Tell us about your PDF experience.

System.Windows.Forms.DataVisualization.Charting Namespace

Reference

Contains methods and properties for the Chart Windows forms control.

Classes

Annotation	Defines properties and methods common to all annotations. This is an abstract class.
AnnotationCollection	Stores chart annotation objects.
AnnotationGroup	Represents a group of annotations.
AnnotationPathPoint	Represents a path point of a PolylineAnnotation or PolygonAnnotation, which is stored in their `PathPoints` property.
AnnotationPathPointCollection	Represents a collection of PolylineAnnotation or PolygonAnnotation path points, which can be accessed using the `GraphicPathPoints` property.
AnnotationPositionChangingEventArgs	Provides data for the events in the Chart control that change the position of the annotation.
AnnotationSmartLabelStyle	Stores the smart label properties of an annotation.
AnovaResult	Represents the results of an ANOVA statistical test.
ArrowAnnotation	Represents an arrow annotation.
Axis	Provides methods and properties that manage axes in the Chart control.
AxisScaleBreakStyle	Represents a scale break on the Y-axis of the chart.
AxisScaleView	Represents a scaled view of the chart.
AxisScrollBar	Represents the scrollbar of a chart axis.
Border3DAnnotation	Represents an annotation with a 3D border.
BorderSkin	Represents the drawing properties of the 3D border skin.
CalloutAnnotation	Represents a callout annotation.
Chart	Serves as the root class of the Chart control.

ChartArea	Represents a chart area on the chart image.
ChartArea3DStyle	Manages the display attributes for three-dimensional (3D) chart areas.
ChartAreaCollection	Represents a collection of ChartArea objects.
ChartElement	Base class for all the chart elements. The ChartElement class represents the most basic element of the chart element hierarchy.
ChartElementCollection<T>	Represents the base class for all chart element collections.
ChartElementOutline	Represents an array of marker points and the outline path used for visual object selection in the chart.
ChartGraphics	Provides numerous conversion functions and exposes a GDI+ Graphics object.
ChartNamedElement	Represents the base class for most chart elements. Chart elements such as series, chart areas, and legends must have unique names, and reuse the unique name generation and validation logic provided by the ChartNamedElementCollection<T> class.
ChartNamedElement Collection<T>	Represents the base class for all collections of named chart elements. Performs name management and enforces the uniqueness of the names.
ChartPaintEventArgs	Provides data for the paint events of the Chart control.
ChartSerializer	Enables serialization of the Chart control.
Cursor	Represents a horizontal or vertical line that defines a position along an axis.
CursorEventArgs	Provides properties that are used with the following events in the root Chart object: CursorPositionChanged, CursorPositionChanging, SelectionRangeChanged and SelectionRangeChanging.
CustomizeLegendEventArgs	Provides data for the CustomizeLegend event of the Chart control.
CustomLabel	Represents a custom label for an axis.
CustomLabelsCollection	Represents a collection of CustomLabel objects.
CustomProperties	Represents the custom properties for the data visualization charting.
DataFormula	Provides properties and methods to prepare series data for technical analysis, apply formulas on the series data and prepare

	output data to be displayed as a chart. The actual calculations are performed in the StatisticFormula class.
DataManipulator	Provides method that performs data manipulation operations at run time. This class is exposed by the DataManipulator property of the root Chart object.
DataPoint	Represents a data point that is stored in the DataPointCollection class.
DataPointCollection	Represents a collection of DataPoint objects.
DataPointComparer	Compares data points.
DataPointCustomProperties	Defines custom properties that are common to data points, series and empty points.
ElementPosition	Represents the base class for many visual elements of the chart such as the legend, title, and chart areas. Defines the position of the chart element in relative coordinates, which range from (0,0) to (100,100).
EllipseAnnotation	Represents an ellipse annotation.
FormatNumberEventArgs	Represents the event arguments of the localized number-formatting event.
FTestResult	Stores results of the FTest statistical methods.
Grid	Provides methods and properties that manage the major and minor grids of Axis objects.
HitTestResult	Represents the results of a hit test.
HorizontalLineAnnotation	Represents a horizontal line annotation.
ImageAnnotation	Represents an image annotation.
LabelStyle	Provides properties that define the visual appearance of the axis labels.
Legend	Represents the legend for the chart image.
LegendCell	Represents a legend cell in a legend item.
LegendCellCollection	Stores legend cells in a LegendCellCollection object.
LegendCellColumn	Represents a cell column in a legend, which is used to extend the functionality of the default legend.
LegendCellColumnCollection	Stores LegendCellColumn objects, which are applied to default legend items.

LegendCollection	Represents a collection of Legend objects.
LegendItem	Represents a legend item, which is stored in the LegendItemsCollection class.
LegendItemsCollection	Represents a collection of LegendItem objects.
LineAnnotation	Represents a line annotation.
Margins	Represents the margins for various chart elements.
NamedImage	Represents named images, which are stored in the NamedImagesCollection.
NamedImagesCollection	Represents a collection of NamedImage objects.
Point3D	Represents the coordinates of a three-dimensional (3D) data point. This class is used when performing custom drawing with 3D charts.
PolygonAnnotation	Represents a polygon annotation.
PolylineAnnotation	Represents a polyline annotation.
PrintingManager	Represents the print manager for the chart.
RectangleAnnotation	Represents a rectangle annotation.
ScrollBarEventArgs	Used as a parameter object in the AxisScrollBarClicked of the root Chart object.
Series	Represents the data points and series attributes to store.
SeriesCollection	Represents a collection of Series objects.
SmartLabelStyle	Provides methods and properties used to perform intelligent, rule-based data point label positioning to resolve overlapping data point label issues.
StatisticFormula	Calculates statistical formulas.
StripLine	Represents the strip lines on a chart.
StripLinesCollection	Represents a collection of StripLine objects.
TextAnnotation	Represents a text annotation.
TickMark	Represents the major and minor tick marks along an axis.
Title	Represents a title for the entire chart image.
TitleCollection	Represents a collection of Title objects.
ToolTipEventArgs	Used as a parameter object in the GetToolTipText event of the

	root Chart object.
TTestResult	Stores the results of a T Test statistical formula.
VerticalLineAnnotation	Represents a vertical line annotation.
ViewEventArgs	Provides event data that is used as a parameter object in the AxisViewChanged and AxisViewChanging events of the root Chart object.
ZTestResult	Stores the results of a Z Test statistical formula.

Interfaces

IDataPointFilter	Exposes the FilterDataPoint(DataPoint, Series, Int32) method, which allows Series data to be custom filtered.

Enums

AntiAliasingStyles	Specifies the anti-aliasing options for the chart.
AreaAlignmentOrientations	Specifies the alignment orientation of a chart area.
AreaAlignmentStyles	Specifies the alignment styles of chart areas.
ArrowStyle	Specifies styles for an arrow annotation.
AxisArrowStyle	An enumeration of axis arrow styles.
AxisEnabled	Specifies the state of an axis.
AxisName	Specifies the name of an axis.
AxisType	Specifies the axis type for the X and Y-axes of a Series.
BorderSkinStyle	Specifies a style for the border skin of a chart.
BreakLineStyle	Specifies line styles for axis scale breaks.
CalloutStyle	Represents the available callout styles. Applicable only to callout annotations.
ChartColorPalette	Specifies a palette when setting a Palette property.
ChartDashStyle	Specifies a line style.
ChartElementType	Specifies a chart element type.
ChartHatchStyle	Specifies a hatching style for a chart element.

ChartImageAlignmentStyle	Specifies the image alignment for a chart background image.
ChartImageFormat	Specifies the image type of the chart.
ChartImageWrapMode	Specifies the drawing mode for a background image.
ChartValueType	Specifies value types for various chart properties.
CompareMethod	Specifies the type of comparison to be used when filtering data. This enumeration is used by the `Filter` method definitions in the DataManipulator class that take the `CompareMethod` argument.
DateRangeType	Specifies a date/time interval that determines the ranges that will be filtered when you invoke one of the `Filter` method definitions found in the DataManipulator class.
DateTimeIntervalType	Specifies an interval type.
Docking	Specifies where a chart element, such as a legend or title, will be docked on the chart.
FinancialFormula	Specifies a financial formula.
GradientStyle	Specifies a gradient style to be used for a chart element.
GridTickTypes	Specifies the options for displaying a grid line or a tick mark for a custom label.
IntervalAutoMode	Specifies the automatic axis interval mode.
IntervalType	Specifies an interval type.
LabelAlignmentStyles	Specifies a label alignment when SmartLabelStyle is used.
LabelAutoFitStyles	Specifies style changes that can automatically be made to a label when the LabelAutoFitStyle property is used.
LabelCalloutStyle	Specifies the type of leader line that should be drawn from the point to the label.
LabelMarkStyle	Specifies a mark for custom labels.
LabelOutsidePlotAreaStyle	Specifies the degree to which a label can be drawn outside a plotting area.
LegendCellColumnType	Specifies a legend cell column type.
LegendCellType	Specifies a legend cell type.
LegendImageStyle	Specifies the style to apply to the symbol for a legend item.
LegendItemOrder	Specifies the order in which items are added to the legend.

LegendSeparatorStyle	Specifies the styles for legend separator lines.
LegendStyle	Specifies the legend style.
LegendTableStyle	Specifies the legend table style.
LightStyle	Specifies a lighting style for a three-dimensional (3D) chart area.
LineAnchorCapStyle	Specifies the shape - that is, the anchor cap - to draw on the point end of the label callout line.
MarkerStyle	Specifies a style for markers.
PointSortOrder	Indicates whether the sort order for data points is ascending or descending.
ScrollBarButtonStyles	Specifies the type of button that will be displayed on the scrollbar.
ScrollBarButtonType	Specifies the button clicked by a user in the AxisScrollBarClicked event.
ScrollType	Specifies the scrolling that will occur when the one of the `Scroll` methods in the AxisScaleView class is called.
SerializationContents	Specifies the type of chart content to be serialized.
SerializationFormat	Represents the formatting type to use when the chart is serialized.
SeriesChartType	Specifies a chart type for a Series.
StartFromZero	Indicates whether an axis segment should start from zero when a scale break is used.
TextAntiAliasingQuality	Specifies the level of anti-aliasing quality to be used with the TextAntiAliasingQuality property.
TextOrientation	Specifies text orientation in a chart element.
TextStyle	Specifies text drawing styles.
TickMarkStyle	Specifies a tick mark style when setting the TickMarkStyle property of a TickMark object.

Annotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Defines properties and methods common to all annotations. This is an abstract class.

C#

```
public abstract class Annotation :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation

Derived System.Windows.Forms.DataVisualization.Charting.AnnotationGroup
System.Windows.Forms.DataVisualization.Charting.ArrowAnnotation
System.Windows.Forms.DataVisualization.Charting.ImageAnnotation
System.Windows.Forms.DataVisualization.Charting.LineAnnotation
System.Windows.Forms.DataVisualization.Charting.PolylineAnnotation
More...

Remarks

Annotations are commonly used to comment or elaborate on chart elements such as data points. Annotations can also be used to draw custom shapes.

By default, annotations are positioned using relative coordinates, with (0,0) representing the top-left corner, and (100,100) representing the bottom-right corner of the chart image. It is also possible to switch from this relative coordinate system to a system that uses axis values. With an axis coordinate system, X and Y, which represent the position of the top-left corner of an annotation, are set using X axis and Y axis values, instead of values that range from 0-100.

There are two ways to use axis values when you specify the position and size of an annotation:

- Set the AxisX, AxisY or both of these annotation properties to the AxisX and AxisY property values of a ChartArea object.
- Use the AnchorDataPoint property to anchor the annotation to a data point. In this case, its positioning is automatically calculated.

All annotations are derived from the Annotation class, which can be used to set attributes common to all Annotation objects, such as color, position, anchoring and so forth.

Constructors

Annotation()	Initializes a new instance of the Annotation class. This is the parameterless constructor and should not be used.

Properties

Alignment	Gets or sets the content alignment of an annotation.
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse.
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse.
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse.
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse.
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse.
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text.
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point.
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored.
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored.
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation.

AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation.
AnchorX	Gets or sets the X coordinate to which the annotation is anchored.
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored.
AnnotationGroup	Gets the group, if any, to which the annotation belongs.
AnnotationType	Gets or sets an annotation's type name.
AxisX	Gets or sets the X axis to which an annotation is attached.
AxisXName	Gets or sets the name of the X axis to which an annotation is attached.
AxisY	Gets or sets the Y axis to which an annotation is attached.
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to.
BackColor	Gets or sets the color of an annotation background.
BackGradientStyle	Gets or sets the gradient style of an annotation background.
BackHatchStyle	Gets or sets the hatching style of an annotation background.
BackSecondaryColor	Gets or sets the secondary color of an annotation background.
Bottom	Gets or sets the bottom boundary of an annotation position.
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped.
Font	Gets or sets the font to use for the text of an annotation.
ForeColor	Gets or sets the text color of an annotation.
Height	Gets or sets the height of an annotation.
IsSelected	Gets or sets a flag that determines if an annotation is selected.
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates.
LineColor	Gets or sets the line color of an annotation.
LineDashStyle	Gets or sets the line style of an annotation.
LineWidth	Gets or sets the line width of an annotation.

Name	Gets or sets the unique name of an annotation.
Right	Gets or sets the right boundary position of an annotation.
ShadowColor	Gets or sets the color of an annotation's shadow.
ShadowOffset	Gets or sets the size of an annotation's shadow.
SmartLabelStyle	Gets or sets the smart label style of an annotation.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	Gets or sets the text style of an annotation.
ToolTip	Gets or sets the tooltip text for an annotation.
Visible	Gets or sets a flag that specifies whether an annotation is visible.
Width	Gets or sets the width, in pixels, of an annotation.
X	Gets or sets the X coordinate of an annotation.
Y	Gets or sets the Y coordinate of the annotation.
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached.

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse.
BringToFront()	Brings an annotation to the front of all annotations.
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources.
EndPlacement()	Ends the end user placement of an annotation.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance.

	(Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size.
SendToBack()	Sends an annotation to the back of all the annotations.
SetAnchor(DataPoint)	Anchors an annotation to the specified data point.
SetAnchor(DataPoint, Data Point)	Anchors an annotation to two data points.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Annotation class. This is the parameterless constructor and should not be used.

C#

```
protected Annotation ();
```

Remarks

Annotation is an abstract class and cannot be created directly. To create a new annotation object, use constructors of classes such as LineAnnotation, RectangleAnnotation, and the like.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the content alignment of an annotation.

C#

```
public virtual System.Drawing.ContentAlignment Alignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment enumeration value that represents the content alignment.

Remarks

This property is used to align text for TextAnnotation, RectangleAnnotation, EllipseAnnotation and CalloutAnnotation objects, and to align a non-scaled image inside an ImageAnnotation object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AllowAnchorMoving Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse.

C#

```
public virtual bool AllowAnchorMoving { get; set; }
```

Property Value

Boolean

`true` if the annotation anchor may be moved; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AllowMoving Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse.

```
C#
public virtual bool AllowMoving { get; set; }
```

Property Value

Boolean

`true` if the annotation may be moved; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AllowPathEditing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse.

```
C#
public virtual bool AllowPathEditing { get; set; }
```

Property Value

Boolean

`true` if the polygon annotation's points may be moved; otherwise, `false`.

Remarks

This property needs to be set to `true` to free draw a polyline annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AllowResizing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse.

```
C#
public virtual bool AllowResizing { get; set; }
```

Property Value

Boolean

`true` if the annotation may be resized; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AllowSelecting Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse.

```
C#
public virtual bool AllowSelecting { get; set; }
```

Property Value

Boolean

`true` if the annotation may be selected; otherwise `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AllowTextEditing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text.

```
C#
public virtual bool AllowTextEditing { get; set; }
```

Property Value

Boolean

`true` if the annotation text may be edited; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnchorAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of an annotation to the anchor point.

C#

```
public virtual System.Drawing.ContentAlignment AnchorAlignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment enumeration value that represents the alignment of the annotation to the anchor point.

Remarks

The annotation must be anchored by either using the AnchorDataPoint property, or by setting the AnchorX and AnchorY properties. Its X and Y properties must be set to NaN.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnchorDataPoint Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the data point to which an annotation is anchored.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AnchorPointValueConverter))]
public virtual System.Windows.Forms.DataVisualization.Charting.DataPoint AnchorDataPoint { get; set; }
```

Property Value

DataPoint

A DataPoint object to which an annotation is anchored.

Attributes TypeConverterAttribute

Remarks

The annotation is anchored to the X and Y values of the specified data point, and automatically uses the same axes coordinates as the data point.

To automatically position the annotation relative to the anchor point, make sure its X and Y properties are set to NaN. The AnchorAlignment property can be used to change the automatic position alignment of the annotation to the anchor point. The AnchorOffsetX and AnchorOffsetY properties may be used to add extra spacing.

When you use this property, make sure the AnchorX and AnchorY properties are set to Double.NaN, as they have precedence.

Set this value to `null` to disable annotation anchoring to a data point.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnchorDataPointName Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of a data point to which an annotation is anchored.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public virtual string AnchorDataPointName { get; set; }
```

Property Value

String

A `string` value that represents the name of the data point that an annotation is anchored to.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnchorOffsetX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the offset from the anchor point for the X position of an annotation.

C#

```
public virtual double AnchorOffsetX { get; set; }
```

Property Value

Double

A `double` value that represents the offset of the X position from the anchor point.

Remarks

The annotation must be anchored using either the AnchorDataPoint or the AnchorX property, and its X property must be set to NaN.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnchorOffsetY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the offset from the anchor point for the Y position of an annotation.

C#

```
public virtual double AnchorOffsetY { get; set; }
```

Property Value

Double

A `double` value that represents the offset of the Y position from the anchor point.

Remarks

The annotation must be anchored using either the AnchorDataPoint or AnchorY property, and its Y property must be set to NaN.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnchorX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the X coordinate to which the annotation is anchored.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public virtual double AnchorX { get; set; }
```

Property Value

Double

A `double` value that represents the X coordinate an annotation is anchored to.

Attributes TypeConverterAttribute

Remarks

The annotation is anchored to the X coordinate and is specified in relative chart or axes coordinates, depending on the AxisX property value.

To automatically position an annotation relative to an anchor point, make sure its X property is set to NaN. The AnchorAlignment property can be used to change the automatic position alignment of the annotation to the anchor point. The AnchorOffsetX and AnchorOffsetY properties may be used to add extra spacing.

This property has a higher priority than the AnchorDataPoint property.

Set this value to NaN to disable annotation anchoring to the value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnchorY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Y coordinate to which the annotation is anchored.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public virtual double AnchorY { get; set; }
```

Property Value

Double

A `double` value that represents the Y coordinate an annotation is anchored to.

Attributes TypeConverterAttribute

Remarks

The annotation is anchored to the Y coordinate and is specified in relative chart or axes coordinates, depending on the AxisX property value.

To automatically position an annotation relative to an anchor point make sure its Y property is set to NaN. The AnchorAlignment property can be used to change the automatic position alignment of the annotation to the anchor point. The AnchorOffsetX and AnchorOffsetY properties may be used to add extra spacing.

This property has a higher priority than the AnchorDataPoint property.

Set this value to NaN to disable annotation anchoring to the value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnnotationGroup Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the group, if any, to which the annotation belongs.

C#

```
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.AnnotationGroup
AnnotationGroup { get; }
```

Property Value

AnnotationGroup

An AnnotationGroup object that represents the group which the annotation belongs to.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an annotation's type name.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public abstract string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

This property is used to get the name of each annotation type, for example, Line, Rectangle, Ellipse, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AxisX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the X axis to which an annotation is attached.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AnnotationAxisValueConverter))]
public virtual System.Windows.Forms.DataVisualization.Charting.Axis AxisX {
get; set; }
```

Property Value

Axis

An Axis object to which the annotation is attached.

Attributes TypeConverterAttribute

Remarks

When an annotation is attached to an axis, its X position is always specified in axes coordinates. To also define the size of an annotation in axes coordinates, make sure the IsSizeAlwaysRelative property is set to `false`.

Set this value to `null` to disable attachment to the axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AxisXName Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the X axis to which an annotation is attached.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public virtual string AxisXName { get; set; }
```

Property Value

String

A `string` value that represents the name of the X axis which an annotation is attached to.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AxisY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Y axis to which an annotation is attached.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AnnotationAxisValueConverter))]
public virtual System.Windows.Forms.DataVisualization.Charting.Axis AxisY {
get; set; }
```

Property Value

Axis

An Axis object to which this annotation is attached.

Attributes TypeConverterAttribute

Remarks

When an annotation is attached to an axis, its Y position is always specified in axes coordinates. To also define the size of an annotation in axes coordinates, make sure the IsSizeAlwaysRelative property is set to `false`.

Set to `null` to disable annotation attachment to an axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.AxisYName Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the Y axis which an annotation is attached to.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public virtual string AxisYName { get; set; }
```

Property Value

String

A `string` value that represents the name of the Y axis that an annotation is attached to.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of an annotation background.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value used to draw an annotation background.

Attributes TypeConverterAttribute

Remarks

This property does not apply to line-type annotations, image annotations and text annotations.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the gradient style of an annotation background.

C#

```
public virtual System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle value used to draw an annotation background.

Remarks

Two colors are used to draw the gradients, BackColor and BackSecondaryColor.

This property does not apply to line-type annotations, image annotations and text annotations.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatching style of an annotation background.

C#

```
public virtual
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle value used to draw the background.

Remarks

Two colors are used to draw the hatching, BackColor and BackSecondaryColor.

This property does not apply to line-type annotations, image annotations and text annotations.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color of an annotation background.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value used to draw an annotation background with hatching or gradient fill.

Attributes TypeConverterAttribute

Remarks

The secondary color is used with the BackColor when BackHatchStyle or BackGradientStyle is used.

This property does not apply to line-type annotations, image annotations and text annotations.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Bottom Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the bottom boundary of an annotation position.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public virtual double Bottom { get; set; }
```

Property Value

Double

A `double` value that represents the position of the bottom boundary of an annotation.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

To use axes coordinates, anchor an annotation to a data point using the AnchorDataPoint property, or set the annotation axes using the AxisX or AxisY properties and set the IsSizeAlwaysRelative property to `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.ClipToChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the chart area to which an annotation is clipped.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.LegendAreaNameConverter))]
public virtual string ClipToChartArea { get; set; }
```

Property Value

String

A `string` that represents the name of an existing chart area to which an annotation is clipped.

Attributes TypeConverterAttribute

Remarks

If the chart area name is specified, an annotation will only be drawn inside the plotting area of the chart area specified. All parts of the annotation outside of the plotting area will be clipped.

To disable chart area clipping, set the property to Empty.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font to use for the text of an annotation.

C#

```
public virtual System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object used to draw the annotation text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color of an annotation.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value used for the text color of an annotation.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Height Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the height of an annotation.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public virtual double Height { get; set; }
```

Property Value

Double

A `double` value that represents the height of an annotation.

Attributes TypeConverterAttribute

Remarks

The height of an annotation can be a negative value, in which case the annotation orientation is switched.

Annotation width can be in relative chart coordinates or in axes coordinates. Chart relative coordinates are used by default.

To use axes coordinates, anchor an annotation to a data point using the AnchorDataPoint property, or set the annotation axes using the AxisX or AxisY properties and set the IsSizeAlwaysRelative property to `false`.

Set the width to NaN to achieve automatic size calculation for annotations with text. The size will automatically be calculated based on the annotation text and font size.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.IsSelected Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines if an annotation is selected.

C#

```
[System.ComponentModel.Browsable(false)]
public virtual bool IsSelected { get; set; }
```

Property Value

Boolean

`true` if the annotation is selected; otherwise, `false`.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.IsSizeAlwaysRelative Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates.

C#

```
public virtual bool IsSizeAlwaysRelative { get; set; }
```

Property Value

Boolean

`true` if the Width and Height of an annotation are always expressed in relative chart coordinates; otherwise, `false`.

Remarks

The height and width of an annotation may be set in relative chart or in axes coordinates. Relative chart coordinates are the default.

To express the size of an annotation in axes coordinates, set the IsSizeAlwaysRelative property to `false`. Then anchor the annotation to a data point or set the AxisX and AxisY properties.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line color of an annotation.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color value used to draw an annotation's line.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line style of an annotation.

C#

```
public virtual
System.Windows.Forms.DataVisualization.Charting.ChartDashStyle LineDashStyle
{ get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle value used to draw the line style of an annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line width of an annotation.

C#

```
public virtual int LineWidth { get; set; }
```

Property Value

Int32

An `integer` value that defines the line width, in pixels, of an annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unique name of an annotation.

C#

```
[System.ComponentModel.Bindable(true)]
public override string Name { get; set; }
```

Property Value

String

A `string` that represents the unique name of an annotation.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Right Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the right boundary position of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public virtual double Right { get; set; }
```

Property Value

Double

A `double` value that represents the position of the right boundary of an annotation.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

To use axes coordinates, anchor an annotation to a data point using the AnchorDataPoint property, or set the annotation axes using the AxisX and AxisY properties and set the IsSizeAlwaysRelative property to `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.ShadowColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of an annotation's shadow.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color ShadowColor { get; set; }
```

Property Value

Color

A Color value used to draw the annotation shadow.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.ShadowOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of an annotation's shadow.

C#

```
public virtual int ShadowOffset { get; set; }
```

Property Value

Int32

An `integer` value that represents the size of the shadow.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.SmartLabelStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the smart label style of an annotation.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(true)]
public
System.Windows.Forms.DataVisualization.Charting.AnnotationSmartLabelStyle
SmartLabelStyle { get; set; }
```

Property Value

AnnotationSmartLabelStyle

An AnnotationSmartLabelStyle object that represents an annotation's smart labels style properties.

Attributes BindableAttribute, BrowsableAttribute

Remarks

Smart labels are used to prevent an annotation from overlapping the data point labels and other annotations.

> **Note**
>
> Data point labels must also have smart labels enabled.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.TextStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text style of an annotation.

C#

```
public virtual System.Windows.Forms.DataVisualization.Charting.TextStyle
TextStyle { get; set; }
```

Property Value

TextStyle

A TextStyle value used to draw the text of an annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip text for an annotation.

```
C#
public virtual string ToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip text for an annotation.

Remarks

Special keywords can be used in the text when an annotation is anchored to a data point using the AnchorDataPoint property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Visible Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether an annotation is visible.

C#

```
public virtual bool Visible { get; set; }
```

Property Value

Boolean

`true` if the annotation is visible; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Width Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width, in pixels, of an annotation.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public virtual double Width { get; set; }
```

Property Value

Double

A `double` value that represents the width of an annotation.

Attributes TypeConverterAttribute

Remarks

For more information, see the Height property description.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.X Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the X coordinate of an annotation.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public virtual double X { get; set; }
```

Property Value

Double

A `double` value that represents the X coordinate of an annotation's position.

Attributes TypeConverterAttribute

Remarks

The X coordinate of an annotation is specified in relative chart coordinates or in axes coordinates. Relative chart coordinates are used by default.

To use axes coordinates, anchor an annotation to a data point using the AnchorDataPoint property, or set the annotation axes using the AxisX and AxisY properties.

Set the X position to NaN to automatically calculate position when the annotation has been anchored using the AnchorDataPoint property or the AnchorX and AnchorY properties.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Y Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Y coordinate of the annotation.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public virtual double Y { get; set; }
```

Property Value

Double

A `double` value that represents the Y coordinate of an annotation's position.

Attributes TypeConverterAttribute

Remarks

The Y coordinate of an annotation is specified in relative chart coordinates or in axes coordinates. Relative chart coordinates are used by default.

To use axes coordinates, anchor an annotation to a data point using the AnchorDataPoint property, or set the annotation axes using the AxisX or AxisY properties.

Set the Y position to NaN to automatically calculate the position when the annotation has been anchored using the AnchorDataPoint property or the AnchorX and AnchorY properties.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.YAxisName Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the Y axis to which an annotation is attached.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public virtual string YAxisName { get; set; }
```

Property Value

String

A `string` value that represents the name of the Y axis which an annotation is attached to.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.BeginPlacement Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Begins the end user placement of an annotation using the mouse.

C#

```
public virtual void BeginPlacement ();
```

Remarks

When this method is called, the end user is allowed to place an annotation using the mouse.

Placement finishes when the end user specifies all required points, or when the EndPlacement method is called.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.BringToFront Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Brings an annotation to the front of all annotations.

