisible on the scale ring.

EnableMultiScaleRange(*int iIndex*)

Enable a scale range to make it visible on the scale ring.