C#

```
public virtual void BringToFront ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged resources and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.EndPlacement Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Ends the end user placement of an annotation.

C#

```
public virtual void EndPlacement ();
```

Remarks

This method ends an annotation placement operation previously started by a BeginPlacement method call.

Calling this method is not required, because placement will automatically end when an end user enters all required points. However, it is useful when an annotation placement operation needs to be aborted for some reason.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.ResizeToContent Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Resizes an annotation according to its content size.

C#

```
public virtual void ResizeToContent ();
```

Remarks

Sets the annotation width and height to fit the specified text.

This method applies to TextAnnotation, RectangleAnnotation, EllipseAnnotation and CalloutAnnotation objects only.

> **Note**
>
> This method requires that the annotation be added to the **AnnotationCollection** to function correctly.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.SendToBack Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sends an annotation to the back of all the annotations.

C#

```
public virtual void SendToBack ();
```

Remarks

Sends an annotation to the back of the all annotations.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Annotation.SetAnchor Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Anchors an annotation to one or more data points.

Overloads

SetAnchor(DataPoint)	Anchors an annotation to the specified data point.
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points.

SetAnchor(DataPoint)

Anchors an annotation to the specified data point.

C#

```
public void SetAnchor
(System.Windows.Forms.DataVisualization.Charting.DataPoint dataPoint);
```

Parameters

`dataPoint` DataPoint

The DataPoint to which the annotation will be anchored.

Remarks

Anchors an annotation to the specified data point.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SetAnchor(DataPoint, DataPoint)

Anchors an annotation to two data points.

C#

```
public void SetAnchor
(System.Windows.Forms.DataVisualization.Charting.DataPoint dataPoint1,
System.Windows.Forms.DataVisualization.Charting.DataPoint dataPoint2);
```

Parameters

`dataPoint1` DataPoint

The first anchor DataPoint.

`dataPoint2` DataPoint

The second anchor DataPoint.

Remarks

Anchors an annotation's top-left and bottom-right corners to the specified data points.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Stores chart annotation objects.

C#

```
public class AnnotationCollection :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<
System.Windows.Forms.DataVisualization.Charting.Annotation>
```

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<Annotation> → AnnotationCollection

Remarks

All chart annotations are stored in this collection.

This class includes methods to add, insert, iterate through and remove annotations from the collection.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Item[String]	Gets or sets the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)
NamePrefix	Gets the name prefix that is used to create unique chart element

	names. (Inherited from ChartNamedElementCollection<T>)

Methods

Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds an annotation in the collection by name.
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(String)	Finds the index of the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>.

	(Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index. (Inherited from ChartNamedElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection. (Inherited from ChartNamedElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection. (Inherited from ChartNamedElementCollection<T>)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection. (Inherited from ChartNamedElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element. (Inherited from ChartNamedElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array,	Copies the elements of the ICollection to an Array, starting at a

Int32)	particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T>

	object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a

	transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a

	condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>,	Groups the elements of a sequence according to a specified key selector function and

IEqualityComparer<TKey>)	compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>,	Produces the set intersection of two

IEnumerable<TSource>)	sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.

Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Nullable<Decimal>>)	of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>,	Projects each element of a sequence into a new form.

Func<TSource,TResult>)	
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this

	method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.

Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.

ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source

	collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationCollection.FindByName(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Finds an annotation in the collection by name.

C#

```
public override System.Windows.Forms.DataVisualization.Charting.Annotation
FindByName (string name);
```

Parameters

name String

The name of the annotation to find.

Returns

Annotation

An Annotation object, or `null` if the object does not exist.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a group of annotations.

C#

```
public class AnnotationGroup :
System.Windows.Forms.DataVisualization.Charting.Annotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → AnnotationGroup

Remarks

The AnnotationGroup class represents a collection of annotations. This class can be used to manipulate annotations relative to each other.

Constructors

AnnotationGroup()	Initializes a new instance of the AnnotationGroup class.

Properties

Alignment	Gets or sets the content alignment of an annotation.
AllowAnchorMoving	Gets or sets a flag that specifies whether an annotation anchor may be moved with a mouse by the end user.
AllowMoving	Gets or sets a flag that specifies whether an annotation may be moved with a mouse by the end user.
AllowPathEditing	Gets or sets a flag that specifies whether the points of a polygon annotation may be moved with a mouse by the end user.

AllowResizing	Gets or sets a flag that specifies whether an annotation may be resized with a mouse by the end user.
AllowSelecting	Gets or sets a flag that specifies whether an annotation may be selected with a mouse by the end user.
AllowTextEditing	Gets or sets a flag that specifies whether an annotation's text may be edited when the end user double clicks the text.
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point. (Inherited from Annotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
Annotations	Gets the collection of annotations in the group.
AnnotationType	Gets the annotation type.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is

	attached to. (Inherited from Annotation)
BackColor	Gets or sets the background color of an annotation.
BackGradientStyle	Gets or sets the background gradient style of an annotation.
BackHatchStyle	Gets or sets the hatch style of an annotation background.
BackSecondaryColor	Gets or sets the secondary color for the background of an annotation.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which the annotation is clipped.
Font	Gets or sets the font to use for the text of an annotation.
ForeColor	Gets or sets the text color for an annotation.
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected.
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates.
LineColor	Gets or sets the line color of the annotation.
LineDashStyle	Gets or sets the line style of the annotation.
LineWidth	Gets or sets the line width, in pixels, of the annotation.
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of the annotation's shadow.
ShadowOffset	Gets or sets the size of the annotation's shadow.
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	Gets or sets the text style of an annotation.

ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that indicates whether the annotation is visible.
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
EndPlacement()	Ends the end user placement of an annotation. (Inherited from Annotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size.

	(Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, Data Point)	Anchors an annotation to two data points. (Inherited from Annotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AnnotationGroup class.

C#

```
public AnnotationGroup ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the content alignment of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.ContentAlignment Alignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment value that represents the content alignment of an annotation.

Attributes BrowsableAttribute

Remarks

This property is used to align text for TextAnnotation, RectangleAnnotation, EllipseAnnotation and CalloutAnnotation objects, and to align a nonscaled image inside an ImageAnnotation object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.AllowAnchorMoving Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether an annotation anchor may be moved with a mouse by the end user.

C#

```
public override bool AllowAnchorMoving { get; set; }
```

Property Value

Boolean

`true` if the annotation anchor may be moved; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.AllowMoving Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether an annotation may be moved with a mouse by the end user.

C#

```
public override bool AllowMoving { get; set; }
```

Property Value

Boolean

`true` if the annotation may be moved; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.AllowPathEditing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the points of a polygon annotation may be moved with a mouse by the end user.

C#

```
public override bool AllowPathEditing { get; set; }
```

Property Value

Boolean

`true` if the points of the polygon annotation may be moved; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.AllowResizing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether an annotation may be resized with a mouse by the end user.

```
C#

public override bool AllowResizing { get; set; }
```

Property Value

Boolean

`true` if the annotation may be resized; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.AllowSelecting Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether an annotation may be selected with a mouse by the end user.

```
C#

public override bool AllowSelecting { get; set; }
```

Property Value

Boolean

`true` if the annotation may be selected; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.AllowTextEditing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether an annotation's text may be edited when the end user double clicks the text.

C#

```
public override bool AllowTextEditing { get; set; }
```

Property Value

Boolean

`true` if the annotation text may be edited; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.Annotations Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the collection of annotations in the group.

C#

```
public System.Windows.Forms.DataVisualization.Charting.AnnotationCollection
Annotations { get; }
```

Property Value

AnnotationCollection

An AnnotationCollection object.

Remarks

Note that the coordinates of all annotations in the group are relative to the group annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

This property is used to get the name of each annotation type, for example. Line, Rectangle, Ellipse, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value used for the background of an annotation.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background gradient style of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle value used to draw the background of an annotation.

Attributes BrowsableAttribute

Remarks

Two colors are used to draw the gradient, BackColor and BackSecondaryColor.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatch style of an annotation background.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle value used to draw the background of an annotation.

Attributes BrowsableAttribute

Remarks

Two colors are used to draw the hatching, BackColor and BackSecondaryColor.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color for the background of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value used to draw the background with hatching or gradient fill.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

A secondary color is used with BackColor when BackHatchStyle or BackGradientStyle are used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.ClipToChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the chart area to which the annotation is clipped.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.LegendAreaNameConverter))]
public override string ClipToChartArea { get; set; }
```

Property Value

String

A `string` that represents the name of an existing chart area.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

If a chart area name is specified, the annotation will only be drawn inside the plotting area of that chart area. All parts of the annotation outside of the plotting area will be clipped.

To disable chart area clipping, set this property to NaN or an empty string.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font to use for the text of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object used for the text of an annotation.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color for an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value used for the text color of an annotation.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.IsSelected Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines if an annotation is selected.

C#

```
[System.ComponentModel.Browsable(false)]
public override bool IsSelected { get; set; }
```

Property Value

Boolean

`true` if the annotation is selected; otherwise, `false`.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.IsSizeAlwaysRelative Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates.

```
C#
public override bool IsSizeAlwaysRelative { get; set; }
```

Property Value

Boolean

`true` if the Width and Height of an annotation are always in chart relative coordinates; otherwise, `false`.

Remarks

The height and width of an annotation may be set in relative chart or in axes coordinates. By default, relative chart coordinates are used.

To use axes coordinates, set the IsSizeAlwaysRelative property to `false`. Then either anchor the annotation to a data point or set the AxisX or AxisY properties.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line color of the annotation.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color value used to draw an annotation line.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line style of the annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartDashStyle LineDashStyle
{ get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle value used to draw an annotation line.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line width, in pixels, of the annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public override int LineWidth { get; set; }
```

Property Value

Int32

An `integer` value defining the width, in pixels, of an annotation line.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.ShadowColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the annotation's shadow.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color ShadowColor { get; set; }
```

Property Value

Color

A Color value used to draw the shadow of an annotation.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.ShadowOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the annotation's shadow.

C#

```
[System.ComponentModel.Browsable(false)]
public override int ShadowOffset { get; set; }
```

Property Value

Int32

An `integer` value that represents the offset between an annotation and its shadow.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.TextStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text style of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Windows.Forms.DataVisualization.Charting.TextStyle
TextStyle { get; set; }
```

Property Value

TextStyle

A TextStyle enumeration value used to draw the text of an annotation.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.Visible Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the annotation is visible.

C#

```
public override bool Visible { get; set; }
```

Property Value

Boolean

`true` if the annotation is visible; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationGroup.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPoint Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a path point of a PolylineAnnotation or PolygonAnnotation, which is stored in their `PathPoints` property.

C#

```
public class AnnotationPathPoint :
System.Windows.Forms.DataVisualization.Charting.ChartElement
```

Inheritance Object → ChartElement → AnnotationPathPoint

Remarks

At run time, use the PolylineAnnotation.GraphicsPath or PolygonAnnotation.GraphicsPath property to set the path of a polyline or polygon.

Constructors

AnnotationPathPoint()	Initializes a new instance of the AnnotationPathPoint class.
AnnotationPathPoint(Single, Single)	Initializes a new instance of the AnnotationPathPoint class with the specified `x` and `y` values.
AnnotationPathPoint(Single, Single, Byte)	Initializes a new instance of the AnnotationPathPoint class with the specified `x`, `y` and point `type` values.

Properties

Name	Gets the name for the path point of an annotation.
PointType	Gets or sets the type for the path point of an annotation.
Tag	Gets or sets an object associated with this chart element.

	(Inherited from ChartElement)
X	Gets or sets the X coordinate for the path point of an annotation.
Y	Gets or sets the Y coordinate for the path point of an annotation.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources. (Inherited from ChartElement)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPoint Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AnnotationPathPoint class.

Overloads

AnnotationPathPoint()	Initializes a new instance of the AnnotationPathPoint class.
AnnotationPathPoint(Single, Single)	Initializes a new instance of the AnnotationPathPoint class with the specified `x` and `y` values.
AnnotationPathPoint(Single, Single, Byte)	Initializes a new instance of the AnnotationPathPoint class with the specified `x`, `y` and point `type` values.

AnnotationPathPoint()

Initializes a new instance of the AnnotationPathPoint class.

C#

```
public AnnotationPathPoint ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPoint(Single, Single)

Initializes a new instance of the AnnotationPathPoint class with the specified `x` and `y` values.

```
C#
public AnnotationPathPoint (float x, float y);
```

Parameters

x Single

Point X-value.

y Single

Point Y-value.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPoint(Single, Single, Byte)

Initializes a new instance of the AnnotationPathPoint class with the specified `x`, `y` and point `type` values.

```
C#
public AnnotationPathPoint (float x, float y, byte type);
```

Parameters

x Single

Point X-value.

y Single

Point Y-value.

`type` Byte

Point type.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPoint.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the name for the path point of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public string Name { get; }
```

Property Value

String

A `string` value that represents the name of the path point.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPoint.PointType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the type for the path point of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public byte PointType { get; set; }
```

Property Value

Byte

A `byte` value.

Attributes BrowsableAttribute

Remarks

See the PathPointType enumeration for more details.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPoint.X Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the X coordinate for the path point of an annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public float X { get; set; }
```

Property Value

Single

A `float` that represents the X value.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPoint.Y Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Y coordinate for the path point of an annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public float Y { get; set; }
```

Property Value

Single

A `float` that represents the Y value.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPointCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of PolylineAnnotation or PolygonAnnotation path points, which can be accessed using the `GraphicPathPoints` property.

C#

```
public class AnnotationPathPointCollection :
System.Windows.Forms.DataVisualization.Charting.ChartElementCollection<Syste
m.Windows.Forms.DataVisualization.Charting.AnnotationPathPoint>
```

Inheritance Object → Collection<T> → ChartElementCollection<AnnotationPathPoint> → AnnotationPathPointCollection

Remarks

This collection is used at design time only, and employs serialization to expose the shape of a polyline or polygon annotation, using their respective `GraphicPathPoints` collection properties.

At run time, use the `GraphicsPath` property to set the path of a PolylineAnnotation or PolygonAnnotation.

Constructors

AnnotationPathPointCollection(PolylineAnnotation)	Initializes a new instance of the AnnotationPathPointCollection class with the specified polyline annotation.

Properties

Count	Gets the number of elements actually contained in the

	Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)

Methods

Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)

Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts an element into the Collection<T> at the specified index location. (Inherited from ChartElementCollection<T>)
Invalidate()	Forces the invalidation of the chart element.
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the element at the specified index location of the Collection<T>. (Inherited from ChartElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the element at the specified index location. (Inherited from ChartElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)

ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.

Aggregate<TSource,TAccumulate> (IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult> (IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource> (IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by

	invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.

Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>,	Groups the elements of a sequence according to a key selector function. The keys are

Func<TSource,TElement>, IEquality Comparer<TKey>)	compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>,	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.

Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Nullable<Double>>)	of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum

	nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.

SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining

	elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>,	Computes the sum of the sequence of

Func<TSource,Nullable<Single>>)	nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.

ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source

	collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPointCollection(PolylineAnnotation) Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AnnotationPathPointCollection class with the specified polyline annotation.

C#

```
public AnnotationPathPointCollection
(System.Windows.Forms.DataVisualization.Charting.PolylineAnnotation
annotation);
```

Parameters

`annotation` PolylineAnnotation

A polyline annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPointCollection.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPathPointCollection.Invalidate Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Forces the invalidation of the chart element.

C#

```
public override void Invalidate ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides data for the events in the Chart control that change the position of the annotation.

C#

```
public class AnnotationPositionChangingEventArgs : EventArgs
```

Inheritance Object → EventArgs → AnnotationPositionChangingEventArgs

Remarks

The AnnotationPositionChangingEventArgs class is used for the two annotation-positioning events, AnnotationPositionChanging and AnnotationPositionChanged, in the Chart control.

The Annotation property gets the annotation object that is being positioned. This class also contains properties that retrieve the new location, width, height and anchor location of the annotation that is repositioned.

Constructors

AnnotationPositionChangingEventArgs()	Initializes a new instance of the AnnotationPositionChangingEventArgs class.

Properties

Annotation	Gets or sets the Annotation object.
NewAnchorLocation	Gets or sets the new anchor location of the annotation.

NewAnchorLocationX	Gets or sets the new annotation anchor point X location.
NewAnchorLocationY	Gets or sets the new annotation anchor point Y location.
NewLocationX	Gets or sets the new X coordinate location of the annotation.
NewLocationY	Gets or sets the new Y coordinate of the annotation location.
NewPosition	Gets or sets the new location and size of the annotation.
NewSizeHeight	Gets or sets the new height of the annotation.
NewSizeWidth	Gets or sets the new width of the annotation.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AnnotationPositionChangingEventArgs class.

C#

```
public AnnotationPositionChangingEventArgs ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.Annotation Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Annotation object.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Annotation Annotation
{ get; set; }
```

Property Value

Annotation

The Annotation object.

Remarks

The Annotation property allows the developer to set Annotation properties during the AnnotationPositionChanging and AnnotationPositionChanged events of the Chart control.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.NewAnchorLocation Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the new anchor location of the annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public System.Drawing.PointF NewAnchorLocation { get; set; }
```

Property Value

PointF

A PointF object that represents the new anchor position.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.NewAnchorLocationX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the new annotation anchor point X location.

C#

```
public double NewAnchorLocationX { get; set; }
```

Property Value

Double

A `double` value that represents the new annotation anchor point X location.

Remarks

When the AnnotationPositionChanging and AnnotationPositionChanging events are raised, the NewAnchorLocationX property determines the new X coordinate of the annotation. This information is useful when the annotation is newly anchored to a data point.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.NewAnchorLocationY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the new annotation anchor point Y location.

C#

```
public double NewAnchorLocationY { get; set; }
```

Property Value

Double

A `double` value representing the new annotation anchor point Y location.

Remarks

When the AnnotationPositionChanging and AnnotationPositionChanged events are raised, the NewAnchorLocationY property determines the new Y coordinate of the annotation. This information is useful when the annotation is newly anchored to a data point.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.NewLocationX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the new X coordinate location of the annotation.

C#

```
public double NewLocationX { get; set; }
```

Property Value

Double

A `double` value that represents the new X coordinate location of the annotation.

Remarks

When the AnnotationPositionChanging and AnnotationPositionChanged events are raised, the NewLocationX property defines the X-coordinate location of the annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.NewLocationY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the new Y coordinate of the annotation location.

C#

```
public double NewLocationY { get; set; }
```

Property Value

Double

A `double` value that represents the new Y coordinate of the annotation location.

Remarks

When the AnnotationPositionChanging and AnnotationPositionChanged events are raised, the NewLocationY property defines the Y-coordinate location of the annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.NewPosition Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the new location and size of the annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public System.Drawing.RectangleF NewPosition { get; set; }
```

Property Value

RectangleF

A RectangleF object.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.NewSizeHeight Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the new height of the annotation.

C#

```
public double NewSizeHeight { get; set; }
```

Property Value

Double

A `double` value that represents the new height of the annotation.

Remarks

When the AnnotationPositionChanging and AnnotationPositionChanged events are raised, use the NewSizeHeight property to determine the new height of the annotation. This information can be used when you resize the annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationPositionChangingEventArgs.NewSizeWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the new width of the annotation.

C#

```
public double NewSizeWidth { get; set; }
```

Property Value

Double

A `double` value that represents the new width of the annotation.

Remarks

When the AnnotationPositionChanging and AnnotationPositionChanged events are raised, use the NewSizeWidth property to determine the new width of the annotation. This information can be used when you resize the annotation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Stores the smart label properties of an annotation.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public class AnnotationSmartLabelStyle :
System.Windows.Forms.DataVisualization.Charting.SmartLabelStyle
```

Inheritance Object → SmartLabelStyle → AnnotationSmartLabelStyle

Attributes TypeConverterAttribute

Remarks

This class is derived from the SmartLabelStyle class used for Series objects.

Constructors

AnnotationSmartLabelStyle()	Initializes a new instance of the AnnotationSmartLabelStyle class.
AnnotationSmartLabelStyle(Object)	Initializes a new instance of the AnnotationSmartLabelStyle class with the specified chart element.

Properties

AllowOutsidePlotArea	Gets or sets a flag that specifies whether a SmartLabelStyle object can be drawn outside the plotting area. (Inherited from SmartLabelStyle)
CalloutBackColor	Gets or sets the back color of the label callout. This property is only applicable when the CalloutStyle property is set to a value

	of Box.
CalloutLineAnchorCapStyle	Gets or sets the anchor cap style for the label callout line.
CalloutLineColor	Gets or sets the color of the label callout line.
CalloutLineDashStyle	Gets or sets the style of the label callout line.
CalloutLineWidth	Gets or sets the width of the label callout line.
CalloutStyle	Gets or sets the callout style of the repositioned smart labels.
Enabled	Gets or sets a flag that indicates whether a SmartLabelStyle algorithm is enabled. (Inherited from SmartLabelStyle)
IsMarkerOverlappingAllowed	Gets or sets a flag that specifies whether the point labels are allowed to overlap a point marker. (Inherited from SmartLabelStyle)
IsOverlappedHidden	Gets or sets a flag that indicates whether overlapped labels that cannot be repositioned will be hidden. (Inherited from SmartLabelStyle)
MaxMovingDistance	Gets or sets the maximum distance, in pixels, that the overlapped SmartLabelStyle object can be moved from the marker. (Inherited from SmartLabelStyle)
MinMovingDistance	Gets or sets the minimum distance, in pixels, that the overlapped SmartLabelStyle object can be moved from the marker. (Inherited from SmartLabelStyle)
MovingDirection	Gets or sets the direction(s) in which the overlapped SmartLabelStyle object is allowed to be moved. (Inherited from SmartLabelStyle)

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)

ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AnnotationSmartLabelStyle class.

Overloads

AnnotationSmartLabelStyle()	Initializes a new instance of the AnnotationSmartLabelStyle class.
AnnotationSmartLabelStyle(Object)	Initializes a new instance of the AnnotationSmartLabelStyle class with the specified chart element.

AnnotationSmartLabelStyle()

Initializes a new instance of the AnnotationSmartLabelStyle class.

C#

```
public AnnotationSmartLabelStyle ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle(Object)

Initializes a new instance of the AnnotationSmartLabelStyle class with the specified chart element.

```
C#

public AnnotationSmartLabelStyle (object chartElement);
```

Parameters

`chartElement` Object

Chart element that this style belongs to.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle.CalloutBackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the back color of the label callout. This property is only applicable when the CalloutStyle property is set to a value of Box.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color CalloutBackColor { get; set; }
```

Property Value

Color

A Color object.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is for internal use and is hidden at design time and run time.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle.CalloutLineAnchorCapStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the anchor cap style for the label callout line.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle
CalloutLineAnchorCapStyle { get; set; }
```

Property Value

LineAnchorCapStyle

A CalloutLineAnchorCapStyle object.

Attributes BrowsableAttribute

Remarks

This method is for internal use and is hidden at design and run time.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle.CalloutLineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the label callout line.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color CalloutLineColor { get; set; }
```

Property Value

Color

A Color object.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is for internal use and is hidden at design time and run time.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle.CalloutLineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the label callout line.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
CalloutLineDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle object.

Attributes BrowsableAttribute

Remarks

This property is for internal use and is hidden at design time and run time.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle.CalloutLineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of the label callout line.

C#

```
[System.ComponentModel.Browsable(false)]
public override int CalloutLineWidth { get; set; }
```

Property Value

Int32

The width of the callout line.

Attributes BrowsableAttribute

Remarks

This property is for internal use and is hidden at design time and run time.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnnotationSmartLabelStyle.CalloutStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the callout style of the repositioned smart labels.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.LabelCalloutStyle
CalloutStyle { get; set; }
```

Property Value

LabelCalloutStyle

A LabelCalloutStyle object.

Attributes BrowsableAttribute

Remarks

This property is for internal use and is hidden at design time and run time.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the results of an ANOVA statistical test.

C#

```
public class AnovaResult
```

Inheritance Object → AnovaResult

Constructors

AnovaResult()	Initializes a new instance of the AnovaResult class.

Properties

DegreeOfFreedomBetween Groups	Gets the degree of freedom between groups.
DegreeOfFreedomTotal	Gets the total degree of freedom.
DegreeOfFreedomWithin Groups	Gets the degree of freedom within groups.
FCriticalValue	Gets the F critical value.
FRatio	Gets the F ratio.
MeanSquareVarianceBetween Groups	Gets the mean square variance between groups.
MeanSquareVarianceWithin Groups	Gets the mean square variance within groups.
SumOfSquaresBetween Groups	Gets the sum of the squares between groups.

SumOfSquaresTotal	Gets the total sum of the squares.
SumOfSquaresWithinGroups	Gets the sum of the squares within groups.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AnovaResult class.

C#

```
public AnovaResult ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.DegreeOfFreedomBetween Groups Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the degree of freedom between groups.

C#

```
public double DegreeOfFreedomBetweenGroups { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.DegreeOfFreedomTotal Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the total degree of freedom.

C#

```
public double DegreeOfFreedomTotal { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.DegreeOfFreedomWithinGroups Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the degree of freedom within groups.

C#

```
public double DegreeOfFreedomWithinGroups { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.FCriticalValue Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the F critical value.

C#

```
public double FCriticalValue { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.FRatio Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the F ratio.

C#

```
public double FRatio { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.MeanSquareVarianceBetweenGroups Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the mean square variance between groups.

C#

```
public double MeanSquareVarianceBetweenGroups { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.MeanSquareVarianceWithinGroups Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the mean square variance within groups.

C#

```
public double MeanSquareVarianceWithinGroups { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.SumOfSquaresBetween Groups Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the sum of the squares between groups.

C#

```
public double SumOfSquaresBetweenGroups { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.SumOfSquaresTotal Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the total sum of the squares.

C#

```
public double SumOfSquaresTotal { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AnovaResult.SumOfSquaresWithinGroups Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the sum of the squares within groups.

C#

```
public double SumOfSquaresWithinGroups { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AntiAliasingStyles Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the anti-aliasing options for the chart.

This enumeration supports a bitwise combination of its member values.

C#

```
[System.Flags]
public enum AntiAliasingStyles
```

Inheritance Object → ValueType → Enum → AntiAliasingStyles

Attributes FlagsAttribute

Fields

All	3	Use anti-aliasing for text and graphics.
Graphics	2	Use anti-aliasing when drawing graphics primitives such as circles or rectangles.
None	0	Do not use anti-aliasing.
Text	1	Use anti-aliasing when drawing text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AreaAlignmentOrientations Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the alignment orientation of a chart area.

This enumeration supports a bitwise combination of its member values.

C#

```
[System.Flags]
public enum AreaAlignmentOrientations
```

Inheritance Object → ValueType → Enum → AreaAlignmentOrientations

Attributes FlagsAttribute

Fields

All	3	Chart areas are aligned horizontally and vertically.
Horizontal	2	Chart areas are aligned horizontally.
None	0	Chart areas are not automatically aligned.
Vertical	1	Chart areas are aligned vertically.

Remarks

The AreaAlignmentOrientations enumeration represents the physical orientation, such as vertical or horizontal, of a chart area.

In contrast, the AreaAlignmentStyles enumeration represents the chart elements, such as the cursor or inner plot position, which can be used to align a chart area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AreaAlignmentStyles Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the alignment styles of chart areas.

This enumeration supports a bitwise combination of its member values.

C#

```
[System.Flags]
public enum AreaAlignmentStyles
```

Inheritance Object → ValueType → Enum → AreaAlignmentStyles

Attributes FlagsAttribute

Fields

All	15	Chart areas are aligned using all values.
AxesView	4	Chart areas are aligned by their data view.
Cursor	8	Chart areas are aligned using their cursors.
None	0	Chart areas are not automatically aligned.
PlotPosition	2	Chart areas are aligned by their inner plot position.
Position	1	Chart areas are aligned by the chart area position.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ArrowAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents an arrow annotation.

C#

```
public class ArrowAnnotation :
System.Windows.Forms.DataVisualization.Charting.Annotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → ArrowAnnotation

Remarks

Arrow annotations can be used to connect to points on the chart or to highlight a single chart area.

Different arrow styles and sizes can be applied to the annotation.

Constructors

ArrowAnnotation()	Initializes a new instance of the ArrowAnnotation class.

Properties

Alignment	Gets or sets the content alignment of an annotation. (Inherited from Annotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse.

	(Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the position alignment of the annotation to the anchor point.
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the name of the annotation type. For internal use, should not be used.
ArrowSize	Gets or sets an arrow size, in pixels.

ArrowStyle	Gets or sets the arrow style.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	Gets or sets the color of an annotation background. (Inherited from Annotation)
BackGradientStyle	Gets or sets the gradient style of an annotation background. (Inherited from Annotation)
BackHatchStyle	Gets or sets the hatching style of an annotation background. (Inherited from Annotation)
BackSecondaryColor	Gets or sets the secondary color of an annotation background. (Inherited from Annotation)
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
Font	Gets or sets the font to use for the text of an annotation. (Inherited from Annotation)
ForeColor	Gets or sets the text color of an annotation. (Inherited from Annotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	Gets or sets the line color of an annotation.

	(Inherited from Annotation)
LineDashStyle	Gets or sets the line style of an annotation. (Inherited from Annotation)
LineWidth	Gets or sets the line width of an annotation. (Inherited from Annotation)
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	Gets or sets the text style of an annotation. (Inherited from Annotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends the end user placement of an annotation. (Inherited from Annotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ArrowAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ArrowAnnotation class.

C#

```
public ArrowAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ArrowAnnotation.AnchorAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the position alignment of the annotation to the anchor point.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.ContentAlignment AnchorAlignment { get; set;
}
```

Property Value

ContentAlignment

A ContentAlignment enumeration value that represents the alignment of the annotation to the anchor point.

Attributes BrowsableAttribute

Remarks

The annotation must be anchored using either AnchorDataPoint, or the AnchorX and AnchorY properties.

Its X and Y properties must be set to NaN.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ArrowAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the name of the annotation type. For internal use, should not be used.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

A `string` value that represents the name of the annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

Gets the name of each annotation type such as Line, Rectangle, Ellipse, and so forth.

This property is for internal use and is hidden at design time and run time.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ArrowAnnotation.ArrowSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an arrow size, in pixels.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual int ArrowSize { get; set; }
```

Property Value

Int32

An `integer` value that represents arrow size - that is, its thickness - in pixels.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ArrowAnnotation.ArrowStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the arrow style.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual System.Windows.Forms.DataVisualization.Charting.ArrowStyle
ArrowStyle { get; set; }
```

Property Value

ArrowStyle

The ArrowStyle of an annotation.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ArrowStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies styles for an arrow annotation.

C#

```
public enum ArrowStyle
```

Inheritance Object → ValueType → Enum → ArrowStyle

Fields

DoubleArrow	1	An arrow that points in two directions.
Simple	0	A simple one-pointed arrow.
Tailed	2	An arrow with a tail.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides methods and properties that manage axes in the Chart control.

C#

```
public class Axis :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → Axis

Remarks

This class encapsulates all the functionality for axes in the Chart control, and is exposed by the AxisX, AxisY, AxisX2 and AxisY2 properties of ChartArea objects.

For all charts - except bar, stacked bar, pie and doughnut - the primary and secondary axes are as follows:

Axis Type	Description
Primary X-Axis	Bottom horizontal axis.
Secondary X-Axis	Top horizontal axis.
Primary Y-Axis	Left vertical axis.
Secondary Y-Axis	Right vertical axis.

Bar and stacked-bar charts have their axes rotated 90 degrees clockwise, so the primary X axis for these two chart types is the left-vertical axis.

Note There can be only one set of axes - two vertical and two horizontal - for each ChartArea object.

The Axis class also contains several methods that can be used to convert axis values to absolute pixel coordinates, which are useful for 2D custom drawing, or to relative coordinates, which are useful for 3D custom drawing. Absolute and relative coordinates can also be converted to axis values.

The following are the most important axis elements exposed by the Axis class:

Axis Element	Description
Axis Labels	The labels of an axis, and their associated properties.
Grid Lines	The major and minor grid lines of an axis.
Tick Marks	The major and minor tick marks of an axis.
Strip Lines	Strip lines.

Constructors

Axis()	Initializes a new instance of the Axis class.
Axis(ChartArea, AxisName)	Initializes a new instance of the Axis class with the specified `chartArea` and `axisTypeName` values.

Properties

ArrowStyle	Gets or sets the arrow style of a 2D axis; this style is not applied to 3D chart areas.
AxisName	Gets the name of the axis, which can be either X, Y, X2 or Y2.
Crossing	Gets or sets the location at which an axis is crossed by its associated axis.
CustomLabels	Gets a CustomLabelsCollection object used to store CustomLabel objects.
Enabled	Gets or sets a value that indicates whether an axis is enabled.
InterlacedColor	Gets or sets the color of interlaced strip lines.
Interval	Gets or sets the interval of an axis.
IntervalAutoMode	Gets or sets a flag that determines if a fixed number of intervals is used on the axis, or if the number of intervals depends on the axis size.

IntervalOffset	Gets or sets the interval offset of an axis.
IntervalOffsetType	Gets or sets the interval offset type of an axis.
IntervalType	Gets or sets the interval type of an axis.
IsInterlaced	Gets or sets a flag that determines if interlaced strip lines are displayed for an axis.
IsLabelAutoFit	Gets or sets a flag that determines whether axis labels are automatically fitted.
IsLogarithmic	Gets or sets a flag which indicates whether the axis is logarithmic. Zeros or negative data values are not allowed on logarithmic charts.
IsMarginVisible	Gets or sets a flag that determines whether to add a margin to the axis.
IsMarksNextToAxis	Gets or sets a flag that determines whether tick marks and axis labels move with an axis if the crossing point of an axis changes.
IsReversed	Gets or sets a flag which indicates whether the axis is reversed.
IsStartedFromZero	Gets or sets a flag which indicates whether the minimum value of the axis will be automatically set to zero if all data point values are positive. If there are negative data point values, the minimum value of the data points will be used.
LabelAutoFitMaxFontSize	Gets or sets the maximum font size that can be used by the label auto-fitting algorithm.
LabelAutoFitMinFontSize	Gets or sets the minimum font size that can be used by the label auto-fitting algorithm.
LabelAutoFitStyle	Gets or sets the allowable label changes that can be made to enable the label to be fit along an axis.
LabelStyle	Gets or sets the LabelStyle properties of an axis.
LineColor	Gets or sets the line color of an axis.
LineDashStyle	Gets or sets the line style of an axis.
LineWidth	Gets or sets the line width of an axis, in pixels.
LogarithmBase	Gets or sets a value for the logarithm base for the logarithmic axis.
MajorGrid	Gets or sets a Grid object used to set the major grid line properties for an axis.
MajorTickMark	Gets or sets a TickMark object used to set the major tick mark

	properties of an axis.
Maximum	Gets or sets the maximum value of an axis.
MaximumAutoSize	Gets or sets the maximum size of the axis, measured as a percentage of the chart area. This value is used by the automatic layout algorithm.
Minimum	Gets or sets the minimum value of an axis.
MinorGrid	Gets or sets a Grid object used to specify the minor grid lines attributes of an axis.
MinorTickMark	Gets or sets a TickMark object used to set the minor tick mark properties of an axis.
Name	Gets or sets the name of the axis.
ScaleBreakStyle	Gets or sets the axis scale break style.
ScaleView	Gets or sets the view of an axis.
ScrollBar	Gets or sets an axis scrollbar.
StripLines	Gets a StripLinesCollection collection object.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextOrientation	Gets or sets the orientation of the text in the axis title.
Title	Gets or sets the title of the axis.
TitleAlignment	Gets or sets the alignment of an axis title.
TitleFont	Gets or sets the title font properties of an axis.
TitleForeColor	Gets or sets the text color of the axis title.
ToolTip	Gets or sets the tooltip used for the axis.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)

GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetPosition(Double)	Converts an axis value to a relative position (0-100%). If an axis is logarithmic, the value is converted to a linear scale.
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
PixelPositionToValue(Double)	Converts an absolute pixel position along an axis to an axis value. This method only works in paint events.
PositionToValue(Double)	Converts a relative coordinate along an axis to an axis value. This method only works in paint events.
RoundAxisValues()	Automatically rounds axis values.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)
ValueToPixelPosition(Double)	Converts an axis value to an absolute coordinate along an axis. Measured in pixels. This method only works in paint events.
ValueToPosition(Double)	Converts an axis value to its relative position (0-100%). If an axis has a logarithmic scale, the value is converted to a linear scale. This method only works in paint events.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Axis class.

Overloads

Axis()	Initializes a new instance of the Axis class.
Axis(ChartArea, AxisName)	Initializes a new instance of the Axis class with the specified `chartArea` and `axisTypeName` values.

Axis()

Initializes a new instance of the Axis class.

C#

```
public Axis ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis(ChartArea, AxisName)

Initializes a new instance of the Axis class with the specified `chartArea` and `axisTypeName` values.

C#

```
public Axis (System.Windows.Forms.DataVisualization.Charting.ChartArea
chartArea, System.Windows.Forms.DataVisualization.Charting.AxisName
axisTypeName);
```

Parameters

`chartArea` ChartArea

The chart area.

`axisTypeName` AxisName

The axis name.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.ArrowStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the arrow style of a 2D axis; this style is not applied to 3D chart areas.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.AxisArrowStyle
ArrowStyle { get; set; }
```

Property Value

AxisArrowStyle

An AxisArrowStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.AxisName Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the name of the axis, which can be either X, Y, X2 or Y2.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public virtual System.Windows.Forms.DataVisualization.Charting.AxisName
AxisName { get; }
```

Property Value

AxisName

An AxisName enumeration value.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.Crossing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the location at which an axis is crossed by its associated axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisCrossingValueConverter))]
public virtual double Crossing { get; set; }
```

Property Value

Double

A `double` value that represents where an axis is crossed by its associated axis. The default value is NaN.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Setting this property for a primary axis will determine where the other primary axis crosses it, and similarly setting it for a secondary axis will determine where the other secondary axis crosses it. For example, setting the Crossing property of the primary X-axis determines where the primary Y-axis will cross it.

Four modes can be used for the Crossing property of an axis:

- "Auto", which means that the crossing value will be set to the minimum or maximum value for the relevant axis. If the axis is primary, the value will be its minimum value, if the axis is secondary it will its maximum value. To set to "Auto" at run time, use a NaN value.

- "Minimum", which means the crossing value of the axis will be its minimum value. To set to "Minimum" at run-time use a MinValue value.
- "Maximum", which means the crossing value of the axis will be its maximum value. To set to "Maximum" at run-time use a MaxValue value.
- A specified `double` value that is between the minimum and maximum values for the relevant axis. If the crossing value is less then minimum value, or the crossing value is greater than maximum value, an exception will be thrown.

When a non-default crossing value is used, the labels and tick marks of the axis may or may not follow the crossing axis, depending on the IsMarksNextToAxis property setting. By default they will move with the crossing axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.CustomLabels Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a CustomLabelsCollection object used to store CustomLabel objects.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.CustomLabelsCollection
CustomLabels { get; }
```

Property Value

CustomLabelsCollection

A CustomLabelsCollection object.

Attributes BindableAttribute

Remarks

Use this property to add or remove custom labels for an axis. Using CustomLabel objects enables the developer to arbitrarily set the text for axis labels.

Only custom labels can be used in the second row of labels. Also, if a custom label is used in the first label row, then all labels in this first row must be custom labels.

Labels can also be set from the data points, or generated automatically from the axis scale.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that indicates whether an axis is enabled.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.AxisEnabled Enabled {
get; set; }
```

Property Value

AxisEnabled

An AxisEnabled enumeration value. The default value is Auto.

Attributes BindableAttribute

Remarks

If an axis is not enabled, the axis, along with its attributes (tick marks, strip lines, labels, and so forth), will not be displayed.

If an axis is enabled, the axis, along with all its attributes (tick marks, strip lines, labels, and so forth), will be displayed regardless of whether or not it is being used to plot a Series.

If a value of Auto is used, an axis may or may not be displayed, depending on whether it is being used to plot a Series. The axes that are used to plot data are determined by the XAxisType and YAxisType property settings.

> **Note**
>
> If an axis is not enabled but an adjacent axis is using grid lines, one of those grid lines will be displayed in place of the axis that is not enabled. The same situation

will occur if the **Enabled** property of an axis is set to **Auto**, but the axis is not being used to plot data; the grid line of an adjacent axis will be drawn instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.InterlacedColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of interlaced strip lines.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color InterlacedColor { get; set; }
```

Property Value

Color

A Color object that represents the color of interlaced strip lines.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Use this property to specify the color of interlaced strip lines that are displayed when you invoke the InterlacedColor property.

To specify the border color of the interlaced strip lines, you must display the strips using the StripLines collection property instead of the InterlacedColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.Interval Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisIntervalValueConverter))]
public double Interval { get; set; }
```

Property Value

Double

A `double` value that represents the interval of an axis. The default value is "Auto", which is represented by a value of zero (0).

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property determines how often the labels and major tick marks and grid lines associated with the axis are drawn if the `Interval` property value of a label, tick mark, or grid line is set to NaN, which signifies that it is not set.

The `Interval` property of a major tick mark, grid line or label of the axis has priority over the Interval property setting of the Axis object.

A value of zero represents an "Auto" value. This Interval property cannot be set to NaN. If you assign this property a value of NaN, a value of zero will be used instead.

Note The minor grid lines and tick marks of an axis never use this property for their interval.

The interval type of the axis is determined by the IntervalType property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IntervalAutoMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines if a fixed number of intervals is used on the axis, or if the number of intervals depends on the axis size.

C#

```
public System.Windows.Forms.DataVisualization.Charting.IntervalAutoMode
IntervalAutoMode { get; set; }
```

Property Value

IntervalAutoMode

An IntervalAutoMode enumeration value. The default value is FixedCount.

Remarks

When the Interval property is set to "Auto", the IntervalAutoMode property determines how many intervals should be calculated. If your chart is resized or redrawn often, you can set the IntervalAutoMode property to VariableCount, in which case the chart will dynamically calculate axis intervals. This is especially useful in situations where the chart intervals are displayed differently, depending on their data.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IntervalOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval offset of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisIntervalValueConverter))]
public double IntervalOffset { get; set; }
```

Property Value

Double

A `double` value that represents the interval offset of an axis. The default value is "Auto", which is represented by a value of zero (0).

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The IntervalOffset property determines the interval offset of an axis. This value is used as the interval offset of an axis' labels and the major tick marks and grid lines associated with the axis if the `IntervalOffset` property value of the labels, tick marks, or grid lines is not set, which is represented by a value of NaN.

The IntervalOffset property of the tick marks, labels and grid lines of an axis has priority over the IntervalOffset property setting of the Axis object.

A value of zero represents an "Auto" value. Also, Interval cannot be set to not set (that is, a NaN). If you assign NaN, a value of "Auto" - that is, a value of zero - will be used instead.

Note The minor grid lines and tick marks of an axis never use this property for their interval offset.

The interval offset type of the axis is determined by the IntervalOffsetType property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IntervalOffsetType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval offset type of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalOffsetType { get; set; }
```

Property Value

DateTimeIntervalType

The interval offset type of an axis. The default value is Auto.

Attributes BindableAttribute

Remarks

This property determines the interval offset type of an axis, and is used for the interval offset type of an axis' labels, and major tick marks or grid lines if the axis element's own `IntervalOffsetType` property value is set to NotSet.

Setting this property to NotSet will result in a value of Auto being used (only the axis elements such as major grid lines, etc. can be set to NotSet).

> **Note**
>
> The minor grid lines and tick marks of an axis never use this property for their interval offset type.

The interval of the axis is determined by the IntervalOffset property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IntervalType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval type of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType that represents the interval type of an axis. The default value is Auto.

Attributes BindableAttribute

Remarks

This property determines the interval type of an axis, and is used for the interval type of an axis' labels, major tick marks, or major grid lines if the axis element's own `IntervalType` property value is set to NotSet.

Setting this property to NotSet will result in a value of Auto being used. Only the axis elements, such as grid lines, can be set to NotSet.

Note The minor grid lines and tick marks of an axis never use this property for their interval type.

The interval of the axis is determined by the Interval property.

ⓘ **Important**

If the axis interval type is represented by a time such as hours, minutes, and so forth, then in order for zooming to function correctly, the **IntervalType** must be set to a time value as well. For example, if ChartArea1.AxisX.IntervalType is set to hours, then ChartArea1.CursorX.IntervalType could be set to minutes.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IsInterlaced Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines if interlaced strip lines are displayed for an axis.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsInterlaced { get; set; }
```

Property Value

Boolean

`true` if interlaced strip lines are displayed for an axis; otherwise, `False`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IsLabelAutoFit Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether axis labels are automatically fitted.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsLabelAutoFit { get; set; }
```

Property Value

Boolean

`true` if axis labels are automatically fitted; otherwise, `False`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IsLogarithmic Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag which indicates whether the axis is logarithmic. Zeros or negative data values are not allowed on logarithmic charts.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsLogarithmic { get; set; }
```

Property Value

Boolean

`true` if the axis is logarithmic; otherwise, `False`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IsMarginVisible Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether to add a margin to the axis.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsMarginVisible { get; set; }
```

Property Value

Boolean

`true` if a space is added between the first or last data point and the border of chart area.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IsMarksNextToAxis Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether tick marks and axis labels move with an axis if the crossing point of an axis changes.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual bool IsMarksNextToAxis { get; set; }
```

Property Value

Boolean

`true` if tick marks and axis labels move with an axis when the crossing point of an axis changes; otherwise, `False`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IsReversed Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag which indicates whether the axis is reversed.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsReversed { get; set; }
```

Property Value

Boolean

`true` if the axis is reversed; otherwise, `False`.

Attributes BindableAttribute

Remarks

If set to reversed, the values on the axis are in reversed sort order and the direction of values on the axis is flipped.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.IsStartedFromZero Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag which indicates whether the minimum value of the axis will be automatically set to zero if all data point values are positive. If there are negative data point values, the minimum value of the data points will be used.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsStartedFromZero { get; set; }
```

Property Value

Boolean

`true` if the minimum value of the axis is automatically set to zero when all data point values are positive; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.LabelAutoFitMaxFontSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the maximum font size that can be used by the label auto-fitting algorithm.

C#

```
[System.ComponentModel.Bindable(true)]
public int LabelAutoFitMaxFontSize { get; set; }
```

Property Value

Int32

An `integer` value.

Attributes BindableAttribute

Remarks

The font size cannot be less than 5.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.LabelAutoFitMinFontSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the minimum font size that can be used by the label auto-fitting algorithm.

C#

```
[System.ComponentModel.Bindable(true)]
public int LabelAutoFitMinFontSize { get; set; }
```

Property Value

Int32

An `integer` value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.LabelAutoFitStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the allowable label changes that can be made to enable the label to be fit along an axis.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.LabelAutoFitStyles
LabelAutoFitStyle { get; set; }
```

Property Value

LabelAutoFitStyles

A LabelAutoFitStyles enumeration value.

Attributes BindableAttribute

Remarks

The IsLabelAutoFit property must be set to `true`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.LabelStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the LabelStyle properties of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.LabelStyle LabelStyle
{ get; set; }
```

Property Value

LabelStyle

A LabelStyle object, which defines the axis label style.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Use this property to specify the style, formatting, and so forth, of axis labels.

If the IsLabelAutoFit property is `true`, the font size, font angle and the use of offset labels are determined automatically. If you set any one of the LabelStyle.Font.Size, Angle or IsStaggered properties, the IsLabelAutoFit property will be set to `false`.

Note that if a format that consists of a date without a time is used for labels, and the DateTime data point values that are used specify times, the times will affect the positions of the data points, even though they are not displayed due to the label formatting.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line color of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color object that represents the line color used to draw the axis. The default is Black.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Use the LineColor property to specify the line color of an axis.

Note If an axis is not enabled but an adjacent axis is using grid lines, one of those grid lines will be displayed in place of the axis that is not enabled. The same situation will occur if the Enabled property of an axis is set to Auto. In this case, the axis is not being used to plot data, and the grid line of an adjacent axis will be drawn instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line style of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
LineDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value.

Attributes BindableAttribute

Remarks

The LineDashStyle property determines the line style used for an axis. The line can be solid, dotted, and so forth.

Note If an axis is not enabled but an adjacent axis is using grid lines, one of those grid lines will be displayed in place of the axis that is not enabled. The same situation will occur if the Enabled property of an axis is set to Auto. In this case, the axis is not being used to plot data, and the grid line of an adjacent axis will be drawn instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line width of an axis, in pixels.

C#

```
[System.ComponentModel.Bindable(true)]
public int LineWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the width of an axis line. The default value is one (1) pixel.

Attributes BindableAttribute

Remarks

The LineWidth property gets or sets the width of an axis line, in pixels. If the width is set to zero (0), the line will not be displayed.

Note If an axis is not enabled but an adjacent axis is using grid lines, one of those grid lines will be displayed in place of the axis that is not enabled. The same situation will occur if the Enabled property of an axis is set to Auto. In this case, the axis is not being used to plot data, and the grid line of an adjacent axis will be drawn instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.LogarithmBase Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value for the logarithm base for the logarithmic axis.

C#

```
[System.ComponentModel.Bindable(true)]
public double LogarithmBase { get; set; }
```

Property Value

Double

A `double` value. The default value is base 10.

Attributes BindableAttribute

Remarks

This property only has an effect when the IsLogarithmic property is set to `true`.

Note that this property is only applicable to value axes, which are used to plot data. Setting this property for a categorical axis will result in an exception being thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.MajorGrid Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a Grid object used to set the major grid line properties for an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.Grid MajorGrid { get; set; }
```

Property Value

Grid

A Grid object used to get or set the major grid properties of an axis.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Major grid lines are always drawn in conjunction with a data point. That is, they will occur at the same point along an axis as a data point, for categorical axis tick marks. For value axes, major grid lines by default are drawn wherever labels occur.

You can use this property to change major grid line frequency, color, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.MajorTickMark Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a TickMark object used to set the major tick mark properties of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.TickMark
MajorTickMark { get; set; }
```

Property Value

TickMark

A TickMark object used to set the properties of a major tick mark of an axis.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Major tick marks are always drawn in conjunction with a data point. Categorical axis tick marks will occur at the same point along an axis as a data point. For value axes, major tick marks are drawn by default wherever labels occur.

You can use this property to change the major tick mark properties such as frequency, color, or size.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.Maximum Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the maximum value of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisMinMaxAutoValueConverter))]
public double Maximum { get; set; }
```

Property Value

Double

A `double` value that represents the maximum value of an axis. The default value is NaN.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

To automatically round the maximum and minimum axis values call the RoundAxisValues method. The Maximum value must be greater than the Minimum value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.MaximumAutoSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the maximum size of the axis, measured as a percentage of the chart area. This value is used by the automatic layout algorithm.

C#

```
public float MaximumAutoSize { get; set; }
```

Property Value

Single

A `float` value that represents the maximum size of the axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.Minimum Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the minimum value of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisMinMaxAutoValueConverter))]
public double Minimum { get; set; }
```

Property Value

Double

A `double` value that represents the minimum value of an axis. The default value is NaN.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Note that if you set this value explicitly, the X-values of data elements must be taken into account. If all data points have X-values of zero, the Chart control will assume the first data point occurs at zero. Also, if the Minimum value is explicitly set, the IsStartedFromZero property will be ignored. The Minimum value must be less than the Maximum value.

To automatically round the maximum and minimum axis values. call the RoundAxisValues method.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.MinorGrid Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a Grid object used to specify the minor grid lines attributes of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.Grid MinorGrid { get; set; }
```

Property Value

Grid

A Grid object used to get or set the attributes of the minor grid lines of an axis.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Minor grid lines are drawn in between the major grid lines.

Use this property to change the attributes of a minor grid line, such as frequency, color, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.MinorTickMark Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a TickMark object used to set the minor tick mark properties of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.TickMark
MinorTickMark { get; set; }
```

Property Value

TickMark

A TickMark object used for the minor tick mark properties of an axis.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Minor tick marks should be drawn between major tick marks, where labels usually occur.

Use this property to change the minor tick mark frequency, color, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the text name of the axis.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.ScaleBreakStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the axis scale break style.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public virtual
System.Windows.Forms.DataVisualization.Charting.AxisScaleBreakStyle
ScaleBreakStyle { get; set; }
```

Property Value

AxisScaleBreakStyle

An AxisScaleBreakStyle object, which represents the scale break style of the chart axis.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.ScaleView Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the view of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.AxisScaleView
ScaleView { get; set; }
```

Property Value

AxisScaleView

An AxisScaleView object, which represents the scale view of an axis.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.ScrollBar Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an axis scrollbar.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.AxisScrollBar ScrollBar { get; set; }
```

Property Value

AxisScrollBar

An AxisScrollBar object, which represents the scrollbar of an axis.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Use this property to manipulate the scrollbar of an axis.

For further details, refer to the AxisScrollBar class documentation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.StripLines Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a StripLinesCollection collection object.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.StripLinesCollection
StripLines { get; }
```

Property Value

StripLinesCollection

A StripLinesCollection collection object, which stores all StripLine objects for an axis.

Attributes BindableAttribute

Remarks

Strip lines are used to draw rectangular strips across a ChartArea object, and are always drawn across the entire <A href="">area.

These strip lines are not related to the grid lines.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.TextOrientation Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the orientation of the text in the axis title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.TextOrientation
TextOrientation { get; set; }
```

Property Value

TextOrientation

A TextOrientation enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.Title Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the title of the axis.

C#

```
[System.ComponentModel.Bindable(true)]
public string Title { get; set; }
```

Property Value

String

A `string` value that represents the title of the axis.

Attributes BindableAttribute

Remarks

Gets or sets the title of the axis.

You can set the font properties of the title using the TitleFont property.

You can set the color of the title using the TitleForeColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.TitleAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of an axis title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.StringAlignment TitleAlignment { get; set; }
```

Property Value

StringAlignment

A StringAlignment enumeration value.

Attributes BindableAttribute

Remarks

Possible alignment values are Center, Far, and Near. A value of Far equates to an alignment to the top or the right, depending on the orientation of the axis or strip line. In the same manner, a value of Near equates to an alignment to the bottom or the right.

If you want more control over the positioning of an axis title, you must perform custom drawing using the PostPaint or PrePaint event.

You can further align and position the titles of StripLine objects by using their TextLineAlignment and TitleAngle properties.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.TitleFont Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the title font properties of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.Font TitleFont { get; set; }
```

Property Value

Font

A Font object used for the font properties of a title.

Attributes BindableAttribute

Remarks

Gets or sets title font properties such as face or size.

To set font properties at run time, a new Font object must be created. Any attempt to change the properties of the current Font object at run time will result in a compilation error.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.TitleForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color of the axis title.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color TitleForeColor { get; set; }
```

Property Value

Color

A Color structure. The default color is Black.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Gets or sets the text color of an Axis object title.

You can use any valid ARGB (alpha, red, green, blue) color.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip used for the axis.

C#

```
[System.ComponentModel.Bindable(true)]
public string ToolTip { get; set; }
```

Property Value

String

A `string` value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.GetPosition(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Converts an axis value to a relative position (0-100%). If an axis is logarithmic, the value is converted to a linear scale.

```
C#
public double GetPosition (double axisValue);
```

Parameters

`axisValue` Double

The axis value.

Returns

Double

A `double` value that represents the relative position (0-100%).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.PixelPositionToValue(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Converts an absolute pixel position along an axis to an axis value. This method only works in paint events.

C#

```
public double PixelPositionToValue (double position);
```

Parameters

position Double

The pixel position.

Returns

Double

A `double` value that represents the axis value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.PositionToValue(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Converts a relative coordinate along an axis to an axis value. This method only works in paint events.

```
C#
public double PositionToValue (double position);
```

Parameters

`position` Double

The relative position (0-100%).

Returns

Double

A `double` value that represents the axis value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.RoundAxisValues Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Automatically rounds axis values.

C#

```
public void RoundAxisValues ();
```

Remarks

Call the RoundAxisValues method to round the values of an axis. Note that the interval used for the axis will not be changed when this method is called. This method is only supported for the X and X2 axes. The Y and Y2 axes values are rounded by default.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.ValueToPixelPosition(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Converts an axis value to an absolute coordinate along an axis. Measured in pixels. This method only works in paint events.

C#

```
public double ValueToPixelPosition (double axisValue);
```

Parameters

`axisValue` Double

The axis value.

Returns

Double

A `double` value that represents the pixel position.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Axis.ValueToPosition(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Converts an axis value to its relative position (0-100%). If an axis has a logarithmic scale, the value is converted to a linear scale. This method only works in paint events.

C#

```
public double ValueToPosition (double axisValue);
```

Parameters

`axisValue` Double

The axis value.

Returns

Double

A `double` value that represents the relative position (0-100%).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisArrowStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

An enumeration of axis arrow styles.

```
C#
public enum AxisArrowStyle
```

Inheritance Object → ValueType → Enum → AxisArrowStyle

Fields

Lines	3	A line-shaped arrow is used for the relevant axis.
None	0	No arrow is used for the relevant axis.
SharpTriangle	2	A sharp triangular arrow is used for the relevant axis.
Triangle	1	A triangular arrow is used for the relevant axis.

Remarks

The AxisArrowStyle enumeration represents the style for an axis arrow. If the ArrowStyle property is set to None, no arrow is displayed for an axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisEnabled Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the state of an axis.

C#

```
public enum AxisEnabled
```

Inheritance Object → ValueType → Enum → AxisEnabled

Fields

Auto	0	The axis is only enabled if it used to plot a Series.
False	2	The axis is never enabled.
True	1	The axis is always enabled.

Remarks

The AxisEnabled enumeration is used with the Enabled property to enable or disable an axis.

If the Enabled property is set to Auto, an axis may or may not be displayed, depending on whether it is being used to plot a Series. The axes that are used to plot data for a series are determined by the XAxisType and YAxisType property settings.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisName Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the name of an axis.

C#

```
public enum AxisName
```

Inheritance Object → ValueType → Enum → AxisName

Fields

X	0	Primary X-axis.
X2	2	Secondary X-axis.
Y	1	Primary Y-axis.
Y2	3	Secondary Y-axis.

Remarks

The AxisName enumeration represents the four different axes that can be used for a chart.

Bar and stacked bar charts have their axes rotated 90 degrees clockwise. For example, the primary X-axis for these two charts is the left-vertical axis.

For all charts except bar, stacked bar, pie and doughnut charts, the primary and secondary axes are as follows:

Axis Type	Description
Primary X-Axis	Bottom horizontal axis.

Axis Type	Description
Secondary X-Axis	Top horizontal axis.
Primary Y-Axis	Left vertical axis.
Secondary Y-Axis	Right vertical axis.

Bar and stacked bar charts have their axes rotated 90 degrees clockwise. For example, the primary X-axis for these two charts is the left-vertical axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a scale break on the Y-axis of the chart.

C#

```
public class AxisScaleBreakStyle
```

Inheritance Object → AxisScaleBreakStyle

Remarks

Scale breaks are intentional discontinuities on the Y-axis that are most often used to redistribute the data points in a series on a chart. This feature improves readability when there are large differences between the high and low values of the data in one series being plotted. Scale breaks are not drawn for differences between data in multiple series.

By default, scale breaks are disabled. You can enable scale breaks on the Y-axis by setting the Enabled property to `true`. To change the amount of vertical space of the scale break, you can specify a Spacing value. This is calculated as a percentage of the Y-axis scale.

You must have sufficient separation between ranges of data points in a single series to draw a scale break. By default, a scale break can be added only if there is a separation between the data ranges of at least 25% of the chart. If you enable a scale break but it does not appear, even though there is sufficient distance between the data ranges, you can set the CollapsibleSpaceThreshold property to a value less than 25.

Charts support up to five scale breaks per chart; however, the chart can become unreadable if you display more than one scale break. If you have more than two data ranges, consider using a different method for displaying this data. Use the MaxNumberOfBreaks property to reduce the number of scale breaks that can be displayed on the chart.

When a scale break is used, the Y-axis labels may become offset. To disable the label offset, set the IsStaggered property to `false`.

Scale breaks are not supported under any of the following conditions:

- Pie, doughnut, funnel, pyramid, radial or any stacked chart types are used.
- Custom intervals for labels, tick marks or grid lines are enabled.
- The minimum or maximum value for the axis is set.
- Custom labels are used.
- A logarithmic Y-axis is specified.
- Axis views on the Y-axis, which include scrolling and zooming, are used.
- 3-D charts are used.

Constructors

AxisScaleBreakStyle()	Initializes a new instance of the AxisScaleBreakStyle class.

Properties

BreakLineStyle	Gets or sets the style of the break line that will be used to draw the scale break.
CollapsibleSpaceThreshold	Gets or sets the threshold of space on the chart at which scale breaks are drawn.
Enabled	Gets or sets a flag that indicates whether scale breaks are enabled.
LineColor	Gets or sets the color of the scale break line.
LineDashStyle	Gets or sets the style of the scale break line.
LineWidth	Gets or sets the width of the scale break line.
MaxNumberOfBreaks	Gets or sets the maximum number of scale breaks to be displayed on the chart.
Spacing	Gets or sets the spacing gap between the lines of the scale break. The spacing gap is represented as a percentage of the Y-axis.

StartFromZero	Gets or sets a StartFromZero enumeration value that indicates whether to start the scale break from zero.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AxisScaleBreakStyle class.

C#

```
public AxisScaleBreakStyle ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.BreakLineStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the break line that will be used to draw the scale break.

C#

```
public System.Windows.Forms.DataVisualization.Charting.BreakLineStyle
BreakLineStyle { get; set; }
```

Property Value

BreakLineStyle

A BreakLineStyle enumeration value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.CollapsibleSpaceThreshold Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the threshold of space on the chart at which scale breaks are drawn.

C#

```
public int CollapsibleSpaceThreshold { get; set; }
```

Property Value

Int32

An `integer` value that specifies the threshold of space on the chart at which scale breaks are drawn.

Remarks

This property is measured as a percentage of the Y-axis scale. By default, its value is set to 25. This means that data points in a series that have a difference of at least 25% of the scale will be collapsed with a scale break.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether scale breaks are enabled.

C#

```
public bool Enabled { get; set; }
```

Property Value

Boolean

A `Boolean` value that specifies whether scale breaks are enabled.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the scale break line.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color value that represents the color of the scale break line.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the scale break line.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
LineDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle value that represents the style of the scale break line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of the scale break line.

C#

```
public int LineWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the width of the scale break line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.MaxNumberOfBreaks Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the maximum number of scale breaks to be displayed on the chart.

```
C#
public int MaxNumberOfBreaks { get; set; }
```

Property Value

Int32

An `integer` value that represents the maximum number of scale breaks to be displayed on the chart.

Remarks

The Chart is capable of displaying up to five scale breaks.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.Spacing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the spacing gap between the lines of the scale break. The spacing gap is represented as a percentage of the Y-axis.

C#

```
public double Spacing { get; set; }
```

Property Value

Double

A `double` value that represents the spacing gap between the lines of the scale break.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleBreakStyle.StartFromZero Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a StartFromZero enumeration value that indicates whether to start the scale break from zero.

C#

```
public System.Windows.Forms.DataVisualization.Charting.StartFromZero
StartFromZero { get; set; }
```

Property Value

StartFromZero

A StartFromZero enumeration value that indicates whether to start the scale break from zero.

Remarks

If the StartFromZero property is set to Auto, the chart will automatically calculate whether to start the Y-axis scale minimum from the lowest value in the series or from zero.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a scaled view of the chart.

C#

```
public class AxisScaleView
```

Inheritance Object → AxisScaleView

Constructors

AxisScaleView()	Initializes a new instance of the AxisScaleView class.

Properties

IsZoomed	Gets a flag that indicates whether the axis is zoomed in.
MinSize	Gets or sets the minimum size of the AxisScaleView object.
MinSizeType	Gets or sets the unit of measurement of the MinSize property.
Position	Gets or sets the position of the scale view.
Size	Gets or sets the size of the scale view.
SizeType	Gets or sets the unit of measurement for the Size property.
SmallScrollMinSize	Gets or sets the minimum small scrolling size. Used only if the small scrolling size is not set.
SmallScrollMinSizeType	Gets or sets the unit of measurement for the SmallScrollMinSize property.
SmallScrollSize	Gets or sets the small scrolling size.
SmallScrollSizeType	Gets or sets the unit of measurement for the SmallScrollMinSize

	property.
ViewMaximum	Gets the maximum axis value of a data view.
ViewMinimum	Gets the minimum axis value of a data view.
Zoomable	Gets or sets a flag that indicates whether the zooming user interface is enabled.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Scroll(DateTime)	Scrolls to a position along an axis specified by the `newPosition` parameter, which is expressed as a DateTime value.
Scroll(Double)	Scrolls to a position along an axis specified by the `newPosition` parameter.
Scroll(ScrollType)	Scrolls along an axis according to the direction and size indicated by the `scrollType` parameter.
ToString()	Returns a string that represents the current object. (Inherited from Object)
Zoom(Double, Double)	Sets a new axis data view and/or position based on the specified start and end values.
Zoom(Double, Double, DateTimeIntervalType)	Sets a new axis data view and/or position based on the specified start position, view size and unit of measurement.
Zoom(Double, Double, DateTimeIntervalType, Boolean)	Sets a new axis data view and/or position based on the specified start position, view size, unit of measurement and save state.
ZoomReset()	Resets one zoom operation by restoring the axis data view.
ZoomReset(Int32)	Resets the specified number of zoom operations by restoring the axis data view.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AxisScaleView class.

C#

```
public AxisScaleView ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.IsZoomed Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a flag that indicates whether the axis is zoomed in.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public bool IsZoomed { get; }
```

Property Value

Boolean

`true` if the axis is zoomed in; otherwise, `false`.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.MinSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the minimum size of the AxisScaleView object.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public double MinSize { get; set; }
```

Property Value

Double

A `double` that represents the minimum size of the scale view.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.MinSizeType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unit of measurement of the MinSize property.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
MinSizeType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that represents the unit of measurement of the MinSize property.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.Position Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the position of the scale view.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleDateNanValueConverter))]
public double Position { get; set; }
```

Property Value

Double

A `double` that represents the position of the scale view.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.Size Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the scale view.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleNanValueConverter))]
public double Size { get; set; }
```

Property Value

Double

A `double` that represents the size of the scale view.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.SizeType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unit of measurement for the Size property.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
SizeType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that represents the unit of measurement for the Size property.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.SmallScrollMinSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the minimum small scrolling size. Used only if the small scrolling size is not set.

C#

```
[System.ComponentModel.Bindable(true)]
public double SmallScrollMinSize { get; set; }
```

Property Value

Double

A `double` that represents the small scrolling size.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.SmallScrollMinSizeType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unit of measurement for the SmallScrollMinSize property.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
SmallScrollMinSizeType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that represents the unit of measurement for the SmallScrollMinSize property.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.SmallScrollSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the small scrolling size.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisMinMaxAutoValueConverter))]
public double SmallScrollSize { get; set; }
```

Property Value

Double

A `double` value that represents the small scrolling size.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.SmallScrollSizeType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unit of measurement for the SmallScrollMinSize property.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
SmallScrollSizeType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that represents the unit of measurement for the SmallScrollMinSize property.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.ViewMaximum Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the maximum axis value of a data view.

C#

```
[System.ComponentModel.Browsable(false)]
public double ViewMaximum { get; }
```

Property Value

Double

A `double` that represents the maximum axis value for the data view.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.ViewMinimum Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the minimum axis value of a data view.

C#

```
[System.ComponentModel.Browsable(false)]
public double ViewMinimum { get; }
```

Property Value

Double

A `double` that represents the minimum axis value for the data view.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.Zoomable Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the zooming user interface is enabled.

C#

```
[System.ComponentModel.Bindable(true)]
public bool Zoomable { get; set; }
```

Property Value

Boolean

`true` if the zooming user interface is enabled; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.Scroll Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Scrolls to a position along an axis using the specified parameters.

Overloads

Scroll(DateTime)	Scrolls to a position along an axis specified by the `newPosition` parameter, which is expressed as a DateTime value.
Scroll(Double)	Scrolls to a position along an axis specified by the `newPosition` parameter.
Scroll(ScrollType)	Scrolls along an axis according to the direction and size indicated by the `scrollType` parameter.

Scroll(DateTime)

Scrolls to a position along an axis specified by the `newPosition` parameter, which is expressed as a DateTime value.

C#

```
public void Scroll (DateTime newPosition);
```

Parameters

`newPosition` DateTime

The new position to scroll.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Scroll(Double)

Scrolls to a position along an axis specified by the `newPosition` parameter.

C#

```
public void Scroll (double newPosition);
```

Parameters

`newPosition` Double

The new position to scroll.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Scroll(ScrollType)

Scrolls along an axis according to the direction and size indicated by the `scrollType` parameter.

C#

```
public void Scroll
(System.Windows.Forms.DataVisualization.Charting.ScrollType scrollType);
```

Parameters

`scrollType` ScrollType

The direction and size to scroll.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.Zoom Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets a new axis data view and/or position.

Overloads

Zoom(Double, Double)	Sets a new axis data view and/or position based on the specified start and end values.
Zoom(Double, Double, DateTimeIntervalType)	Sets a new axis data view and/or position based on the specified start position, view size and unit of measurement.
Zoom(Double, Double, DateTimeIntervalType, Boolean)	Sets a new axis data view and/or position based on the specified start position, view size, unit of measurement and save state.

Zoom(Double, Double)

Sets a new axis data view and/or position based on the specified start and end values.

C#

```
public void Zoom (double viewStart, double viewEnd);
```

Parameters

`viewStart` Double

The new start position for the axis scale view.

`viewEnd` Double

The new end position for the axis scale view.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Zoom(Double, Double, DateTimeIntervalType)

Sets a new axis data view and/or position based on the specified start position, view size and unit of measurement.

C#

```
public void Zoom (double viewPosition, double viewSize,
System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
viewSizeType);
```

Parameters

`viewPosition` Double

The new start position for the axis scale view.

`viewSize` Double

The new size for the axis scale view.

`viewSizeType` DateTimeIntervalType

The new unit of measurement of the size.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Zoom(Double, Double, DateTimeIntervalType, Boolean)

Sets a new axis data view and/or position based on the specified start position, view size, unit of measurement and save state.

C#

```
public void Zoom (double viewPosition, double viewSize,
System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
viewSizeType, bool saveState);
```

Parameters

`viewPosition` Double

The new start position for the axis scale view.

`viewSize` Double

The new size for the axis scale view.

`viewSizeType` DateTimeIntervalType

The new unit of measurement of the size.

`saveState` Boolean

Indicates whether the current size and/or position must be saved.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScaleView.ZoomReset Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Resets the zoom operations by restoring the axis data view.

Overloads

ZoomReset()	Resets one zoom operation by restoring the axis data view.
ZoomReset(Int32)	Resets the specified number of zoom operations by restoring the axis data view.

ZoomReset()

Resets one zoom operation by restoring the axis data view.

C#

```
public void ZoomReset ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZoomReset(Int32)

Resets the specified number of zoom operations by restoring the axis data view.

C#

```
public void ZoomReset (int numberOfViews);
```

Parameters

`numberOfViews` Int32

Number of zoom operations to reset.

Remarks

Set the `numberOfViews` parameter to zero (0) to reset all zoom operations.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the scrollbar of a chart axis.

C#

```
public class AxisScrollBar : IDisposable
```

Inheritance Object → AxisScrollBar

Implements IDisposable

Remarks

The AxisScrollBar class is exposed as the ScrollBar property of the Axis class.

Scrollbars allow the end user to scroll through data when a subset of data is being displayed. A subset of data is referred to as a view, and is represented by the AxisScaleView class.

Scrollbars are only shown when zooming occurs. In other words, even if a scrollbar is enabled, it will only be visible when a view is being displayed.

> ⓘ **Important**
>
> To fix the position of scrollbars, the plotting area position of a chart area must not be set to **Auto**. To explicitly set the plotting area position, use the **InnerPlotPosition** property.

To display scrollbars when a chart is first displayed, use the AxisScaleView class properties and methods so that a view is initially displayed to end users.

The background color, button color and width of scrollbars can be set using this AxisScrollBar class.

Note that there are two main types of scrolling: large scrolling, which occurs when the end user clicks in the background of a scrollbar - that is, that part of a scrollbar not covered by the thumb; and small scrolling, which occurs when the end user clicks a left/right or up/down button. To control the large scrolling size, use the Size property of the ScaleView object. To set the small scrolling size, use the SmallScrollSize property of the AxisScaleView class.

Constructors

AxisScrollBar()	Initializes a new instance of the AxisScrollBar class.

Properties

Axis	Gets the Axis object that uses the scrollbar.
BackColor	Gets or sets the background color of a scrollbar.
ButtonColor	Gets or sets the color of the scrollbar buttons.
ButtonStyle	Gets or sets the style of the scrollbar button.
ChartArea	Gets the chart area that contains this scrollbar.
Enabled	Gets or sets a flag that determines whether a scrollbar is enabled.
IsPositionedInside	Gets or sets a flag that indicates whether scroll bar is positioned inside or outside the chart area.
IsVisible	Gets a flag that indicates whether the scrollbar is visible.
LineColor	Gets or sets the line color of a scrollbar.
Size	Gets or sets the width of a scrollbar, in pixels.

Methods

Dispose()	Releases all resources used by the current instance of the AxisScrollBar.
Dispose(Boolean)	Releases the unmanaged resources used by the AxisScrollBar

	class and optionally releases the managed resources.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AxisScrollBar class.

C#

```
public AxisScrollBar ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.Axis Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Axis object that uses the scrollbar.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.Axis Axis { get; }
```

Property Value

Axis

An Axis object.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of a scrollbar.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value that represents the background color of the scrollbar. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property can be set to any valid ARGB (alpha, red, green, blue) color.

If the value of this property is Empty, the background color of the scrollbar will be determined by the BackColor property value of the chart area the scrollbar is being used for.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.ButtonColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the scrollbar buttons.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ButtonColor { get; set; }
```

Property Value

Color

A Color value that represents the color of the scrollbar buttons.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Use this property to get or set the color of the left/right, up/down, reset and thumb buttons of a scrollbar.

If the value of this property is Empty, the color used for the buttons will be determined by the LineColor property value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.ButtonStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the scrollbar button.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ScrollBarButtonStyles
ButtonStyle { get; set; }
```

Property Value

ScrollBarButtonStyles

A ScrollBarButtonStyles enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.ChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the chart area that contains this scrollbar.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.ChartArea ChartArea {
get; }
```

Property Value

ChartArea

A ChartArea object.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether a scrollbar is enabled.

C#

```
[System.ComponentModel.Bindable(true)]
public bool Enabled { get; set; }
```

Property Value

Boolean

`true` if the scrollbar is enabled for an axis, `false` if the scrollbar is not enabled. The default value is `true`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.IsPositionedInside Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether scroll bar is positioned inside or outside the chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsPositionedInside { get; set; }
```

Property Value

Boolean

`true` if the scroll bar is positioned inside or outside the chart area; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.IsVisible Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a flag that indicates whether the scrollbar is visible.

C#

```
[System.ComponentModel.Browsable(false)]
public bool IsVisible { get; }
```

Property Value

Boolean

`true` if the scrollbar is visible; otherwise, `false`.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line color of a scrollbar.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color value that represents the line color of the scrollbar. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property represents the color used for the arrows in up/down and left/right buttons, the symbol inside the reset button, as well as the border of the entire scrollbar.

If the value of this property is Empty, the line color used will be determined by the LineColor property value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.Size Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of a scrollbar, in pixels.

C#

```
[System.ComponentModel.Bindable(true)]
public double Size { get; set; }
```

Property Value

Double

The width of a scrollbar in pixels. The default value is 14 pixels. The value can range from 5 to 20 pixels.

Attributes BindableAttribute

Remarks

Gets or sets the width of a scrollbar, which can be from 5 to 20 pixels.

Set this property to zero to disable a scrollbar.

> **Important**
>
> To fix the position of scrollbars, the plotting area position of a chart area must not be set to **Auto**. To explicitly set the plotting area position, use the **InnerPlotPosition** property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisScrollBar.Dispose Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the resources used by the AxisScrollBar class.

Overloads

Dispose()	Releases all resources used by the current instance of the AxisScrollBar.
Dispose(Boolean)	Releases the unmanaged resources used by the AxisScrollBar class and optionally releases the managed resources.

Dispose()

Releases all resources used by the current instance of the AxisScrollBar.

C#

```
public void Dispose ();
```

Implements

Dispose()

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Dispose(Boolean)

Releases the unmanaged resources used by the AxisScrollBar class and optionally releases the managed resources.

```
C#

protected virtual void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AxisType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the axis type for the X and Y-axes of a Series.

C#

```
public enum AxisType
```

Inheritance Object → ValueType → Enum → AxisType

Fields

Primary	0	Designates an axis as the primary axis.
Secondary	1	Designates an axis as the secondary axis.

Remarks

The AxisType enumeration represents the axis type used for the X and Y-axes of a Series.

A Series is plotted using two axes, with the exception of pie and doughnut charts. This enumeration is used in conjunction with the XAxisType and YAxisType properties to set the axes used for plotting the associated data points of the series.

For all charts except for bar, stacked bar, pie and doughnut types, the primary and secondary axes are as follows:

Axis Type	Description
Primary X-Axis	Bottom horizontal axis.
Secondary X-Axis	Top horizontal axis.
Primary Y-Axis	Left vertical axis.

Axis Type	Description
Secondary Y-Axis	Right vertical axis.

Bar and stacked bar charts have their axes rotated 90 degrees clockwise. For example, the primary X-axis for these two charts is the left-vertical axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Border3DAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents an annotation with a 3D border.

C#

```
public class Border3DAnnotation :
System.Windows.Forms.DataVisualization.Charting.RectangleAnnotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → TextAnnotation → RectangleAnnotation → Border3DAnnotation

Remarks

A Border3DAnnotation can also display inner text.

Constructors

Border3DAnnotation()	Initializes a new instance of the Border3DAnnotation class.

Properties

Alignment	Gets or sets the content alignment of an annotation. (Inherited from Annotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed

	to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point. (Inherited from Annotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets or sets the annotation type name.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached.

	(Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	Gets or sets the background color of the annotation. (Inherited from RectangleAnnotation)
BackGradientStyle	Gets or sets the gradient style of the annotation. (Inherited from RectangleAnnotation)
BackHatchStyle	Gets or sets the hatch style of an annotation background. (Inherited from RectangleAnnotation)
BackSecondaryColor	Gets or sets the secondary color for an annotation background. (Inherited from RectangleAnnotation)
BorderSkin	Gets or sets the skin attributes of the 3D border.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
Font	Gets or sets the font for the annotation text. (Inherited from TextAnnotation)
ForeColor	Gets or sets the text color of an annotation. (Inherited from Annotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsMultiline	Gets or sets a flag that indicates whether the annotation text is multiline. (Inherited from TextAnnotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	Gets or sets the line color of an annotation. (Inherited from RectangleAnnotation)

LineDashStyle	Gets or sets the line style of an annotation. (Inherited from RectangleAnnotation)
LineWidth	Gets or sets the line width, in pixels, of an annotation. (Inherited from RectangleAnnotation)
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the text of the annotation. (Inherited from TextAnnotation)
TextStyle	Gets or sets the text style of an annotation. (Inherited from Annotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)
BeginTextEditing()	Begins editing of the annotation text by an end-user. (Inherited from TextAnnotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends user placement of an annotation. (Inherited from TextAnnotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
StopTextEditing()	Stops end-user editing of the annotation text. (Inherited from TextAnnotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Border3DAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Border3DAnnotation class.

C#

```
public Border3DAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Border3DAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the annotation type name.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

This abstract property is used to get the name of each annotation type, such as Line, Rectangle, Ellipse, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Border3DAnnotation.BorderSkin Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the skin attributes of the 3D border.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.LegendConverter))]
public System.Windows.Forms.DataVisualization.Charting.BorderSkin BorderSkin
{ get; set; }
```

Property Value

BorderSkin

A BorderSkin object, which determines the attributes of the 3D border.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the drawing properties of the 3D border skin.

C#

```
public class BorderSkin :
System.Windows.Forms.DataVisualization.Charting.ChartElement
```

Inheritance Object → ChartElement → BorderSkin

Constructors

BorderSkin()	Initializes a new instance of the BorderSkin class.

Properties

BackColor	Gets or sets the background color of a skin frame.
BackGradientStyle	Gets or sets the background gradient style of a skin frame.
BackHatchStyle	Gets or sets the background hatch style of a skin frame.
BackImage	Gets or sets the background image of a skin frame.
BackImageAlignment	Gets or sets the background image alignment of a skin frame.
BackImageTransparentColor	Gets or sets a color that is replaced with a transparent color when the background image of a border skin frame is drawn.
BackImageWrapMode	Gets or sets the drawing mode for the background image of a border skin frame.
BackSecondaryColor	Gets or sets the secondary background color of a border skin frame.
BorderColor	Gets or sets the border color of a skin frame.

BorderDashStyle	Gets or sets the style of the border line of a border skin frame.
BorderWidth	Gets or sets the width of the border line of a border skin frame.
PageColor	Gets or sets the page color of a border skin.
SkinStyle	Gets or sets the style of a border skin.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources. (Inherited from ChartElement)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the BorderSkin class.

C#

```
public BorderSkin ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of a skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background gradient style of a skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background hatch style of a skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BackImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image of a skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
public string BackImage { get; set; }
```

Property Value

String

A `string` value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BackImageAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image alignment of a skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.ChartImageAlignmentStyle
BackImageAlignment { get; set; }
```

Property Value

ChartImageAlignmentStyle

A ChartImageAlignmentStyle enumeration value.

Attributes BindableAttribute

Remarks

The BackImageAlignment property is used by Unscaled drawing mode.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BackImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color that is replaced with a transparent color when the background image of a border skin frame is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackImageTransparentColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BackImageWrapMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the drawing mode for the background image of a border skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartImageWrapMode
BackImageWrapMode { get; set; }
```

Property Value

ChartImageWrapMode

A ChartImageWrapMode enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary background color of a border skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This color is combined with the primary background color, which is set with the BackColor property, when the BackHatchStyle and BackGradientStyle properties are used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of a skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the border line of a border skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.BorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of the border line of a border skin frame.

C#

```
[System.ComponentModel.Bindable(true)]
public int BorderWidth { get; set; }
```

Property Value

Int32

An `integer` value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.PageColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the page color of a border skin.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color PageColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkin.SkinStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of a border skin.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.BorderSkinStyle
SkinStyle { get; set; }
```

Property Value

BorderSkinStyle

A BorderSkinStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BorderSkinStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a style for the border skin of a chart.

```
C#
public enum BorderSkinStyle
```

Inheritance Object → ValueType → Enum → BorderSkinStyle

Fields

Emboss	1	Use an embossed border skin.
FrameThin1	4	Use a thin border skin with rounded corners.
FrameThin2	5	Use a thin border skin with rounded top corners.
FrameThin3	6	Use a thin border skin with square corners.
FrameThin4	7	Use a thin border skin with square outside corners and rounded inside corners.
FrameThin5	8	Use a thin border skin with rounded corners and screws.
FrameThin6	9	Use a thin border skin with square inside corners and rounded outside corners.
FrameTitle1	10	Use a border skin with rounded corners.
FrameTitle2	11	Use a border skin with rounded top corners.
FrameTitle3	12	Use a border skin with square corners.
FrameTitle4	13	Use a border skin with rounded inside corners and square outside corners.
FrameTitle5	14	Use a border skin with rounded corners and screws.
FrameTitle6	15	Use a border skin with rounded outside corners and square inside corners.

FrameTitle7	16	Use a border skin with rounded corners. No border on the right side.
FrameTitle8	17	Use a border skin with rounded corners on top and bottom sides only.
None	0	No border skin is used.
Raised	2	Use a raised border skin.
Sunken	3	Use a sunken border skin.

Remarks

The BorderSkin class represents the style of the border skin used for a chart image. If you set the SkinStyle property to None, no skin will be used. This is the default value for the SkinStyle property.

> **Note**
>
> The difference between the sets of border skin styles that use frames and follow either the naming convention "FrameThin*" or "FrameTitle*" (for example, **FrameThin1** or **FrameTitle1**) is the shape and thickness of their frames.

In addition, if you use one of the "FrameThin" style border skins, the title of the chart will not be displayed. However, if you use one of the "FrameTitle" style border skins, the title of the chart will appear in the top-horizontal part of the frame.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

BreakLineStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies line styles for axis scale breaks.

C#

```
public enum BreakLineStyle
```

Inheritance Object → ValueType → Enum → BreakLineStyle

Fields

None	0	Scale break is not displayed.
Ragged	3	Scale break is displayed as a ragged line.
Straight	1	Scale break is displayed as a straight line.
Wave	2	Scale break is displayed as a wavy line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a callout annotation.

C#

```
public class CalloutAnnotation :
System.Windows.Forms.DataVisualization.Charting.TextAnnotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → TextAnnotation → CalloutAnnotation

Remarks

A callout annotation is the only annotation that draws a connection between the annotation position and the anchor point. This class can also display text and automatically calculate its required size.

Different callout styles are supported.

Constructors

CalloutAnnotation()	Initializes a new instance of the CalloutAnnotation class.

Properties

Alignment	Gets or sets the content alignment of an annotation. (Inherited from Annotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed

	to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the annotation position alignment to the anchor point.
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset that the X-position of the annotation has from the anchor point.
AnchorOffsetY	Gets or sets the offset that the Y-position of the annotation has from the anchor point.
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets or sets the annotation type name. For internal use, should not be used.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)

AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	Gets or sets the background color of the annotation.
BackGradientStyle	Gets or sets the annotation background gradient style.
BackHatchStyle	Gets or sets the background hatch style of the annotation.
BackSecondaryColor	Gets or sets the secondary color for the annotation background.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
CalloutAnchorCap	Gets or sets the style of the callout line anchor cap.
CalloutStyle	Gets or sets the annotation callout style.
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
Font	Gets or sets the font for the annotation text. (Inherited from TextAnnotation)
ForeColor	Gets or sets the text color of an annotation. (Inherited from Annotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsMultiline	Gets or sets a flag that indicates whether the annotation text is multiline. (Inherited from TextAnnotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	Gets or sets the annotation line color.
LineDashStyle	Gets or sets the annotation line style.

LineWidth	Gets or sets the line width of the annotation.
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the text of the annotation. (Inherited from TextAnnotation)
TextStyle	Gets or sets the text style of an annotation. (Inherited from Annotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)

BeginTextEditing()	Begins editing of the annotation text by an end-user. (Inherited from TextAnnotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends user placement of an annotation. (Inherited from TextAnnotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
StopTextEditing()	Stops end-user editing of the annotation text. (Inherited from TextAnnotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the CalloutAnnotation class.

C#

```
public CalloutAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.AnchorAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the annotation position alignment to the anchor point.

C#

```
public override System.Drawing.ContentAlignment AnchorAlignment { get; set;
}
```

Property Value

ContentAlignment

A ContentAlignment value that represents the alignment of the annotation to the anchor point.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.AnchorOffsetX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the offset that the X-position of the annotation has from the anchor point.

C#

```
public override double AnchorOffsetX { get; set; }
```

Property Value

Double

A `double` value that represents the offset of the annotation's X-position from the anchor point.

Remarks

The CalloutAnnotation must be anchored using either the AnchorDataPoint or the AnchorX property, and its X property must be set to Double.NaN.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.AnchorOffsetY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the offset that the Y-position of the annotation has from the anchor point.

C#

```
public override double AnchorOffsetY { get; set; }
```

Property Value

Double

A `double` value that represents the offset of the annotation's Y-position from the anchor point.

Remarks

The CalloutAnnotation must be anchored using either the AnchorDataPoint or the AnchorY property, and its Y property must be set to Double.NaN.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the annotation type name. For internal use, should not be used.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

For internal use, should not be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the annotation.

C#

```
[System.ComponentModel.Browsable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color enumeration value used for the background of an annotation.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the annotation background gradient style.

C#

```
[System.ComponentModel.Browsable(true)]
public override
System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration value used to draw the annotation background.

Attributes BrowsableAttribute

Remarks

Two colors are used to draw the gradient, and are set with the BackColor and BackSecondaryColor properties.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background hatch style of the annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle value used to draw the annotation background.

Attributes BrowsableAttribute

Remarks

Two colors are used to draw the hatching, and are set with the BackColor and BackSecondaryColor properties.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color for the annotation background.

C#

```
[System.ComponentModel.Browsable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color enumeration value used in conjunction with a primary color to draw the annotation background with hatching or gradient fill.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This color is used with the BackColor property when the BackHatchStyle or BackGradientStyle properties are used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.CalloutAnchorCap Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the callout line anchor cap.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle
CalloutAnchorCap { get; set; }
```

Property Value

LineAnchorCapStyle

A LineAnchorCapStyle enumeration value that represents the style to use for a callout line anchor cap.

Attributes BindableAttribute

Remarks

The CalloutAnchorCap property sets the style of the anchor cap for the line that connects the annotation with the anchor point. This property is only applicable when a SimpleLine or Borderline callout style is used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.CalloutStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the annotation callout style.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual System.Windows.Forms.DataVisualization.Charting.CalloutStyle
CalloutStyle { get; set; }
```

Property Value

CalloutStyle

A CalloutStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the annotation line color.

C#

```
[System.ComponentModel.Browsable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color value used to draw the annotation line.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the annotation line style.

C#

```
[System.ComponentModel.Browsable(true)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartDashStyle LineDashStyle
{ get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value used to draw the annotation line.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutAnnotation.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line width of the annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public override int LineWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the annotation line width, in pixels.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CalloutStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the available callout styles. Applicable only to callout annotations.

C#

```
public enum CalloutStyle
```

Inheritance Object → ValueType → Enum → CalloutStyle

Fields

Borderline	1	A border is drawn around the callout text and a line points to the anchor point.
Cloud	2	Callout text is inside the cloud and smaller clouds point to the anchor point.
Ellipse	5	An ellipse is drawn around the callout text, which is connected to the anchor point.
Perspective	6	A perspective rectangle is drawn around the callout text, which is connected to the anchor point.
Rectangle	3	A rectangle is drawn around the callout text, which is connected to the anchor point.
RoundedRectangle	4	A rounded rectangle is drawn around the callout text, which is connected to the anchor point.
SimpleLine	0	Callout text is underlined and a line points to the anchor point.

Remarks

The CalloutStyle enumeration is used in conjunction with the CalloutStyle property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Serves as the root class of the Chart control.

C#

```
[System.Drawing.ToolboxBitmap(typeof(System.Windows.Forms.DataVisualization.
Charting.Chart), "ChartControl.ico")]
public class Chart : System.Windows.Forms.Control, IDisposable,
System.ComponentModel.ISupportInitialize
```

Inheritance Object → MarshalByRefObject → Component → Control → Chart

Attributes ToolboxBitmapAttribute

Implements ISupportInitialize , IDisposable

Remarks

This class exposes all of the properties, methods and events of the Chart Windows control.

Two important properties of the Chart class are the Series and ChartAreas properties, both of which are collection properties. The Series collection property stores Series objects, which are used to store data that is to be displayed, along with attributes of that data. The ChartAreas collection property stores ChartArea objects, which are primarily used to draw one or more charts using one set of axes.

Constructors

Chart()	Initializes a new instance of the Chart class.

Properties

AccessibilityObject	Gets the AccessibleObject assigned to the control. (Inherited from Control)
AccessibleDefaultActionDescription	Gets or sets the default action description of the control for use by accessibility client applications. (Inherited from Control)
AccessibleDescription	Gets or sets the description of the control used by accessibility client applications. (Inherited from Control)
AccessibleName	Gets or sets the name of the control used by accessibility client applications. (Inherited from Control)
AccessibleRole	Gets or sets the accessible role of the control. (Inherited from Control)
AllowDrop	Gets or sets a value indicating whether the control can accept data that the user drags onto it. (Inherited from Control)
Anchor	Gets or sets the edges of the container to which a control is bound and determines how a control is resized with its parent. (Inherited from Control)
Annotations	Gets or sets a collection that stores the chart annotations.
AntiAliasing	Gets or sets a value that determines whether anti-aliasing is used when text and graphics are drawn.
AutoScrollOffset	Gets or sets where this control is scrolled to in ScrollControlIntoView(Control). (Inherited from Control)
AutoSize	This property is not relevant for this class. (Inherited from Control)
BackColor	Gets or sets the background color of the Chart object.
BackGradientStyle	Gets or sets the orientation for the background gradient of a Chart control. Also determines whether a gradient is used.
BackgroundImage	The BackgroundImage property is not used. Use the BackImage property instead.
BackgroundImageLayout	Gets or sets the background image layout as defined in the ImageLayout enumeration. (Inherited from Control)

BackHatchStyle	Gets or sets the hatching style of the Chart control.
BackImage	Gets or sets the background image of the Chart control.
BackImageAlignment	Gets or sets the background image alignment used for the Unscaled drawing mode.
BackImageTransparentColor	Gets or sets the color of the Chart control that will be displayed as transparent.
BackImageWrapMode	Gets or sets the drawing mode for the background image of the Chart control.
BackSecondaryColor	Gets or sets the secondary color of the chart background.
BindingContext	Gets or sets the BindingContext for the control. (Inherited from Control)
BorderColor	Gets or sets the color of the chart border.
BorderDashStyle	Gets or sets the style of the border.
BorderlineColor	Gets or sets the color of the border line.
BorderlineDashStyle	Gets or sets the style of the border line.
BorderlineWidth	Gets or sets the width of the border line.
BorderSkin	Gets or sets a BorderSkin object, which provides border skin functionality for the Chart control.
BorderWidth	Gets or set s the width of the chart border.
Bottom	Gets the distance, in pixels, between the bottom edge of the control and the top edge of its container's client area. (Inherited from Control)
Bounds	Gets or sets the size and location of the control including its nonclient elements, in pixels, relative to the parent control. (Inherited from Control)
BuildNumber	Gets the build number of the Chart control.
CanEnableIme	Gets a value indicating whether the ImeMode property can be set to an active value, to enable IME support. (Inherited from Control)
CanFocus	Gets a value indicating whether the control can receive focus. (Inherited from Control)
CanRaiseEvents	Determines if events can be raised on the control. (Inherited from Control)

CanSelect	Gets a value indicating whether the control can be selected. (Inherited from Control)
Capture	Gets or sets a value indicating whether the control has captured the mouse. (Inherited from Control)
CausesValidation	Gets or sets a value indicating whether the control causes validation to be performed on any controls that require validation when it receives focus. (Inherited from Control)
ChartAreas	Gets a read-only ChartAreaCollection object that is used to store ChartArea objects.
ClientRectangle	Gets the rectangle that represents the client area of the control. (Inherited from Control)
ClientSize	Gets or sets the height and width of the client area of the control. (Inherited from Control)
CompanyName	Gets the name of the company or creator of the application containing the control. (Inherited from Control)
Container	Gets the IContainer that contains the Component. (Inherited from Component)
ContainsFocus	Gets a value indicating whether the control, or one of its child controls, currently has the input focus. (Inherited from Control)
ContextMenu	Gets or sets the shortcut menu associated with the control. (Inherited from Control)
ContextMenuStrip	Gets or sets the ContextMenuStrip associated with this control. (Inherited from Control)
Controls	Gets the collection of controls contained within the control. (Inherited from Control)
Created	Gets a value indicating whether the control has been created. (Inherited from Control)
CreateParams	Gets the required creation parameters when the control handle is created. (Inherited from Control)
Cursor	Gets or sets the cursor that is displayed when the mouse pointer is over the control. (Inherited from Control)

DataBindings	Gets the data bindings for the control. (Inherited from Control)
DataContext	Gets or sets the data context for the purpose of data binding. This is an ambient property. (Inherited from Control)
DataManipulator	Gets a DataManipulator object that provides methods and properties that handle data.
DataSource	Gets or sets the data source for the Chart object.
DefaultCursor	Gets or sets the default cursor for the control. (Inherited from Control)
DefaultImeMode	Gets the default Input Method Editor (IME) mode supported by the control. (Inherited from Control)
DefaultMargin	Gets the space, in pixels, that is specified by default between controls. (Inherited from Control)
DefaultMaximumSize	Gets the length and height, in pixels, that is specified as the default maximum size of a control. (Inherited from Control)
DefaultMinimumSize	Gets the length and height, in pixels, that is specified as the default minimum size of a control. (Inherited from Control)
DefaultPadding	Gets the internal spacing, in pixels, of the contents of a control. (Inherited from Control)
DefaultSize	Gets the default control size.
DesignMode	Gets a value that indicates whether the Component is currently in design mode. (Inherited from Component)
DeviceDpi	Gets the DPI value for the display device where the control is currently being displayed. (Inherited from Control)
DisplayRectangle	Gets the rectangle that represents the display area of the control. (Inherited from Control)
Disposing	Gets a value indicating whether the base Control class is in the process of disposing. (Inherited from Control)
Dock	Gets or sets which control borders are docked to its parent

	control and determines how a control is resized with its parent. (Inherited from Control)
DoubleBuffered	Gets or sets a value indicating whether this control should redraw its surface using a secondary buffer to reduce or prevent flicker. (Inherited from Control)
Enabled	Gets or sets a value indicating whether the control can respond to user interaction. (Inherited from Control)
Events	Gets the list of event handlers that are attached to this Component. (Inherited from Component)
Focused	Gets a value indicating whether the control has input focus. (Inherited from Control)
Font	Gets the font properties of the control.
FontHeight	Gets or sets the height of the font of the control. (Inherited from Control)
ForeColor	Gets or sets the text color of the Chart control.
Handle	Gets the window handle that the control is bound to. (Inherited from Control)
HasChildren	Gets a value indicating whether the control contains one or more child controls. (Inherited from Control)
Height	Gets or sets the height of the control. (Inherited from Control)
Images	Gets a NamedImagesCollection object that stores NamedImage objects for the chart.
ImeMode	Gets or sets the Input Method Editor (IME) mode of the control. (Inherited from Control)
ImeModeBase	Gets or sets the IME mode of a control. (Inherited from Control)
InvokeRequired	Gets a value indicating whether the caller must call an invoke method when making method calls to the control because the caller is on a different thread than the one the control was created on. (Inherited from Control)
IsAccessible	Gets or sets a value indicating whether the control is visible to

	accessibility applications. (Inherited from Control)
IsAncestorSiteInDesignMode	Indicates if one of the Ancestors of this control is sited and that site in DesignMode. This property is read-only. (Inherited from Control)
IsDisposed	Gets a value indicating whether the control has been disposed of. (Inherited from Control)
IsHandleCreated	Gets a value indicating whether the control has a handle associated with it. (Inherited from Control)
IsMirrored	Gets a value indicating whether the control is mirrored. (Inherited from Control)
IsSoftShadows	Gets or sets a flag that determines if a smooth gradient is applied when shadows are drawn.
LayoutEngine	Gets a cached instance of the control's layout engine. (Inherited from Control)
Left	Gets or sets the distance, in pixels, between the left edge of the control and the left edge of its container's client area. (Inherited from Control)
Legends	Gets or sets a LegendCollection that stores all Legend objects used by the Chart control.
Location	Gets or sets the coordinates of the upper-left corner of the control relative to the upper-left corner of its container. (Inherited from Control)
Margin	Gets or sets the space between controls. (Inherited from Control)
MaximumSize	Gets or sets the size that is the upper limit that GetPreferredSize(Size) can specify. (Inherited from Control)
MinimumSize	Gets or sets the size that is the lower limit that GetPreferredSize(Size) can specify. (Inherited from Control)
Name	Gets or sets the name of the control. (Inherited from Control)
Padding	Gets or sets padding within the control. (Inherited from Control)

Palette	Gets or sets the palette for the Chart control.
PaletteCustomColors	Gets or sets an array of custom palette colors.
Parent	Gets or sets the parent container of the control. (Inherited from Control)
PreferredSize	Gets the size of a rectangular area into which the control can fit. (Inherited from Control)
Printing	Gets a read-only PrintingManager object used for printing a chart.
ProductName	Gets the product name of the assembly containing the control. (Inherited from Control)
ProductVersion	Gets the version of the assembly containing the control. (Inherited from Control)
RecreatingHandle	Gets a value indicating whether the control is currently re-creating its handle. (Inherited from Control)
Region	Gets or sets the window region associated with the control. (Inherited from Control)
RenderingDpiX	Gets or sets the horizontal resolution of the Chart renderer.
RenderingDpiY	Gets or sets the vertical resolution of the Chart renderer.
RenderRightToLeft	**Obsolete.** This property is now obsolete. (Inherited from Control)
ResizeRedraw	Gets or sets a value indicating whether the control redraws itself when resized. (Inherited from Control)
Right	Gets the distance, in pixels, between the right edge of the control and the left edge of its container's client area. (Inherited from Control)
RightToLeft	Gets or sets a value indicating whether control's elements are aligned to support locales using right-to-left fonts. (Inherited from Control)
ScaleChildren	Gets a value that determines the scaling of child controls. (Inherited from Control)
Serializer	Gets a ChartSerializer object that is used for chart serialization.
Series	Gets a SeriesCollection object, which contains Series objects.

ShowFocusCues	Gets a value indicating whether the control should display focus rectangles. (Inherited from Control)
ShowKeyboardCues	Gets a value indicating whether the user interface is in the appropriate state to show or hide keyboard accelerators. (Inherited from Control)
Site	Gets or sets the site of the control. (Inherited from Control)
Size	Gets or sets the size of the Chart control.
SuppressExceptions	Gets or sets a flag that determines whether non-critical exceptions should be suppressed.
TabIndex	Gets or sets the tab order of the control within its container. (Inherited from Control)
TabStop	Gets or sets a value indicating whether the user can give the focus to this control using the TAB key. (Inherited from Control)
Tag	Gets or sets the object that contains data about the control. (Inherited from Control)
Text	Gets or sets the text associated with this control. (Inherited from Control)
TextAntiAliasingQuality	Gets or sets the TextAntiAliasingQuality type to use when applying anti-aliasing to text.
Titles	Gets or sets a TitleCollection object that is used to store all Title objects used by the Chart control.
Top	Gets or sets the distance, in pixels, between the top edge of the control and the top edge of its container's client area. (Inherited from Control)
TopLevelControl	Gets the parent control that is not parented by another Windows Forms control. Typically, this is the outermost Form that the control is contained in. (Inherited from Control)
UseWaitCursor	Gets or sets a value indicating whether to use the wait cursor for the current control and all child controls. (Inherited from Control)
Visible	Gets or sets a value indicating whether the control and all its child controls are displayed. (Inherited from Control)

Width	Gets or sets the width of the control. (Inherited from Control)
WindowTarget	This property is not relevant for this class. (Inherited from Control)

Methods

AccessibilityNotifyClients(AccessibleEvents, Int32)	Notifies the accessibility client applications of the specified AccessibleEvents for the specified child control. (Inherited from Control)
AccessibilityNotifyClients(AccessibleEvents, Int32, Int32)	Notifies the accessibility client applications of the specified AccessibleEvents for the specified child control . (Inherited from Control)
AlignDataPointsByAxisLabel()	Aligns data points along the X axis using their axis labels. Applicable when multiple series are indexed and their X-values are strings.
AlignDataPointsByAxisLabel(PointSortOrder)	Aligns data points using their axis labels. All series in the chart are aligned, using the specified sort order.
AlignDataPointsByAxisLabel(String)	Aligns data points from different series along the X axis using their axis labels. The specified series in the chart are aligned using an ascending sort order.
AlignDataPointsByAxisLabel(String, PointSortOrder)	Aligns data points using their axis labels.
ApplyPaletteColors()	Sets the automatically assigned series and data point colors, to allow programmatic access at run time.
BeginInit()	Signals to the object that initialization is starting.
BeginInvoke(Action)	Executes the specified delegate asynchronously on the thread that the control's underlying handle was created on. (Inherited from Control)
BeginInvoke(Delegate)	Executes the specified delegate asynchronously on the thread that the control's underlying handle was created on. (Inherited from Control)
BeginInvoke(Delegate, Object[])	Executes the specified delegate asynchronously with the specified arguments, on the thread that the control's underlying handle was created on. (Inherited from Control)
BringToFront()	Brings the control to the front of the z-order. (Inherited from Control)

Contains(Control)	Retrieves a value indicating whether the specified control is a child of the control. (Inherited from Control)
CreateAccessibilityInstance()	When overridden in a derived class, returns the custom AccessibleObject for the entire chart.
CreateControl()	Forces the creation of the visible control, including the creation of the handle and any visible child controls. (Inherited from Control)
CreateControlsInstance()	Creates a new instance of the control collection for the control. (Inherited from Control)
CreateGraphics()	Creates the Graphics for the control. (Inherited from Control)
CreateHandle()	Creates a handle for the control. (Inherited from Control)
CreateObjRef(Type)	Creates an object that contains all the relevant information required to generate a proxy used to communicate with a remote object. (Inherited from MarshalByRefObject)
DataBind()	Data binds the Chart control to a data source.
DataBindCross Table(IEnumerable, String, String, String, String)	Data binds a chart to the table, with one series created per unique value in a given column.
DataBindCross Table(IEnumerable, String, String, String, String, PointSort Order)	Data binds a chart to the table, with one series created per unique value in a given column.
DataBindTable(IEnumerable)	Automatically creates and binds series data to the specified data table.
DataBindTable(IEnumerable, String)	Automatically creates and binds series data to the specified data table, and optionally populates X-values.
DefWndProc(Message)	Sends the specified message to the default window procedure. (Inherited from Control)
DestroyHandle()	Destroys the handle associated with the control. (Inherited from Control)
Dispose()	Releases all resources used by the Component. (Inherited from Component)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.

DoDragDrop(Object, DragDropEffects)	Begins a drag-and-drop operation. (Inherited from Control)
DoDragDrop(Object, DragDropEffects, Bitmap, Point, Boolean)	Begins a drag operation. (Inherited from Control)
DrawToBitmap(Bitmap, Rectangle)	Supports rendering to the specified bitmap. (Inherited from Control)
EndInit()	Signals to the Chart object that initialization is complete.
EndInvoke(IAsyncResult)	Retrieves the return value of the asynchronous operation represented by the IAsyncResult passed. (Inherited from Control)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindForm()	Retrieves the form that the control is on. (Inherited from Control)
Focus()	Sets input focus to the control. (Inherited from Control)
GetAccessibilityObjectById(Int32)	Retrieves the specified AccessibleObject. (Inherited from Control)
GetAutoSizeMode()	Retrieves a value indicating how a control will behave when its AutoSize property is enabled. (Inherited from Control)
GetChartElementOutline(Object, ChartElementType)	Returns the chart element outline.
GetChildAtPoint(Point)	Retrieves the child control that is located at the specified coordinates. (Inherited from Control)
GetChildAtPoint(Point, GetChildAtPointSkip)	Retrieves the child control that is located at the specified coordinates, specifying whether to ignore child controls of a certain type. (Inherited from Control)
GetContainerControl()	Returns the next ContainerControl up the control's chain of parent controls. (Inherited from Control)
GetHashCode()	Serves as the default hash function. (Inherited from Object)

GetLifetimeService()	Retrieves the current lifetime service object that controls the lifetime policy for this instance. (Inherited from MarshalByRefObject)
GetNextControl(Control, Boolean)	Retrieves the next control forward or back in the tab order of child controls. (Inherited from Control)
GetPreferredSize(Size)	Retrieves the size of a rectangular area into which a control can be fitted. (Inherited from Control)
GetScaledBounds(Rectangle, SizeF, BoundsSpecified)	Retrieves the bounds within which the control is scaled. (Inherited from Control)
GetService(Type)	Returns the requested chart service.
GetStyle(ControlStyles)	Retrieves the value of the specified control style bit for the control. (Inherited from Control)
GetTopLevel()	Determines if the control is a top-level control. (Inherited from Control)
GetType()	Gets the Type of the current instance. (Inherited from Object)
Hide()	Conceals the control from the user. (Inherited from Control)
HitTest(Int32, Int32)	Determines the chart element, if any, that is located at a point defined by the given X and Y coordinates.
HitTest(Int32, Int32, Boolean)	Determines the chart element, if any, that is located at a point defined by given X and Y coordinates. Transparent elements can optionally be ignored.
HitTest(Int32, Int32, Boolean, ChartElementType[])	Determines whether a chart element that is one of the specified types is located at a point defined by the given X and Y coordinates.
HitTest(Int32, Int32, Chart ElementType)	Determines if a chart element of a given type is located at a point defined by given X and Y coordinates.
InitializeLifetimeService()	Obtains a lifetime service object to control the lifetime policy for this instance. (Inherited from MarshalByRefObject)
InitLayout()	Called after the control has been added to another container. (Inherited from Control)
Invalidate()	Invalidates the entire surface of the Chart and causes the Chart

	control to be redrawn.
Invalidate(Boolean)	Invalidates a specific region of the control and causes a paint message to be sent to the control. Optionally, invalidates the child controls assigned to the control. (Inherited from Control)
Invalidate(Rectangle)	Invalidates the specified area of the Chart control.
Invalidate(Rectangle, Boolean)	Invalidates the specified region of the control (adds it to the control's update region, which is the area that will be repainted at the next paint operation), and causes a paint message to be sent to the control. Optionally, invalidates the child controls assigned to the control. (Inherited from Control)
Invalidate(Region)	Invalidates the specified region of the control (adds it to the control's update region, which is the area that will be repainted at the next paint operation), and causes a paint message to be sent to the control. (Inherited from Control)
Invalidate(Region, Boolean)	Invalidates the specified region of the control (adds it to the control's update region, which is the area that will be repainted at the next paint operation), and causes a paint message to be sent to the control. Optionally, invalidates the child controls assigned to the control. (Inherited from Control)
Invoke(Action)	Executes the specified delegate on the thread that owns the control's underlying window handle. (Inherited from Control)
Invoke(Delegate)	Executes the specified delegate on the thread that owns the control's underlying window handle. (Inherited from Control)
Invoke(Delegate, Object[])	Executes the specified delegate, on the thread that owns the control's underlying window handle, with the specified list of arguments. (Inherited from Control)
Invoke<T>(Func<T>)	Executes the specified delegate on the thread that owns the control's underlying window handle. (Inherited from Control)
InvokeGotFocus(Control, EventArgs)	Raises the GotFocus event for the specified control. (Inherited from Control)
InvokeLostFocus(Control, EventArgs)	Raises the LostFocus event for the specified control. (Inherited from Control)

InvokeOnClick(Control, EventArgs)	Raises the Click event for the specified control. (Inherited from Control)
InvokePaint(Control, PaintEventArgs)	Raises the Paint event for the specified control. (Inherited from Control)
InvokePaintBackground(Control, PaintEventArgs)	Raises the `PaintBackground` event for the specified control. (Inherited from Control)
IsInputChar(Char)	Determines if a character is an input character that the control recognizes. (Inherited from Control)
IsInputKey(Keys)	Determines whether the specified key is a regular input key or a special key that requires preprocessing. (Inherited from Control)
LoadTemplate(Stream)	Loads a template into the Chart control from an image stream.
LoadTemplate(String)	Loads a template with the specified filename from the disk.
LogicalToDeviceUnits(Int32)	Converts a Logical DPI value to its equivalent DeviceUnit DPI value. (Inherited from Control)
LogicalToDeviceUnits(Size)	Transforms a size from logical to device units by scaling it for the current DPI and rounding down to the nearest integer value for width and height. (Inherited from Control)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
MemberwiseClone(Boolean)	Creates a shallow copy of the current MarshalByRefObject object. (Inherited from MarshalByRefObject)
NotifyInvalidate(Rectangle)	Raises the Invalidated event with a specified region of the control to invalidate. (Inherited from Control)
OnAutoSizeChanged(EventArgs)	Raises the AutoSizeChanged event. (Inherited from Control)
OnBackColorChanged(EventArgs)	Raises the BackColorChanged event. (Inherited from Control)
OnBackgroundImageChanged(EventArgs)	Raises the BackgroundImageChanged event. (Inherited from Control)
OnBackgroundImageLayoutChanged(EventArgs)	Raises the BackgroundImageLayoutChanged event. (Inherited from Control)

OnBindingContext Changed(EventArgs)	Raises the BindingContextChanged event. (Inherited from Control)
OnCausesValidation Changed(EventArgs)	Raises the CausesValidationChanged event. (Inherited from Control)
OnChangeUICues(UICues EventArgs)	Raises the ChangeUICues event. (Inherited from Control)
OnClick(EventArgs)	Raises the Click event. (Inherited from Control)
OnClientSizeChanged(Event Args)	Raises the ClientSizeChanged event. (Inherited from Control)
OnContextMenu Changed(EventArgs)	Raises the ContextMenuChanged event. (Inherited from Control)
OnContextMenuStrip Changed(EventArgs)	Raises the ContextMenuStripChanged event. (Inherited from Control)
OnControlAdded(Control EventArgs)	Raises the ControlAdded event. (Inherited from Control)
OnControlRemoved(Control EventArgs)	Raises the ControlRemoved event. (Inherited from Control)
OnCreateControl()	Raises the CreateControl() method. (Inherited from Control)
OnCursorChanged(EventArgs)	Overrides the OnCursorChanged(EventArgs) method and raises the CursorPositionChanging and CursorPositionChanged events.
OnCustomize()	Raises the Customize event.
OnCustomizeLegend(Legend ItemsCollection, String)	Raises the CustomizeLegend event.
OnDataContext Changed(EventArgs)	(Inherited from Control)
OnDockChanged(EventArgs)	Raises the DockChanged event. (Inherited from Control)
OnDoubleClick(EventArgs)	Overrides the OnDoubleClick(EventArgs) method.
OnDpiChangedAfter Parent(EventArgs)	Raises the DpiChangedAfterParent event. (Inherited from Control)
OnDpiChangedBefore Parent(EventArgs)	Raises the DpiChangedBeforeParent event. (Inherited from Control)
OnDragDrop(DragEventArgs)	Raises the DragDrop event.

	(Inherited from Control)
OnDragEnter(DragEventArgs)	Raises the DragEnter event. (Inherited from Control)
OnDragLeave(EventArgs)	Raises the DragLeave event. (Inherited from Control)
OnDragOver(DragEventArgs)	Raises the DragOver event. (Inherited from Control)
OnEnabledChanged(EventArgs)	Raises the EnabledChanged event. (Inherited from Control)
OnEnter(EventArgs)	Raises the Enter event. (Inherited from Control)
OnFontChanged(EventArgs)	Raises the FontChanged event. (Inherited from Control)
OnForeColorChanged(EventArgs)	Raises the ForeColorChanged event. (Inherited from Control)
OnFormatNumber(FormatNumberEventArgs)	Raises the FormatNumber event.
OnGiveFeedback(GiveFeedbackEventArgs)	Raises the GiveFeedback event. (Inherited from Control)
OnGotFocus(EventArgs)	Raises the GotFocus event.
OnHandleCreated(EventArgs)	Raises the HandleCreated event. (Inherited from Control)
OnHandleDestroyed(EventArgs)	Raises the HandleDestroyed event. (Inherited from Control)
OnHelpRequested(HelpEventArgs)	Raises the HelpRequested event. (Inherited from Control)
OnImeModeChanged(EventArgs)	Raises the ImeModeChanged event. (Inherited from Control)
OnInvalidated(InvalidateEventArgs)	Raises the Invalidated event. (Inherited from Control)
OnKeyDown(KeyEventArgs)	Raises the KeyDown event. (Inherited from Control)
OnKeyPress(KeyPressEventArgs)	Raises the KeyPress event. (Inherited from Control)
OnKeyUp(KeyEventArgs)	Raises the KeyUp event.

	(Inherited from Control)
OnLayout(LayoutEventArgs)	Raises the Layout event. (Inherited from Control)
OnLeave(EventArgs)	Raises the Leave event. (Inherited from Control)
OnLocationChanged(EventArgs)	Overrides the OnLocationChanged(EventArgs) method.
OnLostFocus(EventArgs)	Raises the LostFocus event.
OnMarginChanged(EventArgs)	Raises the MarginChanged event. (Inherited from Control)
OnMouseCaptureChanged(EventArgs)	Raises the MouseCaptureChanged event. (Inherited from Control)
OnMouseClick(MouseEventArgs)	Raises the MouseClick event. (Inherited from Control)
OnMouseDoubleClick(MouseEventArgs)	Raises the MouseDoubleClick event. (Inherited from Control)
OnMouseDown(MouseEventArgs)	Overrides the OnMouseDown(MouseEventArgs) method.
OnMouseEnter(EventArgs)	Raises the MouseEnter event. (Inherited from Control)
OnMouseHover(EventArgs)	Raises the MouseHover event. (Inherited from Control)
OnMouseLeave(EventArgs)	Raises the MouseLeave event. (Inherited from Control)
OnMouseMove(MouseEventArgs)	Overrides the OnMouseMove(MouseEventArgs) method.
OnMouseUp(MouseEventArgs)	Overrides the OnMouseUp(MouseEventArgs) method.
OnMouseWheel(MouseEventArgs)	Raises the MouseWheel event. (Inherited from Control)
OnMove(EventArgs)	Raises the Move event. (Inherited from Control)
OnNotifyMessage(Message)	Notifies the control of Windows messages. (Inherited from Control)
OnPaddingChanged(Event	Raises the PaddingChanged event.

Args)	(Inherited from Control)
OnPaint(PaintEventArgs)	Overrides the OnPaint(PaintEventArgs) method.
OnPaintBackground(PaintEventArgs)	Overrides the OnPaintBackground(PaintEventArgs) method.
OnParentBackColorChanged(EventArgs)	Raises the BackColorChanged event when the BackColor property value of the control's container changes. (Inherited from Control)
OnParentBackgroundImageChanged(EventArgs)	Raises the BackgroundImageChanged event when the BackgroundImage property value of the control's container changes. (Inherited from Control)
OnParentBindingContextChanged(EventArgs)	Raises the BindingContextChanged event when the BindingContext property value of the control's container changes. (Inherited from Control)
OnParentChanged(EventArgs)	Raises the ParentChanged event. (Inherited from Control)
OnParentCursorChanged(EventArgs)	Raises the CursorChanged event. (Inherited from Control)
OnParentDataContextChanged(EventArgs)	(Inherited from Control)
OnParentEnabledChanged(EventArgs)	Raises the EnabledChanged event when the Enabled property value of the control's container changes. (Inherited from Control)
OnParentFontChanged(EventArgs)	Raises the FontChanged event when the Font property value of the control's container changes. (Inherited from Control)
OnParentForeColorChanged(EventArgs)	Raises the ForeColorChanged event when the ForeColor property value of the control's container changes. (Inherited from Control)
OnParentRightToLeftChanged(EventArgs)	Raises the RightToLeftChanged event when the RightToLeft property value of the control's container changes. (Inherited from Control)
OnParentVisibleChanged(EventArgs)	Raises the VisibleChanged event when the Visible property value of the control's container changes. (Inherited from Control)
OnPostPaint(ChartPaintEventArgs)	Raises the PostPaint event.

OnPrePaint(ChartPaintEventArgs)	Raises the PrePaint event.
OnPreviewKeyDown(PreviewKeyDownEventArgs)	Raises the PreviewKeyDown event. (Inherited from Control)
OnPrint(PaintEventArgs)	Raises the Paint event. (Inherited from Control)
OnQueryContinueDrag(QueryContinueDragEventArgs)	Raises the QueryContinueDrag event. (Inherited from Control)
OnRegionChanged(EventArgs)	Raises the RegionChanged event. (Inherited from Control)
OnResize(EventArgs)	Overrides the OnResize(EventArgs) method.
OnRightToLeftChanged(EventArgs)	Overrides the OnRightToLeftChanged(EventArgs) method.
OnSizeChanged(EventArgs)	Raises the SizeChanged event. (Inherited from Control)
OnStyleChanged(EventArgs)	Raises the StyleChanged event. (Inherited from Control)
OnSystemColorsChanged(EventArgs)	Overrides the OnSystemColorsChanged(EventArgs) method.
OnTabIndexChanged(EventArgs)	Raises the TabIndexChanged event. (Inherited from Control)
OnTabStopChanged(EventArgs)	Raises the TabStopChanged event. (Inherited from Control)
OnTextChanged(EventArgs)	Raises the TextChanged event. (Inherited from Control)
OnValidated(EventArgs)	Raises the Validated event. (Inherited from Control)
OnValidating(CancelEventArgs)	Raises the Validating event. (Inherited from Control)
OnVisibleChanged(EventArgs)	Raises the VisibleChanged event. (Inherited from Control)
PerformLayout()	Forces the control to apply layout logic to all its child controls. (Inherited from Control)
PerformLayout(Control, String)	Forces the control to apply layout logic to all its child controls. (Inherited from Control)

PointToClient(Point)	Computes the location of the specified screen point into client coordinates. (Inherited from Control)
PointToScreen(Point)	Computes the location of the specified client point into screen coordinates. (Inherited from Control)
PreProcessControl Message(Message)	Preprocesses keyboard or input messages within the message loop before they are dispatched. (Inherited from Control)
PreProcessMessage(Message)	Preprocesses keyboard or input messages within the message loop before they are dispatched. (Inherited from Control)
ProcessCmdKey(Message, Keys)	Processes a command key. (Inherited from Control)
ProcessDialogChar(Char)	Processes a dialog character. (Inherited from Control)
ProcessDialogKey(Keys)	Processes a dialog key. (Inherited from Control)
ProcessKeyEvent Args(Message)	Processes a key message and generates the appropriate control events. (Inherited from Control)
ProcessKeyMessage(Message)	Processes a keyboard message. (Inherited from Control)
ProcessKeyPreview(Message)	Previews a keyboard message. (Inherited from Control)
ProcessMnemonic(Char)	Processes a mnemonic character. (Inherited from Control)
RaiseDragEvent(Object, Drag EventArgs)	Raises the appropriate drag event. (Inherited from Control)
RaiseKeyEvent(Object, Key EventArgs)	Raises the appropriate key event. (Inherited from Control)
RaiseMouseEvent(Object, MouseEventArgs)	Raises the appropriate mouse event. (Inherited from Control)
RaisePaintEvent(Object, Paint EventArgs)	Raises the appropriate paint event. (Inherited from Control)
RecreateHandle()	Forces the re-creation of the handle for the control. (Inherited from Control)

RectangleToClient(Rectangle)	Computes the size and location of the specified screen rectangle in client coordinates. (Inherited from Control)
RectangleToScreen(Rectangle)	Computes the size and location of the specified client rectangle in screen coordinates. (Inherited from Control)
Refresh()	Forces the control to invalidate its client area and immediately redraw itself and any child controls.
RescaleConstantsForDpi(Int32, Int32)	Provides constants for rescaling the control when a DPI change occurs. (Inherited from Control)
ResetAutoValues()	Resets automatically calculated chart property values to "Auto".
ResetBackColor()	Resets the BackColor property to its default value. (Inherited from Control)
ResetBindings()	Causes a control bound to the BindingSource to reread all the items in the list and refresh their displayed values. (Inherited from Control)
ResetCursor()	Resets the Cursor property to its default value. (Inherited from Control)
ResetFont()	Resets the Font property to its default value. (Inherited from Control)
ResetForeColor()	Resets the ForeColor property to its default value. (Inherited from Control)
ResetImeMode()	Resets the ImeMode property to its default value. (Inherited from Control)
ResetMouseEventArgs()	Resets the control to handle the MouseLeave event. (Inherited from Control)
ResetRightToLeft()	Resets the RightToLeft property to its default value. (Inherited from Control)
ResetText()	Resets the Text property to its default value (Empty). (Inherited from Control)
ResumeLayout()	Resumes usual layout logic. (Inherited from Control)
ResumeLayout(Boolean)	Resumes usual layout logic, optionally forcing an immediate layout of pending layout requests. (Inherited from Control)
RtlTranslate	Converts the specified ContentAlignment to the appropriate

Alignment(ContentAlignment)	ContentAlignment to support right-to-left text. (Inherited from Control)
RtlTranslate Alignment(Horizontal Alignment)	Converts the specified HorizontalAlignment to the appropriate HorizontalAlignment to support right-to-left text. (Inherited from Control)
RtlTranslateAlignment(Left RightAlignment)	Converts the specified LeftRightAlignment to the appropriate LeftRightAlignment to support right-to-left text. (Inherited from Control)
RtlTranslateContent(Content Alignment)	Converts the specified ContentAlignment to the appropriate ContentAlignment to support right-to-left text. (Inherited from Control)
RtlTranslate Horizontal(Horizontal Alignment)	Converts the specified HorizontalAlignment to the appropriate HorizontalAlignment to support right-to-left text. (Inherited from Control)
RtlTranslateLeftRight(LeftRight Alignment)	Converts the specified LeftRightAlignment to the appropriate LeftRightAlignment to support right-to-left text. (Inherited from Control)
SaveImage(Stream, Chart ImageFormat)	Saves the chart image to the specified stream.
SaveImage(Stream, Image Format)	Saves the image to the specified stream.
SaveImage(String, ChartImage Format)	Saves the chart image to the specified file.
SaveImage(String, Image Format)	Saves an image to the specified file.
Scale(Single)	**Obsolete.** Scales the control and any child controls. (Inherited from Control)
Scale(Single, Single)	**Obsolete.** Scales the entire control and any child controls. (Inherited from Control)
Scale(SizeF)	Scales the control and all child controls by the specified scaling factor. (Inherited from Control)
ScaleBitmapLogical ToDevice(Bitmap)	Scales a logical bitmap value to it's equivalent device unit value when a DPI change occurs. (Inherited from Control)
ScaleControl(SizeF, Bounds	Scales a control's location, size, padding and margin.

Specified)	(Inherited from Control)
ScaleCore(Single, Single)	This method is not relevant for this class. (Inherited from Control)
Select()	Activates the control. (Inherited from Control)
Select(Boolean, Boolean)	Activates a child control. Optionally specifies the direction in the tab order to select the control from. (Inherited from Control)
SelectNextControl(Control, Boolean, Boolean, Boolean, Boolean)	Activates the next control. (Inherited from Control)
SendToBack()	Sends the control to the back of the z-order. (Inherited from Control)
SetAutoSizeMode(AutoSizeMode)	Sets a value indicating how a control will behave when its AutoSize property is enabled. (Inherited from Control)
SetBounds(Int32, Int32, Int32, Int32)	Sets the bounds of the control to the specified location and size. (Inherited from Control)
SetBounds(Int32, Int32, Int32, Int32, BoundsSpecified)	Sets the specified bounds of the control to the specified location and size. (Inherited from Control)
SetBoundsCore(Int32, Int32, Int32, Int32, BoundsSpecified)	Performs the work of setting the specified bounds of this control. (Inherited from Control)
SetClientSizeCore(Int32, Int32)	Sets the size of the client area of the control. (Inherited from Control)
SetStyle(ControlStyles, Boolean)	Sets a specified ControlStyles flag to either `true` or `false`. (Inherited from Control)
SetTopLevel(Boolean)	Sets the control as the top-level control. (Inherited from Control)
SetVisibleCore(Boolean)	Sets the control to the specified visible state. (Inherited from Control)
Show()	Displays the control to the user. (Inherited from Control)
SizeFromClientSize(Size)	Determines the size of the entire control from the height and width of its client area. (Inherited from Control)

SuspendLayout()	Temporarily suspends the layout logic for the control. (Inherited from Control)
ToString()	Returns a String containing the name of the Component, if any. This method should not be overridden. (Inherited from Component)
Update()	Causes the control to redraw the invalidated regions within its client area. (Inherited from Control)
UpdateAnnotations()	Updates the annotations in the Chart control.
UpdateBounds()	Updates the bounds of the control with the current size and location. (Inherited from Control)
UpdateBounds(Int32, Int32, Int32, Int32)	Updates the bounds of the control with the specified size and location. (Inherited from Control)
UpdateBounds(Int32, Int32, Int32, Int32, Int32, Int32)	Updates the bounds of the control with the specified size, location, and client size. (Inherited from Control)
UpdateCursor()	Updates the cursor in the Chart control.
UpdateStyles()	Forces the assigned styles to be reapplied to the control. (Inherited from Control)
UpdateZOrder()	Updates the control in its parent's z-order. (Inherited from Control)
WndProc(Message)	Processes Windows messages. (Inherited from Control)

Events

AnnotationPlaced	Occurs when the end-user places an annotation on the chart.
AnnotationPositionChanged	Occurs when the annotation position is changed.
AnnotationPositionChanging	Occurs when the annotation position is about to change.
AnnotationSelectionChanged	Occurs when a selection of the annotation is changed.
AnnotationTextChanged	Occurs when the annotation text is changed.
AutoSizeChanged	This event is not relevant for this class. (Inherited from Control)

AxisScrollBarClicked	Occurs when the axis scroll bar is clicked by the end-user.
AxisViewChanged	Occurs when the axis scale view position or size is changed.
AxisViewChanging	Occurs when the axis scale view position or size is about to change.
BackColorChanged	Occurs when the value of the BackColor property changes. (Inherited from Control)
BackgroundImageChanged	Occurs when the value of the BackgroundImage property changes. (Inherited from Control)
BackgroundImageLayout Changed	Occurs when the BackgroundImageLayout property changes. (Inherited from Control)
BindingContextChanged	Occurs when the value of the BindingContext property changes. (Inherited from Control)
CausesValidationChanged	Occurs when the value of the CausesValidation property changes. (Inherited from Control)
ChangeUICues	Occurs when the focus or keyboard user interface (UI) cues change. (Inherited from Control)
Click	Occurs when the control is clicked. (Inherited from Control)
ClientSizeChanged	Occurs when the value of the ClientSize property changes. (Inherited from Control)
ContextMenuChanged	Occurs when the value of the ContextMenu property changes. (Inherited from Control)
ContextMenuStripChanged	Occurs when the value of the ContextMenuStrip property changes. (Inherited from Control)
ControlAdded	Occurs when a new control is added to the Control.ControlCollection. (Inherited from Control)
ControlRemoved	Occurs when a control is removed from the Control.ControlCollection. (Inherited from Control)
CursorChanged	Occurs when the value of the Cursor property changes. (Inherited from Control)

CursorPositionChanged	Occurs when the cursor position is changed.
CursorPositionChanging	Occurs when the cursor position is about to change.
Customize	Occurs just before the chart image is drawn. Use this event to customize the chart picture.
CustomizeLegend	Occurs when the chart legend must be customized.
DataContextChanged	Occurs when the value of the DataContext property changes. (Inherited from Control)
Disposed	Occurs when the component is disposed by a call to the Dispose() method. (Inherited from Component)
DockChanged	Occurs when the value of the Dock property changes. (Inherited from Control)
DoubleClick	Occurs when the control is double-clicked. (Inherited from Control)
DpiChangedAfterParent	Occurs when the DPI setting for a control is changed programmatically after the DPI of its parent control or form has changed. (Inherited from Control)
DpiChangedBeforeParent	Occurs when the DPI setting for a control is changed programmatically before a DPI change event for its parent control or form has occurred. (Inherited from Control)
DragDrop	Occurs when a drag-and-drop operation is completed. (Inherited from Control)
DragEnter	Occurs when an object is dragged into the control's bounds. (Inherited from Control)
DragLeave	Occurs when an object is dragged out of the control's bounds. (Inherited from Control)
DragOver	Occurs when an object is dragged over the control's bounds. (Inherited from Control)
EnabledChanged	Occurs when the Enabled property value has changed. (Inherited from Control)
Enter	Occurs when the control is entered. (Inherited from Control)
FontChanged	Occurs when the Font property value changes. (Inherited from Control)

ForeColorChanged	Occurs when the ForeColor property value changes. (Inherited from Control)
FormatNumber	Occurs when a numeric value has to be converted to a string.
GetToolTipText	Occurs before showing the tooltip to get the tooltip text.
GiveFeedback	Occurs during a drag operation. (Inherited from Control)
GotFocus	Occurs when the control receives focus. (Inherited from Control)
HandleCreated	Occurs when a handle is created for the control. (Inherited from Control)
HandleDestroyed	Occurs when the control's handle is in the process of being destroyed. (Inherited from Control)
HelpRequested	Occurs when the user requests help for a control. (Inherited from Control)
ImeModeChanged	Occurs when the ImeMode property has changed. (Inherited from Control)
Invalidated	Occurs when a control's display requires redrawing. (Inherited from Control)
KeyDown	Occurs when a key is pressed while the control has focus. (Inherited from Control)
KeyPress	Occurs when a character. space or backspace key is pressed while the control has focus. (Inherited from Control)
KeyUp	Occurs when a key is released while the control has focus. (Inherited from Control)
Layout	Occurs when a control should reposition its child controls. (Inherited from Control)
Leave	Occurs when the input focus leaves the control. (Inherited from Control)
LocationChanged	Occurs when the Location property value has changed. (Inherited from Control)
LostFocus	Occurs when the control loses focus. (Inherited from Control)
MarginChanged	Occurs when the control's margin changes. (Inherited from Control)

MouseCaptureChanged	Occurs when the control loses mouse capture. (Inherited from Control)
MouseClick	Occurs when the control is clicked by the mouse. (Inherited from Control)
MouseDoubleClick	Occurs when the control is double clicked by the mouse. (Inherited from Control)
MouseDown	Occurs when the mouse pointer is over the control and a mouse button is pressed. (Inherited from Control)
MouseEnter	Occurs when the mouse pointer enters the control. (Inherited from Control)
MouseHover	Occurs when the mouse pointer rests on the control. (Inherited from Control)
MouseLeave	Occurs when the mouse pointer leaves the control. (Inherited from Control)
MouseMove	Occurs when the mouse pointer is moved over the control. (Inherited from Control)
MouseUp	Occurs when the mouse pointer is over the control and a mouse button is released. (Inherited from Control)
MouseWheel	Occurs when the mouse wheel moves while the control has focus. (Inherited from Control)
Move	Occurs when the control is moved. (Inherited from Control)
PaddingChanged	Occurs when the control's padding changes. (Inherited from Control)
Paint	Occurs when the control is redrawn. (Inherited from Control)
ParentChanged	Occurs when the Parent property value changes. (Inherited from Control)
PostPaint	Occurs when the chart element is painted.
PrePaint	Occurs when the chart element background is painted.
PreviewKeyDown	Occurs before the KeyDown event when a key is pressed while focus is on this control. (Inherited from Control)

QueryAccessibilityHelp	Occurs when AccessibleObject is providing help to accessibility applications. (Inherited from Control)
QueryContinueDrag	Occurs during a drag-and-drop operation and enables the drag source to determine whether the drag-and-drop operation should be canceled. (Inherited from Control)
RegionChanged	Occurs when the value of the Region property changes. (Inherited from Control)
Resize	Occurs when the control is resized. (Inherited from Control)
RightToLeftChanged	Occurs when the RightToLeft property value changes. (Inherited from Control)
SelectionRangeChanged	Occurs when the selection start position or end position is changed.
SelectionRangeChanging	Occurs when the selection start or end position is about to change.
SizeChanged	Occurs when the Size property value changes. (Inherited from Control)
StyleChanged	Occurs when the control style changes. (Inherited from Control)
SystemColorsChanged	Occurs when the system colors change. (Inherited from Control)
TabIndexChanged	Occurs when the TabIndex property value changes. (Inherited from Control)
TabStopChanged	Occurs when the TabStop property value changes. (Inherited from Control)
TextChanged	Occurs when the Text property value changes. (Inherited from Control)
Validated	Occurs when the control is finished validating. (Inherited from Control)
Validating	Occurs when the control is validating. (Inherited from Control)
VisibleChanged	Occurs when the Visible property value changes. (Inherited from Control)

Explicit Interface Implementations

IDropTarget.OnDragDrop(DragEventArgs)	Raises the DragDrop event. (Inherited from Control)
IDropTarget.OnDragEnter(DragEventArgs)	Raises the DragEnter event. (Inherited from Control)
IDropTarget.OnDragLeave(EventArgs)	Raises the DragLeave event. (Inherited from Control)
IDropTarget.OnDragOver(DragEventArgs)	Raises the DragOver event. (Inherited from Control)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Chart class.

C#

```
public Chart ();
```

Remarks

You can use this constructor to create a new instance of the Chart class at run time.

However, for simplicity, it is recommended that you create a Chart instance at design time.

This is the root object of the Chart control.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Annotations Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a collection that stores the chart annotations.

C#

```
public System.Windows.Forms.DataVisualization.Charting.AnnotationCollection
Annotations { get; }
```

Property Value

AnnotationCollection

An AnnotationCollection object that is used to store chart annotations.

Remarks

All of the chart annotations are stored in this collection property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AntiAliasing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that determines whether anti-aliasing is used when text and graphics are drawn.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.AntiAliasingStyles
AntiAliasing { get; set; }
```

Property Value

AntiAliasingStyles

An AntiAliasingStyles enumeration value that determines whether anti-aliasing is used when text and graphics are drawn.

Attributes BindableAttribute

Remarks

Anti-aliasing smoothens out "jagged edges" by padding pixels with intermediate colors, and is especially effective for smoothing out edges for charts that are not drawn as rectangles. For example, anti-aliasing is very effective for line charts and pie charts, but is not as effective for bar and column charts.

Anti-aliasing can be applied to text and/or graphics primitives such as rectangles, circles, and so forth.

Note that anti-aliasing will affect performance.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the Chart object.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value used to draw the background color of the chart. The default color is White.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set this property to any valid ARGB (alpha, red, green, blue) value. Note that for the Chart object, the alpha component cannot be set to a value other than 255. This also means that, for the Chart object only, a Transparent value cannot be used.

The BackColor value will be the first color used if you are using gradient colors for the background.

The alpha value controls the opacity of a color. If you set this color to "transparent" - that is, if you use an alpha value of 0 - no color will be assigned to the background of the title. As a result, the background will be transparent.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the orientation for the background gradient of a Chart control. Also determines whether a gradient is used.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration that specifies the orientation for the background gradient of a chart and whether a gradient is used. The default value is None.

Attributes BindableAttribute

Remarks

Use the BackGradientStyle property to determine whether a Chart control will display a gradient, and to determine the orientation of the gradient if one is used.

For a listing of available patterns, refer to the GradientStyle enumeration.

To use gradient colors:

Set the primary color of the chart background with the BackColor property.

Set the secondary color of the chart background with the BackSecondaryColor property.

Set this BackGradientStyle property to the desired gradient style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackgroundImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

The BackgroundImage property is not used. Use the BackImage property instead.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.Image BackgroundImage { get; set; }
```

Property Value

Image

An Image object.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatching style of the Chart control.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration that specifies the hatching style of the Chart control. The default value is None.

Attributes BindableAttribute

Remarks

The ChartHatchStyle enumeration specifies the hatch pattern used by a brush of type HatchBrush. The hatch pattern consists of a solid background color and lines drawn over the background.

For a listing of available hatching styles, refer to the ChartHatchStyle enumeration.

Note The color of hatching lines is determined by the BackSecondaryColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image of the Chart control.

C#

```
[System.ComponentModel.Bindable(true)]
public string BackImage { get; set; }
```

Property Value

String

A `string` value that represents the URL of an image file. The default value is an empty string.

Attributes BindableAttribute

Remarks

This property defines a path to a background image for a Chart legend, and can be an absolute or relative URL.

To position the image, use the BackImageAlignment and BackImageWrapMode properties.

You can also specify one of the colors displayed by the image as "transparent", by setting the BackImageTransparentColor property.

An image can originate from any valid graphics file supported by .NET Framework, including icons.

Background images take precedence over other background properties such as gradient color.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackImageAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image alignment used for the Unscaled drawing mode.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.ChartImageAlignmentStyle
BackImageAlignment { get; set; }
```

Property Value

ChartImageAlignmentStyle

A ChartImageAlignmentStyle enumeration value that specifies the background image alignment of the chart. The default value is TopLeft.

Attributes BindableAttribute

Remarks

This property determines the alignment of the background image for the Chart control; it is only applied when the BackImageWrapMode property is set to ChartImageWrapMode.Unscaled.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the Chart control that will be displayed as transparent.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that will be displayed as transparent when the chart image is drawn. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackImageWrapMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the drawing mode for the background image of the Chart control.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartImageWrapMode
BackImageWrapMode { get; set; }
```

Property Value

ChartImageWrapMode

A ChartImageWrapMode enumeration value that specifies the drawing mode for the background image of the chart. The default value is Tile.

Attributes BindableAttribute

Remarks

This property determines how the image is displayed, for example, tiled, scaled, and so forth.

If an image is scaled, it will be stretched to fit within the bounds of the chart. If the image is tiled, it is repeatedly drawn, starting from the top-left corner of the chart.

If the `BackImageWrapMode` property is set to ChartImageWrapMode.Unscaled, the location where the image is drawn is determined by the BackImageAlignment property setting.

For more information about the available wrapping modes, see the ChartImageWrapMode enumeration article.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color of the chart background.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value that represents the secondary color of the chart background. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The BackSecondaryColor property determines the secondary background color of many Chart objects, and is used when gradient colors are being applied.

You can set BackSecondaryColor to any valid ARGB (alpha, red, green, blue) value.

If hatching is used, the BackSecondaryColor property determines the color of the hatching lines for the chart.

To use gradient colors for the background of a chart:

- Set the primary color with the BackColor property.
- Set the secondary color with the BackSecondaryColor property.
- Set the BackGradientStyle property to the desired gradient style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the chart border.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderColor { get; set; }
```

Property Value

Color

A Color value used to draw the chart border.

Attributes BindableAttribute, BrowsableAttribute, TypeConverterAttribute

Remarks

To get or set the border color, use BorderlineColor instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the border.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that specifies the style of the border.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BorderlineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the border line.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderlineColor { get; set; }
```

Property Value

Color

A Color value used to draw the border line.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BorderlineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the border line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderlineDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that specifies the style of the border line.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BorderlineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of the border line.

C#

```
[System.ComponentModel.Bindable(true)]
public int BorderlineWidth { get; set; }
```

Property Value

Int32

An `integer` value that specifies the width of the border line.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BorderSkin Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a BorderSkin object, which provides border skin functionality for the Chart control.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.LegendConverter))]
public System.Windows.Forms.DataVisualization.Charting.BorderSkin BorderSkin
{ get; set; }
```

Property Value

BorderSkin

A BorderSkin object which provides border skin functionality for the chart.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Use this property to specify a border skin for the chart.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or set s the width of the chart border.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public int BorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that determines the border width, in pixels, of the chart.

Attributes BindableAttribute, BrowsableAttribute

Remarks

To get or set of the border width, use BorderlineWidth instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BuildNumber Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the build number of the Chart control.

C#

```
[System.ComponentModel.Browsable(false)]
public string BuildNumber { get; }
```

Property Value

String

A `string` that represents the build number of the chart control.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.ChartAreas Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a read-only ChartAreaCollection object that is used to store ChartArea objects.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartAreaCollection
ChartAreas { get; }
```

Property Value

ChartAreaCollection

A read-only ChartAreaCollection object that contains collection of ChartArea objects.

Attributes BindableAttribute

Remarks

This collection property lets you access the ChartAreaCollection object, which stores ChartArea objects and also provides methods and properties used to manipulate the collection.

ChartArea objects are used to plot one or more charts.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.DataManipulator Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a DataManipulator object that provides methods and properties that handle data.

C#

```
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.DataManipulator
DataManipulator { get; }
```

Property Value

DataManipulator

A DataManipulator object that provides methods and properties that handle data.

Attributes BrowsableAttribute

Remarks

Use this property at run-time to:

- Copy Y-values.
- Filter data.
- Group data.
- Replace missing data with empty points.
- Export data to a DataSet object.
- Sort data.
- Apply a financial formula to data.

Refer to the DataManipulator class overview for further details.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.DataSource Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the data source for the Chart object.

C#

```
[System.ComponentModel.Bindable(true)]
public object DataSource { get; set; }
```

Property Value

Object

An Object that represents the data source for the Chart object.

Attributes BindableAttribute

Remarks

The data points in a series can be bound to a data source at any time using the DataSource property.

To force the chart to bind to the data source, call the DataBind method.

The following is a list of objects that you can use as the data source:

- DataView
- Data readers (SQL, OleDB)
- DataSet
- DataTable
- Binding Source

- IDataSource
- Arrays
- Lists
- All Enumerable objects
- SqlCommand / OleDbCommand (DataSource data-binding only)
- SqlDataAdapter / OleDbDataAdapter (DataSource data-binding only)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.DefaultSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the default control size.

C#

```
protected override System.Drawing.Size DefaultSize { get; }
```

Property Value

Size

A Size object that represents the default size of the control.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the font properties of the control.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object that represents the text font of the control.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color of the Chart control.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value that specifies the text color of the chart.

Attributes BindableAttribute, BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to this class.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Images Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a NamedImagesCollection object that stores NamedImage objects for the chart.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.NamedImagesCollection
Images { get; }
```

Property Value

NamedImagesCollection

A NamedImagesCollection object that contains the collection of NamedImage objects.

Attributes BindableAttribute, BrowsableAttribute

Remarks

Use this property to add NamedImage objects to and remove NamedImage objects from the NamedImagesCollection object of the Chart. This collection property is available at run time only.

An image - that is, an object derived from type Image, such as Bitmap, that is created dynamically - can be stored in this collection, and then assigned at a later time to any image-related Chart property, such as BackImage.

When you assign a NamedImage object to an image-related property in the Chart class, use the name of the NamedImage object for the `string` value of the image-related property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.IsSoftShadows Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines if a smooth gradient is applied when shadows are drawn.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsSoftShadows { get; set; }
```

Property Value

Boolean

`true` if shadows are drawn using smoothing; otherwise, `false`. The default value is `true`.

Attributes BindableAttribute

Remarks

If the value of the `ShadowOffset` property for a series, data point, chart area or legend is nonzero, a shadow will be applied using a color set by their `ShadowColor` property.

If the value of the IsSoftShadows property is `true`, a smooth gradient will be applied to the shadow.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Legends Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a LegendCollection that stores all Legend objects used by the Chart control.

C#

```
public System.Windows.Forms.DataVisualization.Charting.LegendCollection
Legends { get; }
```

Property Value

LegendCollection

A LegendCollection object that contains the Legend objects used by the chart.

Remarks

This collection property stores all Legend objects used by the Chart control.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Palette Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the palette for the Chart control.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartColorPalette
Palette { get; set; }
```

Property Value

ChartColorPalette

A ChartColorPalette enumeration value that determines the palette to be used.

Attributes BindableAttribute

Remarks

A palette is used to automatically assign colors to data elements if both of the following conditions have been met:

- the data points themselves do not have colors assigned to them.
- the data series that the data points belong to also does not have an assigned color.

If no series or data point colors are explicitly specified, the colors used depend on the chart type and the Chart object's palette or the palette of an associated series, if one is specified.

Note

If the `Palette` property for both the **Chart** and **Series** objects is set to **ChartColorPalette.None**, and no color is assigned to the relevant data series or data points, the data points will not be drawn and will be displayed as "empty" areas.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.PaletteCustomColors Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an array of custom palette colors.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.ColorArrayConverter))]
public System.Drawing.Color[] PaletteCustomColors { get; set; }
```

Property Value

Color[]

An array of Color objects that represent the set of colors used for series on the chart.

Attributes TypeConverterAttribute

Remarks

If you need more than 16 colors or you want to apply your own colors to the chart, you must define a custom palette. A custom palette let you add your own colors in the order you want them to appear on the chart. This type of palette is especially helpful if the number of series in your chart is unknown at design time.

This property has no effect when the Palette property is *not* set to ChartColorPalette.None.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Printing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a read-only PrintingManager object used for printing a chart.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.PrintingManager
Printing { get; }
```

Property Value

PrintingManager

A PrintingManager object used for printing a chart.

Attributes BindableAttribute, BrowsableAttribute

Remarks

When a chart is printed, all chart elements except scrollbars will be printed.

The PrintingManager object contains the following methods, which are used for basic printing operations:

PageSetup: Displays a Page Setup dialog, which allows the end-user to select paper size, paper source, page orientation, and so forth.

PrintPreview: Displays a Print Preview dialog, which allows the end-user to zoom, print the chart, and so forth.

Print: Prints the Chart; the Print dialog can optionally be displayed prior to printing.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.RenderingDpiX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the horizontal resolution of the Chart renderer.

C#

```
[System.ComponentModel.Browsable(false)]
public double RenderingDpiX { get; set; }
```

Property Value

Double

A `double` value that represents the horizontal resolution of the chart renderer.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.RenderingDpiY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the vertical resolution of the Chart renderer.

C#

```
[System.ComponentModel.Browsable(false)]
public double RenderingDpiY { get; set; }
```

Property Value

Double

A `double` value that represents the vertical resolution of the chart renderer.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Serializer Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a ChartSerializer object that is used for chart serialization.

C#

```
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.ChartSerializer
Serializer { get; }
```

Property Value

ChartSerializer

A ChartSerializer object that is used for chart serialization.

Attributes BrowsableAttribute

Remarks

Use this property at either design time or run time to serialize the chart.

For a detailed discussion of the ChartSerializer class, refer to the class overview.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Series Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a SeriesCollection object, which contains Series objects.

C#

```
public System.Windows.Forms.DataVisualization.Charting.SeriesCollection
Series { get; }
```

Property Value

SeriesCollection

A SeriesCollection object which contains Series objects.

Remarks

This collection property allows you to access the SeriesCollection class, which stores Series objects and also provides methods and properties used to manipulate this collection.

Series objects store DataPoint objects, as well as data attributes.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Size Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the Chart control.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.Size Size { get; set; }
```

Property Value

Size

A Size object that represents the size of the control.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.SuppressExceptions Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether non-critical exceptions should be suppressed.

```
C#
public bool SuppressExceptions { get; set; }
```

Property Value

Boolean

`true` if non-critical exceptions should be suppressed; otherwise, `false`. The default value is `false`.

Remarks

If you set this property to `true`, non-critical exceptions will be suppressed.

When the Chart control cannot properly plot the chart using the data and other settings provided, an exception will be thrown. Although displaying exceptions is useful during application testing, they should not be shown when the application is released.

> **Note**
>
> Setting this property to `true` may hide serious issues in the user application source code. This setting could also cause a chart image to be generated that might misrepresent your data. It is highly recommended that this mode be only used in Release builds.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.TextAntiAliasingQuality Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the TextAntiAliasingQuality type to use when applying anti-aliasing to text.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.TextAntiAliasingQuality
TextAntiAliasingQuality { get; set; }
```

Property Value

TextAntiAliasingQuality

A TextAntiAliasingQuality enumeration value used to apply anti-aliasing to text. The default value is High.

Attributes BindableAttribute

Remarks

If the anti-aliasing is applied to text, the Chart control will use this property to determine the type of textual anti-aliasing to apply.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Titles Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a TitleCollection object that is used to store all Title objects used by the Chart control.

C#

```
public System.Windows.Forms.DataVisualization.Charting.TitleCollection
Titles { get; }
```

Property Value

TitleCollection

A TitleCollection object that is used to store all Title objects used by the chart control.

Remarks

The Titles property stores all Title objects used by the Chart control.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AlignDataPointsByAxisLabel Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Aligns data points using their axis labels.

Overloads

AlignDataPointsByAxisLabel()	Aligns data points along the X axis using their axis labels. Applicable when multiple series are indexed and their X-values are strings.
AlignDataPointsByAxisLabel(String)	Aligns data points from different series along the X axis using their axis labels. The specified series in the chart are aligned using an ascending sort order.
AlignDataPointsByAxisLabel(PointSortOrder)	Aligns data points using their axis labels. All series in the chart are aligned, using the specified sort order.
AlignDataPointsByAxisLabel(String, PointSortOrder)	Aligns data points using their axis labels.

AlignDataPointsByAxisLabel()

Aligns data points along the X axis using their axis labels. Applicable when multiple series are indexed and their X-values are strings.

C#

```
public void AlignDataPointsByAxisLabel ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AlignDataPointsByAxisLabel(String)

Aligns data points from different series along the X axis using their axis labels. The specified series in the chart are aligned using an ascending sort order.

C#

```
public void AlignDataPointsByAxisLabel (string series);
```

Parameters

`series` String

A comma-separated list of series that will have their data points aligned to the X axis using the data point axis labels.

Remarks

This method is used to align data points from different series along the X axis using their axis labels, and is applicable when:

- Multiple series are displayed and are indexed; that is, all data points have an XValue of 0, or the IsXValueIndexed property is `true`.
- All data points have non-empty axis labels, which are unique to their Series. If they are not unique, an exception will be thrown.

This situation is very common when data-binding results in the creation of multiple series, and `string` values are used for X-axis values.

Two issues arise when multiple, indexed series are plotted, and the X-values are set using `string` values:

- It cannot be guaranteed that data points from different series with the same string X-values will be plotted at the same index along the X axis.

- Only the axis label for the data points in the first series will be displayed, even though data points from other series with the same index have axis labels too.

By default, points are plotted in the order in which they are stored in their series. To specify a sort order, use an `AlignDataPointsByAxisLabel` method overload that has a `sortingOrder` parameter.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AlignDataPointsByAxisLabel(PointSortOrder)

Aligns data points using their axis labels. All series in the chart are aligned, using the specified sort order.

C#

```
public void AlignDataPointsByAxisLabel
(System.Windows.Forms.DataVisualization.Charting.PointSortOrder
sortingOrder);
```

Parameters

`sortingOrder` PointSortOrder

A PointSortOrder object that indicates if ascending or descending sort order is used on axis labels, which in turn determines the order by which points that previously occupied the same axis space are displayed.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AlignDataPointsByAxisLabel(String, PointSortOrder)

Aligns data points using their axis labels.

```
C#

public void AlignDataPointsByAxisLabel (string series,
System.Windows.Forms.DataVisualization.Charting.PointSortOrder
sortingOrder);
```

Parameters

`series` String

A comma-separated list of series that should be aligned by axis label.

`sortingOrder` PointSortOrder

A PointSortOrder enumeration value that determines if ascending or descending sort order is used on axis labels, which in turn determines the order by which points that previously occupied the same axis space are displayed.

Remarks

This method is used to align data points from different series along the X axis using their axis labels, and is applicable when:

- Multiple series are displayed and are indexed; that is, all data points have an XValue of 0, or the IsXValueIndexed property is `true`.
- All data points have non-empty axis labels, which are unique to their Series. If they are not unique, an exception will be thrown.

This situation is very common when data-binding results in the creation of multiple series, and `string` values are used for X-axis values.

Two issues arise when multiple, indexed series are plotted, and the X-values are set using `string` values:

- It cannot be guaranteed that data points from different series with the same string X-values will be plotted at the same index along the X axis.

- Only the axis label for the data points in the first series will be displayed, even though data points from other series with the same index have axis labels too.

By default, points are plotted in the order in which they are stored in their series. To specify a sort order, use an `AlignDataPointsByAxisLabel` method overload that has a `sortingOrder` parameter.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.ApplyPaletteColors Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets the automatically assigned series and data point colors, to allow programmatic access at run time.

C#

```
public void ApplyPaletteColors ();
```

Remarks

When the Chart colors are automatically assigned at run time, there is no way to know what the colors will be prior to the time when the chart rendered; the `Color` property of an automatically assigned value will return Empty at this time.

If you call the ApplyPaletteColors method, the colors for the series and the data points will be set, which allows for programmatic access.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.BeginInit Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Signals to the object that initialization is starting.

C#

```
public void BeginInit ();
```

Implements

BeginInit()

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.CreateAccessibilityInstance Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

When overridden in a derived class, returns the custom AccessibleObject for the entire chart.

C#

```
protected override System.Windows.Forms.AccessibleObject
CreateAccessibilityInstance ();
```

Returns

AccessibleObject

The AccessibleObject for the chart.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.DataBind Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Data binds the Chart control to a data source.

C#

```
public void DataBind ();
```

Remarks

This method will bind the Chart control to the specified data source.

In cases where a data source is set to a chart and no other data operations are required, the DataBind method does not have to be explicitly called. In these cases, the Chart itself will data bind to the data source prior to being rendered.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.DataBindCrossTable Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Data binds a chart to the table, with one series created per unique value in a given column.

Overloads

DataBindCrossTable(IEnumerable, String, String, String, String)	Data binds a chart to the table, with one series created per unique value in a given column.
DataBindCrossTable(IEnumerable, String, String, String, String, PointSortOrder)	Data binds a chart to the table, with one series created per unique value in a given column.

DataBindCrossTable(IEnumerable, String, String, String, String)

Data binds a chart to the table, with one series created per unique value in a given column.

C#

```
public void DataBindCrossTable (System.Collections.IEnumerable
dataSource, string seriesGroupByField, string xField, string yFields,
string otherFields);
```

Parameters

`dataSource` IEnumerable

The data source that is data bound by a chart.

`seriesGroupByField` String

The name of the field used to group data into the series.

`xField` String

The name of the field for X values.

`yFields` String

A comma-separated list of name(s) of the field(s) for Y value(s).

`otherFields` String

The other data point properties that can be bound.

Remarks

Series will be automatically added to the chart, depending on the number of unique values in the given `seriesGroupByField` column of the data source; in other words, the creation of series depends upon grouping of a column.

The data source can be a OleDbDataReader, SqlDataReader or a DataView object. Note that all collections that implement the IEnumerable interface can be used.

Other data point properties besides X and Y-values can be bound. The format is: PointProperty=Field[{Format}] [,PointProperty= Field[{Format}]]. For example:"Tooltip=Price{C1},Href=WebSiteName".

These data point properties are:

- DataPointCustomProperties.AxisLabel
- DataPointCustomProperties.ToolTip
- DataPointCustomProperties.Label
- DataPointCustomProperties.LegendText
- DataPointCustomProperties.LegendToolTip
- DataPointCustomProperties.CustomProperties, which are the names of the custom properties.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataBindCrossTable(IEnumerable, String,

String, String, String, PointSortOrder)

Data binds a chart to the table, with one series created per unique value in a given column.

```
C#

public void DataBindCrossTable (System.Collections.IEnumerable
dataSource, string seriesGroupByField, string xField, string yFields,
string otherFields,
System.Windows.Forms.DataVisualization.Charting.PointSortOrder
sortingOrder);
```

Parameters

`dataSource` IEnumerable

The data source that is data bound by a chart.

`seriesGroupByField` String

The name of the field used to group data into the series.

`xField` String

The name of the field for X values.

`yFields` String

A comma-separated list of name(s) of the field(s) for Y value(s).

`otherFields` String

The other data point properties that can be bound.

`sortingOrder` PointSortOrder

The order in which the series will be sorted by group field values.

Remarks

Series will be automatically added to the chart, depending on the number of unique values in the given `seriesGroupByField` column of the data source; in other words, the creation of series depends upon grouping of a column.

The data source can be a OleDbDataReader, SqlDataReader or a DataView object.

Note that all collections that implement the IEnumerable interface can be used.

Other data point properties besides X and Y-values can be bound. The format is: PointProperty=Field[{Format}] [,PointProperty= Field[{Format}]]. For example:"Tooltip=Price{C1},Href=WebSiteName".

These data point properties are:

- DataPointCustomProperties.AxisLabel
- DataPointCustomProperties.ToolTip
- DataPointCustomProperties.Label
- DataPointCustomProperties.LegendText
- DataPointCustomProperties.LegendToolTip
- CustomProperties, which are the names of the custom properties

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.DataBindTable Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Automatically creates and binds series data to the specified data table.

Overloads

DataBindTable(IEnumerable)	Automatically creates and binds series data to the specified data table.
DataBindTable(IEnumerable, String)	Automatically creates and binds series data to the specified data table, and optionally populates X-values.

DataBindTable(IEnumerable)

Automatically creates and binds series data to the specified data table.

C#

```
public void DataBindTable (System.Collections.IEnumerable dataSource);
```

Parameters

`dataSource` IEnumerable

The data source that is data bound by a chart, which can be any IEnumerable object.

Remarks

Each column of the table becomes a Y-value for a series, with one series per column.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataBindTable(IEnumerable, String)

Automatically creates and binds series data to the specified data table, and optionally populates X-values.

C#

```
public void DataBindTable (System.Collections.IEnumerable dataSource,
string xField);
```

Parameters

`dataSource` IEnumerable

The data source that is data bound by a chart, which can be any IEnumerable object.

`xField` String

The name of the field used for the series X-values.

Remarks

Each column of the table becomes a Y-value for a series, with one series per column.

The series' X-value field can also be provided.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.EndInit Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Signals to the Chart object that initialization is complete.

C#

```
public void EndInit ();
```

Implements

EndInit()

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.GetChartElementOutline(Object, ChartElementType) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the chart element outline.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartElementOutline
GetChartElementOutline (object element,
System.Windows.Forms.DataVisualization.Charting.ChartElementType
elementType);
```

Parameters

element Object

The Chart object.

elementType ChartElementType

The type of the element.

Returns

ChartElementOutline

A ChartElementOutline object that contains:

- An array of points in absolute coordinates that can be used as outline markers around this chart element.
- A GraphicsPath object for drawing an outline around this chart element.

Remarks

If the `chartElement` is not part of the chart, or the `elementType` cannot be combined with `chartElement`, the result will contain an empty array of marker points. The marker points are sorted clockwise.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.GetService(Type) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the requested chart service.

C#

```
public object GetService (Type serviceType);
```

Parameters

`serviceType` Type

The type of requested chart service.

Returns

Object

An Object that represents the service type, or `null` if the service cannot be found.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.HitTest Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Determines the chart element, if any, that is located at a point defined by the given X and Y coordinates.

Overloads

HitTest(Int32, Int32)	Determines the chart element, if any, that is located at a point defined by the given X and Y coordinates.
HitTest(Int32, Int32, Boolean)	Determines the chart element, if any, that is located at a point defined by given X and Y coordinates. Transparent elements can optionally be ignored.
HitTest(Int32, Int32, ChartElementType)	Determines if a chart element of a given type is located at a point defined by given X and Y coordinates.
HitTest(Int32, Int32, Boolean, ChartElementType[])	Determines whether a chart element that is one of the specified types is located at a point defined by the given X and Y coordinates.

Remarks

Call this method to determine the chart element, if any, that is located at a specified point.

This method is often used in some mouse-related event - for example, OnMouseDown - to determine which chart element the end-user clicked. The X and Y mouse coordinates obtained from the event parameters are then used for the `x` and `y` parameter values of this method call.

The properties of the HitTestResult object that is returned can then be used to determine which chart element was clicked; the HitTestResult object also provides an instance of the actual object selected, if any.

HitTest(Int32, Int32)

Determines the chart element, if any, that is located at a point defined by the given X and Y coordinates.

```
C#

public System.Windows.Forms.DataVisualization.Charting.HitTestResult
HitTest (int x, int y);
```

Parameters

x Int32

The X-coordinate value of the point the user clicked.

y Int32

The Y-coordinate value of the point the user clicked.

Returns

HitTestResult

A HitTestResult object, which provides information concerning the chart element, if any, that is at the specified location.

Remarks

Call this method to determine the chart element, if any, which is located at a specified point.

The properties of the HitTestResult object that is returned can then be used to determine which chart element was clicked; the HitTestResult object also provides an instance of the actual object selected, if any.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTest(Int32, Int32, Boolean)

Determines the chart element, if any, that is located at a point defined by given X and Y coordinates. Transparent elements can optionally be ignored.

```
C#

public System.Windows.Forms.DataVisualization.Charting.HitTestResult
HitTest (int x, int y, bool ignoreTransparent);
```

Parameters

`x` Int32

The X-coordinate value of the point the user clicked on.

`y` Int32

The Y-coordinate value of the point the user clicked on.

`ignoreTransparent` Boolean

`true` to ignore transparent elements; otherwise, `false`.

Returns

HitTestResult

A HitTestResult object, which provides information concerning the chart element, if any, that is at the specified location.

Remarks

Call this method to determine the chart element, if any, which is located at a specified point.

To ignore transparent elements, set the `ignoreTransparent` parameter to `true`.

The properties of the HitTestResult object that is returned can then be used to determine which chart element was clicked; the HitTestResult object also provides an instance of the actual object selected, if any.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTest(Int32, Int32, ChartElementType)

Determines if a chart element of a given type is located at a point defined by given X and Y coordinates.

C#

```
public System.Windows.Forms.DataVisualization.Charting.HitTestResult
HitTest (int x, int y,
System.Windows.Forms.DataVisualization.Charting.ChartElementType
requestedElement);
```

Parameters

`x` Int32

The X-coordinate value of the point the user clicked on.

`y` Int32

The Y-coordinate value of the point the user clicked on.

`requestedElement` ChartElementType

A flag that determines the chart element type to be tested.

Returns

HitTestResult

A HitTestResult object, which provides information concerning the chart element, if any, that is at the specified location.

Remarks

Call this method to determine if a chart element of the given type is located at a specified point.

The properties of the HitTestResult object that is returned can then be used to determine which chart element was clicked; the HitTestResult object also provides an instance of the actual object selected, if any.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTest(Int32, Int32, Boolean, ChartElementType[])

Determines whether a chart element that is one of the specified types is located at a point defined by the given X and Y coordinates.

C#

```
public System.Windows.Forms.DataVisualization.Charting.HitTestResult[]
HitTest (int x, int y, bool ignoreTransparent, params
System.Windows.Forms.DataVisualization.Charting.ChartElementType[]
requestedElement);
```

Parameters

`x` Int32

The X-coordinate for the specified data point.

`y` Int32

The Y-coordinate for the specified data point.

`ignoreTransparent` Boolean

`true` to ignore transparent elements; otherwise, `false`.

`requestedElement` ChartElementType[]

An array of ChartElementType objects that specify the types to test for, in order to filter the result. If omitted, checking for element types will be ignored and all element types will be valid.

Returns

HitTestResult[]

An array of HitTestResult objects that provides information about the chart element, if any, found at the specified location. The array contains at least one element, which can be Nothing. The objects in the result are sorted from the top to the bottom of different layers of control.

Remarks

Call this method to determine the gauge element, if any, which is located at a specified point. This method is often used in some mouse-related event, for example, OnMouseDown, to determine which gauge element the end-user clicked. The X and Y mouse coordinates obtained from the event parameters are then used for the X and Y parameter values of this method call.

The properties of the returned HitTestResult object can then be used to determine which chart element was clicked; the HitTestResult object also provides an instance of the actual object selected, if any.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Invalidate Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Invalidates the entire surface of the Chart and causes the Chart control to be redrawn.

Overloads

Invalidate()	Invalidates the entire surface of the Chart and causes the Chart control to be redrawn.
Invalidate(Rectangle)	Invalidates the specified area of the Chart control.

Invalidate()

Invalidates the entire surface of the Chart and causes the Chart control to be redrawn.

C#

```
public void Invalidate ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Invalidate(Rectangle)

Invalidates the specified area of the Chart control.

```
C#

public void Invalidate (System.Drawing.Rectangle rectangle);
```

Parameters

`rectangle` Rectangle

A Rectangle object that specifies the area to invalidate.

Remarks

This method adds the specified area to the Chart control's update area, which is the area that will be repainted at the next paint operation. This method also causes a paint message to be sent to the Chart control.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.LoadTemplate Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Loads a template into the Chart control from a specified source.

Overloads

LoadTemplate(String)	Loads a template with the specified filename from the disk.
LoadTemplate(Stream)	Loads a template into the Chart control from an image stream.

LoadTemplate(String)

Loads a template with the specified filename from the disk.

C#

```
public void LoadTemplate (string name);
```

Parameters

name String

The file name of the template to load. You must specify the full path of the template to be loaded from the disk.

Remarks

The `name` parameter must use a full path, for example, C:\MyPic.jpg. Note that you must also specify the file extension.

This method loads a template from disk that was saved using the Save method.

Make sure to set the IsTemplateMode property to `true` before saving the template with the Save method.

If a loaded template has two data series, the appearance properties of the series are applied sequentially to the chart. In other words, the first template series appearance properties are assigned to the first series of the chart, the second template series appearance properties are applied to the second series of the chart, then the first template series is used again for the third series of the chart, and so on. The same holds true for chart areas, and any other collections that have appearance properties.

Templates can be used to simulate custom palettes.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LoadTemplate(Stream)

Loads a template into the Chart control from an image stream.

```
C#

public void LoadTemplate (System.IO.Stream stream);
```

Parameters

`stream` Stream

The stream to load from.

Remarks

Use this method if you want to load a template from a stream, such as a memory stream, file stream, and so forth.

Make sure to set the IsTemplateMode property to `true` before saving the template with the Save method.

If a loaded template has two data series, the appearance properties of the series are applied sequentially to the chart. In other words, the first template series appearance properties are assigned to the first series of the chart, the second template series

appearance properties are applied to the second series of the chart, then the first template series is used again for the third series of the chart, and so on. The same holds true for chart areas, and any other collections that have appearance properties.

Templates can be used to simulate custom palettes.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnCursorChanged(EventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnCursorChanged(EventArgs) method and raises the CursorPositionChanging and CursorPositionChanged events.

C#

```
protected override void OnCursorChanged (EventArgs e);
```

Parameters

e EventArgs

An EventArgs that contains the event data.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnCustomize Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Raises the Customize event.

C#

```
protected virtual void OnCustomize ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnCustomizeLegend(LegendItemsCollection, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Raises the CustomizeLegend event.

C#

```
protected virtual void OnCustomizeLegend
(System.Windows.Forms.DataVisualization.Charting.LegendItemsCollection
legendItems, string legendName);
```

Parameters

`legendItems` LegendItemsCollection

A LegendItemsCollection object.

`legendName` String

The name of the legend.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnDoubleClick(EventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnDoubleClick(EventArgs) method.

C#

```
protected override void OnDoubleClick (EventArgs e);
```

Parameters

e EventArgs

An EventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnFormatNumber(FormatNumberEventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Raises the FormatNumber event.

C#

```
protected virtual void OnFormatNumber
(System.Windows.Forms.DataVisualization.Charting.FormatNumberEventArgs e);
```

Parameters

e FormatNumberEventArgs

A FormatNumberEventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnGotFocus(EventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Raises the GotFocus event.

C#

```
protected override void OnGotFocus (EventArgs e);
```

Parameters

e EventArgs

An object that contains the event data.

Remarks

Raising the GotFocus event invokes the event handler through a delegate. For more information, see Handling and raising events.

Applies to

Product	Versions
.NET Framework	4.8, 4.8.1

Chart.OnLocationChanged(EventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnLocationChanged(EventArgs) method.

C#

```
protected override void OnLocationChanged (EventArgs e);
```

Parameters

e EventArgs

An EventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnLostFocus(EventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Raises the LostFocus event.

C#

```
protected override void OnLostFocus (EventArgs e);
```

Parameters

e EventArgs

An object that contains the event data.

Remarks

Raising the LostFocus event invokes the event handler through a delegate. For more information, see Handling and raising events.

Applies to

Product	Versions
.NET Framework	4.8, 4.8.1

Chart.OnMouseDown(MouseEventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnMouseDown(MouseEventArgs) method.

C#

```
protected override void OnMouseDown (System.Windows.Forms.MouseEventArgs e);
```

Parameters

e MouseEventArgs

A MouseEventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnMouseMove(MouseEventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnMouseMove(MouseEventArgs) method.

C#

```
protected override void OnMouseMove (System.Windows.Forms.MouseEventArgs e);
```

Parameters

e MouseEventArgs

A MouseEventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnMouseUp(MouseEventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnMouseUp(MouseEventArgs) method.

C#

```
protected override void OnMouseUp (System.Windows.Forms.MouseEventArgs e);
```

Parameters

e MouseEventArgs

A MouseEventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnPaint(PaintEventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnPaint(PaintEventArgs) method.

C#

```
protected override void OnPaint (System.Windows.Forms.PaintEventArgs e);
```

Parameters

e PaintEventArgs

A PaintEventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnPaint Background(PaintEventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnPaintBackground(PaintEventArgs) method.

C#

```
protected override void OnPaintBackground
(System.Windows.Forms.PaintEventArgs pevent);
```

Parameters

pevent PaintEventArgs

A PaintEventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnPostPaint(ChartPaintEventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Raises the PostPaint event.

C#

```
protected virtual void OnPostPaint
(System.Windows.Forms.DataVisualization.Charting.ChartPaintEventArgs e);
```

Parameters

e ChartPaintEventArgs

A ChartPaintEventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnPrePaint(ChartPaintEventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Raises the PrePaint event.

C#

```
protected virtual void OnPrePaint
(System.Windows.Forms.DataVisualization.Charting.ChartPaintEventArgs e);
```

Parameters

e ChartPaintEventArgs

A ChartPaintEventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnResize(EventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnResize(EventArgs) method.

C#

```
protected override void OnResize (EventArgs e);
```

Parameters

e EventArgs

An EventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnRightToLeftChanged(EventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnRightToLeftChanged(EventArgs) method.

C#

```
protected override void OnRightToLeftChanged (EventArgs e);
```

Parameters

e EventArgs

An EventArgs object that contains the event arguments.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.OnSystemColors Changed(EventArgs) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Overrides the OnSystemColorsChanged(EventArgs) method.

C#

```
protected override void OnSystemColorsChanged (EventArgs e);
```

Parameters

e EventArgs

An EventArgs that contains the event data.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Refresh Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Forces the control to invalidate its client area and immediately redraw itself and any child controls.

C#

```
public override void Refresh ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.ResetAutoValues Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Resets automatically calculated chart property values to "Auto".

C#

```
public void ResetAutoValues ();
```

Remarks

Several chart properties can be set to "Auto" - for example, Minimum; these properties are calculated prior to the chart being rendered. "Auto" is usually represented as a NaN value.

Call this method to reset these properties from their calculated value to "Auto" (NaN).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.SaveImage Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Saves the image to the specified target.

Overloads

SaveImage(String, ChartImageFormat)	Saves the chart image to the specified file.
SaveImage(String, ImageFormat)	Saves an image to the specified file.
SaveImage(Stream, ImageFormat)	Saves the image to the specified stream.
SaveImage(Stream, ChartImageFormat)	Saves the chart image to the specified stream.

SaveImage(String, ChartImageFormat)

Saves the chart image to the specified file.

C#

```
public void SaveImage (string imageFileName,
System.Windows.Forms.DataVisualization.Charting.ChartImageFormat format);
```

Parameters

imageFileName String

The name of the file in which image is saved to.

format ChartImageFormat

The chart image format.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SaveImage(String, ImageFormat)

Saves an image to the specified file.

C#

```
public void SaveImage (string imageFileName,
System.Drawing.Imaging.ImageFormat format);
```

Parameters

`imageFileName` String

The name of the file in which image is saved to.

`format` ImageFormat

The image format.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SaveImage(Stream, ImageFormat)

Saves the image to the specified stream.

C#

```
public void SaveImage (System.IO.Stream imageStream,
System.Drawing.Imaging.ImageFormat format);
```

Parameters

`imageStream` Stream

The stream in which the image is saved to.

`format` ImageFormat

The image format.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SaveImage(Stream, ChartImageFormat)

Saves the chart image to the specified stream.

C#

```
public void SaveImage (System.IO.Stream imageStream,
System.Windows.Forms.DataVisualization.Charting.ChartImageFormat format);
```

Parameters

`imageStream` Stream

The stream in which the image is saved to.

`format` ChartImageFormat

The chart image format.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.UpdateAnnotations Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Updates the annotations in the Chart control.

C#

```
public void UpdateAnnotations ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.UpdateCursor Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Updates the cursor in the Chart control.

C#

```
public void UpdateCursor ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AnnotationPlaced Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the end-user places an annotation on the chart.

C#

```
public event EventHandler AnnotationPlaced;
```

Event Type

EventHandler

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AnnotationPositionChanged Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the annotation position is changed.

C#

```
public event EventHandler AnnotationPositionChanged;
```

Event Type

EventHandler

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AnnotationPositionChanging Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the annotation position is about to change.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.AnnotationPosit
ionChangingEventArgs> AnnotationPositionChanging;
```

Event Type

EventHandler<AnnotationPositionChangingEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AnnotationSelectionChanged Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when a selection of the annotation is changed.

C#

```
public event EventHandler AnnotationSelectionChanged;
```

Event Type

EventHandler

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AnnotationTextChanged Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the annotation text is changed.

C#

```
public event EventHandler AnnotationTextChanged;
```

Event Type

EventHandler

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AxisScrollBarClicked Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the axis scroll bar is clicked by the end-user.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.ScrollBarEventA
rgs> AxisScrollBarClicked;
```

Event Type

EventHandler<ScrollBarEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AxisViewChanged Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the axis scale view position or size is changed.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.ViewEventArgs>
AxisViewChanged;
```

Event Type

EventHandler<ViewEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.AxisViewChanging Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the axis scale view position or size is about to change.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.ViewEventArgs>
AxisViewChanging;
```

Event Type

EventHandler<ViewEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.CursorPositionChanged Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the cursor position is changed.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.CursorEventArgs
> CursorPositionChanged;
```

Event Type

EventHandler<CursorEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.CursorPositionChanging Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the cursor position is about to change.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.CursorEventArgs
> CursorPositionChanging;
```

Event Type

EventHandler<CursorEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.Customize Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs just before the chart image is drawn. Use this event to customize the chart picture.

```
C#
public event EventHandler Customize;
```

Event Type

EventHandler

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.CustomizeLegend Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the chart legend must be customized.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.CustomizeLegend
EventArgs> CustomizeLegend;
```

Event Type

EventHandler<CustomizeLegendEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.FormatNumber Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when a numeric value has to be converted to a string.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.FormatNumberEve
ntArgs> FormatNumber;
```

Event Type

EventHandler<FormatNumberEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.GetToolTipText Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs before showing the tooltip to get the tooltip text.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.ToolTipEventArg
s> GetToolTipText;
```

Event Type

EventHandler<ToolTipEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.PostPaint Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the chart element is painted.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.ChartPaintEvent
Args> PostPaint;
```

Event Type

EventHandler<ChartPaintEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.PrePaint Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the chart element background is painted.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.ChartPaintEvent
Args> PrePaint;
```

Event Type

EventHandler<ChartPaintEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.SelectionRangeChanged Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the selection start position or end position is changed.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.CursorEventArgs
> SelectionRangeChanged;
```

Event Type

EventHandler<CursorEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Chart.SelectionRangeChanging Event

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Occurs when the selection start or end position is about to change.

C#

```
public event
EventHandler<System.Windows.Forms.DataVisualization.Charting.CursorEventArgs
> SelectionRangeChanging;
```

Event Type

EventHandler<CursorEventArgs>

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a chart area on the chart image.

C#

```
public class ChartArea :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → ChartArea

Remarks

Within any given chart, the *inner plot position* is the rectangular area, inside the x and y-axis lines, where the data points are drawn. The *plot position* is the rectangular area that encompasses the inner plot position, the tick marks, and the axis labels. The *chart area* is the rectangular area that encompasses the plot position, the tick marks, the axis labels and the axis titles on the chart.

A chart area is represented by a ChartArea class; ChartArea objects are stored in the ChartAreaCollection class.

The number of chart series that can be plotted in a chart area is unlimited.

A ChartArea object:

- Uses one distinct inner plotting area.
- Is used to plot one or more chart series.
- Has primary and secondary x and y-axes, except for pie and doughnut chart types.

When you use Shape chart types, which includes pie, doughnut, funnel, pyramid, polar, and radar charts, the inner plotting position defines the size of the chart type shape. For example, when you increase the value of the InnerPlotPosition property on a pie chart, you will increase the radius of the pie shape.

For all other chart types, each chart area has two sets of X- and Y-axes, primary and secondary. To access properties related to these axes, use the AxisX, AxisX2, AxisY and AxisY2 properties. If you need to set a property that affects all axes on the chart, use the Axes property.

One or more series are associated with a chart area. The series has a ChartArea property that determines the chart area that each series belongs to. By default, the series is connected to the default chart area. To disable the display of a series, set the ChartArea property of that series to a zero-length string.

If another chart area is added to the chart, you can associate any series with the new chart area. In all Shape chart types, as well as bar and stacked bar chart types, each series must be plotted in a separate chart area. In all other chart types, multiple series can be combined in the same chart area.

A 3D chart is displayed on a chart area-by-chart area basis. You can use the Area3DStyle property to enable and customize your chart in 3D.

The chart area minimum and maximum will not be automatically recalculated if you add new data, so the chart will not automatically change axis sizes. You can require the chart area to recalculate these values by using the RecalculateAxesScale method.

To align multiple chart areas vertically, set the AlignmentOrientation property. To align multiple chart areas horizontally, you must first position the chart areas using the Position property, and then set the AlignmentOrientation property to Horizontal.

To suppress the display of a series, set the Series property of that series to a zero-length string.

Constructors

ChartArea()	Initializes a new instance of the ChartArea class.
ChartArea(String)	Initializes a new instance of the ChartArea class with the specified chart area name.

Properties

AlignmentOrientation	Gets or sets the alignment orientation of a chart area.
AlignmentStyle	Gets or sets the alignment style of the ChartArea.
AlignWithChartArea	Gets or sets the name of the ChartArea object to which this chart

	area should be aligned.
Area3DStyle	Gets or sets a ChartArea3DStyle object, which is used to implement three-dimensional (3D) for all series in a chart area.
Axes	Gets or sets an array that represents all axes for a chart area.
AxisX	Gets or sets an Axis object that represents the primary X-axis.
AxisX2	Gets or sets an Axis object that represents the secondary X-axis.
AxisY	Gets or sets an Axis object that represents the primary Y-axis.
AxisY2	Gets or sets an Axis object that represents the secondary Y-axis.
BackColor	Gets or sets the background color of a ChartArea object.
BackGradientStyle	Gets or sets the orientation for the background gradient of a chart area, and also determines whether or not a gradient is used.
BackHatchStyle	Gets or sets the hatching style of a ChartArea object.
BackImage	Gets or sets the background image of a ChartArea object.
BackImageAlignment	Gets or sets the alignment of a ChartArea background image.
BackImageTransparentColor	Gets or sets the color of a ChartArea object background image that will be drawn as transparent.
BackImageWrapMode	Gets or sets the drawing mode for the background image of a ChartArea object.
BackSecondaryColor	Gets or sets the secondary color for the background of a ChartArea object.
BorderColor	Gets or sets the border color of a ChartArea object.
BorderDashStyle	Gets or sets the border style of a ChartArea object.
BorderWidth	Gets or sets the border width of a ChartArea object.
CursorX	Gets or sets a Cursor object that is used for cursors and selected ranges along the X-axis.
CursorY	Gets or sets a Cursor object that is used for cursors and selected ranges along the Y-axis.
InnerPlotPosition	Gets or sets an ElementPosition object, which defines the inner plot position of a chart area object.
IsSameFontSizeForAllAxes	Gets or sets a flag that determines if the labels of the relevant chart area axes are of equal size.

Name	Gets or sets the unique name of a ChartArea object.
Position	Gets or sets an ElementPosition object that defines the position of a ChartArea object within the Chart.
ShadowColor	Gets or sets the shadow color of a ChartArea object.
ShadowOffset	Gets or sets the shadow offset, in pixels, of a ChartArea object.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Visible	Gets or sets a flag that determines if a chart area is visible.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetSeriesDepth(Series)	Returns the depth of a Series in a chart area.
GetSeriesZPosition(Series)	Returns the Z position of a series, which is useful for custom drawing.
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
RecalculateAxesScale()	Recalculates the properties of the chart area after changes have been made to the chart in the Customize event.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)
TransformPoints(Point3D[])	Applies 3D transformations to an array of 3D points, which must be done before GDI+ drawing methods can be called.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ChartArea class.

Overloads

ChartArea()	Initializes a new instance of the ChartArea class.
ChartArea(String)	Initializes a new instance of the ChartArea class with the specified chart area name.

ChartArea()

Initializes a new instance of the ChartArea class.

C#

```
public ChartArea ();
```

Remarks

Use this constructor to create and initialize a new instance of the ChartArea class.

> **Note**
>
> It is highly recommended that you use one of the `Add` or `Insert` methods of the **ChartAreas** collection property to add a chart area at run time.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea(String)

Initializes a new instance of the ChartArea class with the specified chart area name.

C#

```
public ChartArea (string name);
```

Parameters

`name` String

The name for the new ChartArea object.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.AlignmentOrientation Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment orientation of a chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.AreaAlignmentOrientations
AlignmentOrientation { get; set; }
```

Property Value

AreaAlignmentOrientations

An AreaAlignmentOrientations enumeration value that specifies the alignment orientation of a chart area.

Attributes BindableAttribute

Remarks

Through the use of the AlignWithChartArea, AlignmentOrientation and AlignmentStyle properties, it is possible to align or synchronize two or more chart areas horizontally, vertically or both.

First, set the AlignWithChartArea property to the name of a ChartArea object. This chart area will then be aligned, based on the AlignmentStyle setting, which defines the alignment to use, and the AlignmentOrientation setting, which defines the elements of the chart area that should be used to set the alignment.

If the AlignWithChartArea property is not set, the chart area will not be aligned, regardless of the AlignmentStyle and AlignmentOrientation settings.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.AlignmentStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment style of the ChartArea.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.AreaAlignmentStyles
AlignmentStyle { get; set; }
```

Property Value

AreaAlignmentStyles

An AreaAlignmentStyles enumeration value that can represent either a single value or the bitwise-OR of multiple values that define the alignment to use.

Attributes BindableAttribute

Remarks

Use the AlignmentStyle property to align two or more chart areas horizontally, vertically or both. The alignment type is one of the following values, or a bitwise OR combination of the values below.

Member Name	Description
All	Chart areas are aligned using all values.
AxesView	Chart areas are aligned using the View of the specified ChartArea.
Cursor	Chart areas are aligned on the cursor.
None	Chart areas are not automatically aligned.
Position	Chart areas are aligned with the Position of the chart area specified.

Member Name	Description
InnerPlotPosition	Chart areas are aligned with the InnerPlotPosition of the chart area specified.

If the AlignWithChartArea value is not set, the AlignmentStyle property will have no effect.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.AlignWithChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the ChartArea object to which this chart area should be aligned.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.LegendAreaNameConverter))]
public string AlignWithChartArea { get; set; }
```

Property Value

String

A `string` value that represents the name of a ChartArea object to which this chart area should be aligned. The default is the empty string.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Through the use of the AlignWithChartArea, AlignmentOrientation and AlignmentStyle properties, it is possible to align or synchronize two or more chart areas horizontally, vertically or both.

First, set the AlignWithChartArea property to the name of a ChartArea object. This chart area will then be aligned, based on the AlignmentStyle setting, which defines the alignment to use, and the AlignmentOrientation setting, which defines the elements of the chart area that should be used to set the alignment.

If the AlignWithChartArea property is not set, the chart area will not be aligned, regardless of the AlignmentStyle and AlignmentOrientation settings.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.Area3DStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a ChartArea3DStyle object, which is used to implement three-dimensional (3D) for all series in a chart area.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.ChartArea3DStyle Area3DStyle { get; set; }
```

Property Value

ChartArea3DStyle

A ChartArea3DStyle object that implements 3D for all series plotted in a chart area.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

To enable or disable 3D for a chart area, set the Enable3D property.

When the chart is drawn in 3D, scale breaks are not supported on the axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.Axes Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an array that represents all axes for a chart area.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxesArrayConverter))]
public System.Windows.Forms.DataVisualization.Charting.Axis[] Axes { get; set; }
```

Property Value

Axis[]

An array of Axis objects that represents all axes used by a chart area.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The main purpose of this property is to provide access to the properties of the X, Y, X2 and Y2 axes at design time.

At run time, the axes for a chart area should be accessed using the individual axis properties, such as AxisX, AxisX2, and so forth.

If you decide to use this property at run time - for example, in an event - you can access an item in this array by specifying an AxisName enumeration member as the identifier.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.AxisX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an Axis object that represents the primary X-axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.Axis AxisX { get; set; }
```

Property Value

Axis

An Axis object that represents the primary X-axis.

Attributes BindableAttribute, BrowsableAttribute, TypeConverterAttribute

Remarks

Use this property to manipulate the primary X-axis in the chart area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.AxisX2 Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an Axis object that represents the secondary X-axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.Axis AxisX2 { get; set; }
```

Property Value

Axis

An Axis object that represents the secondary X-axis.

Attributes BindableAttribute, BrowsableAttribute, TypeConverterAttribute

Remarks

Use this property to manipulate the secondary X-axis in the chart area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.AxisY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an Axis object that represents the primary Y-axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.Axis AxisY { get; set; }
```

Property Value

Axis

An Axis object that represents the primary Y-axis.

Attributes BindableAttribute, BrowsableAttribute, TypeConverterAttribute

Remarks

Use this property to manipulate the primary Y-axis in the chart area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.AxisY2 Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an Axis object that represents the secondary Y-axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.Axis AxisY2 { get; set; }
```

Property Value

Axis

An Axis object that represents the secondary Y-axis.

Attributes BindableAttribute, BrowsableAttribute, TypeConverterAttribute

Remarks

Use this property to manipulate the secondary Y-axis in the chart area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value that specifies the background color of a chart area. The default color is White.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set this property to any valid ARGB (alpha, red, green, blue) value.

The BackColor value will be the first color used if you have specified gradient colors for the background.

The alpha value controls the opacity of a color. If you set this color to "transparent" - that is, if you use an alpha value of 0 - no color will be assigned to the background of the chart area. As a result, the background will be transparent.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the orientation for the background gradient of a chart area, and also determines whether or not a gradient is used.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration that specifies the orientation for the background gradient of a chart area and determines whether a gradient is used. The default value is None.

Attributes BindableAttribute

Remarks

Use the BackGradientStyle property to determine whether a chart area will display a gradient, and to determine the orientation of the gradient if one is used.

For a listing of available patterns, refer to the GradientStyle enumeration.

To use gradient colors:

- Set the primary color of the chart area background with the BackColor property.
- Set the secondary color of the chart area background with the BackSecondaryColor property.
- Set this BackGradientStyle property to the desired gradient style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatching style of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration that specifies the hatching style of a chart area. The default value is None.

Attributes BindableAttribute

Remarks

The ChartHatchStyle enumeration specifies the hatch pattern used by a brush of type HatchBrush. The hatch pattern consists of a solid background color and lines drawn over the background.

For a listing of available hatching styles, refer to the ChartHatchStyle enumeration.

> **Note**
>
> The color of hatching lines is determined by the **BackSecondaryColor** property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BackImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
public string BackImage { get; set; }
```

Property Value

String

A `string` value that represents the URL of an image file, used as the background image of a chart area. The default is an empty string.

Attributes BindableAttribute

Remarks

This property defines a path to a background image for a ChartArea, and can be an absolute or relative URL.

To position the image, use the BackImageAlignment and BackImageWrapMode properties.

You can also specify one of the colors displayed by the image as "transparent", by setting the BackImageTransparentColor property.

An image be any valid graphics file supported by .NET Framework, including icons.

Background images take precedence over other background properties such as gradient color.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BackImageAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of a ChartArea background image.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.ChartImageAlignmentStyle
BackImageAlignment { get; set; }
```

Property Value

ChartImageAlignmentStyle

A ChartImageAlignmentStyle enumeration value that specifies the alignment of the background image of the chart area. The default value is TopLeft.

Attributes BindableAttribute

Remarks

This property determines the alignment of the background image for the chart area, and is only applied when the ChartImageWrapMode property is set to Unscaled.

For more information about the available alignments, see the ChartImageAlignmentStyle enumeration topic.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BackImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of a ChartArea object background image that will be drawn as transparent.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that represents the color of a background image that will be drawn as transparent. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BackImageWrapMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the drawing mode for the background image of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartImageWrapMode
BackImageWrapMode { get; set; }
```

Property Value

ChartImageWrapMode

A ChartImageWrapMode enumeration value that specifies the drawing mode for the background image of a chart area. The default value is Tile.

Attributes BindableAttribute

Remarks

The BackImageWrapMode property determines how the image is displayed; the image can be tiled, scaled, and so forth.

If an image is scaled, it will be stretched to fit within the bounds of the chart area. If the image is tiled, it is repeatedly drawn, starting from the top-left corner of the chart area.

For more information about the available wrapping modes, see the ChartImageWrapMode enumeration topic.

If the BackImageWrapMode property is set to Unscaled, the location where the image is drawn is determined by the BackImageAlignment property setting.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color for the background of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value that specifies the secondary color for the background of a chart area. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This BackSecondaryColor property is one of several related properties that determine the secondary background color of Chart objects, and is used when gradient colors are being applied.

You can set BackSecondaryColor to any valid ARGB (alpha, red, green, blue) value.

If hatching is used, the BackSecondaryColor property determines the color of the hatching lines for the chart area.

To use gradient colors for the background of a chart area:

- Set the primary color with the BackColor property.
- Set the secondary color with the BackSecondaryColor property.

- Set the BackGradientStyle property to the desired gradient style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderColor { get; set; }
```

Property Value

Color

A Color value that specifies the border color of a chart area. The default color is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set the BorderColor property to any valid ARGB (alpha, red, green, blue) value.

> **Note**
>
> For pie and doughnut charts, this property determines the color of radial and horizontal lines if data labels are drawn outside the pie or doughnut - that is, if the `LabelStyle` custom property is set to "Outside".

This property also determines the marker lines for 3D charts if the `ShowMarkerLines` custom property is set to `true`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border style of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that determines the border style of the chart area.

Attributes BindableAttribute

Remarks

The BorderDashStyle property determines the line style to use for a border. The line can be solid, dotted, and so forth.

For other border styles, see the ChartDashStyle enumeration topic.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.BorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
public int BorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that determines the border width, in pixels, of the chart area.

Attributes BindableAttribute

Remarks

Note that the BorderWidth property can also set the width of lines for line charts, custom legend items displayed with a line style, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.CursorX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a Cursor object that is used for cursors and selected ranges along the X-axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.Cursor CursorX { get; set; }
```

Property Value

Cursor

A Cursor object used for cursors and selected ranges along the X-axis.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This Cursor object is used for the X-axis cursor and its selected ranges. For all charts except bar chart types, the X-axis cursor represents the vertical cursor. For bar chart types, the X-axis cursor represents the horizontal cursor.

Refer to the Cursor class documentation for a description of its members, as well as sample source code.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.CursorY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a Cursor object that is used for cursors and selected ranges along the Y-axis.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public System.Windows.Forms.DataVisualization.Charting.Cursor CursorY { get; set; }
```

Property Value

Cursor

A Cursor object used for cursors and selected ranges along the Y-axis.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This Cursor object is used for the Y-axis cursor and its selected ranges. For all charts except bar chart types, the Y-axis cursor represents the horizontal cursor. For bar chart types, the Y-axis cursor represents the vertical cursor.

Refer to the Cursor class documentation for a description of its members, as well as sample source code.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.InnerPlotPosition Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an ElementPosition object, which defines the inner plot position of a chart area object.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.ElementPositionConverter))]
public System.Windows.Forms.DataVisualization.Charting.ElementPosition
InnerPlotPosition { get; set; }
```

Property Value

ElementPosition

An ElementPosition object, which defines the inner plot position of a ChartArea object.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The InnerPlotPosition property defines the rectangle within a chart area element that is used for plotting data; it excludes tick marks, axis labels, and so forth.

The coordinates used for this property (0,0 to 100,100) are related to the ChartArea object, and not to the entire Chart.

The InnerPlotPosition property can be used to align multiple chart areas. However, if one chart area has tick marks and axis labels and another one does not, their axis lines cannot be aligned.

Important

This property can also be used to change the size of charts, such as pie or doughnut, without changing the size of the generated chart image.

To change the position of an entire ChartArea object use the Position property instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.IsSameFontSizeForAllAxes Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines if the labels of the relevant chart area axes are of equal size.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsSameFontSizeForAllAxes { get; set; }
```

Property Value

Boolean

`true` if all axes that have their IsLabelAutoFit property set to `true` and will display their labels using the same font size; otherwise, `false`.

Attributes BindableAttribute

Remarks

The font size is automatically calculated by the Chart control, with the largest size that fits being used.

This property applies to all chart area axes that have their IsLabelAutoFit property set to `true`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unique name of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the unique name of a `ChartArea` object.

Attributes BindableAttribute

Remarks

The name of a ChartArea object must be unique, otherwise an exception will be thrown.

This property can be used as the indexer when retrieving a ChartArea object from the ChartAreaCollection object.

If you set this property to a zero-length string. an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.Position Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an ElementPosition object that defines the position of a ChartArea object within the Chart.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.ElementPositionConverter))]
public System.Windows.Forms.DataVisualization.Charting.ElementPosition
Position { get; set; }
```

Property Value

ElementPosition

An ElementPosition object that defines the position of a chart area object within the chart image.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property defines the position of a ChartArea object within the Chart, and includes tick marks, axis labels, and so forth.

The coordinates used for this property (0,0 to 100,100) are related to the entire chart image.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.ShadowColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow color of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ShadowColor { get; set; }
```

Property Value

Color

A Color value that specifies the shadow color of a chart area. The default color is Black.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property can be set to any valid ARGB (alpha, red, green, blue) value.

The alpha value can be used to achieve a "realistic" shadowing effect; try a color value of "128,0,0,0" for shadows.

To implement a shadow, set the ShadowOffset property to a value other than zero (0).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.ShadowOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow offset, in pixels, of a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
public int ShadowOffset { get; set; }
```

Property Value

Int32

An `integer` that represents the shadow offset, in pixels, of the chart area.

Attributes BindableAttribute

Remarks

To implement shadowing, use a value other than zero (0). A positive value will result in a shadow that shifts down and to the right. A negative value will result in a shadow that shifts up and to the left.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.Visible Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines if a chart area is visible.

```
C#
public virtual bool Visible { get; set; }
```

Property Value

Boolean

`true` if the chart area is visible; otherwise, `false`.

Remarks

When this flag is set to `false`, all series, legends, titles and annotation objects associated with the chart area will also be hidden.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.GetSeriesDepth(Series) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the depth of a Series in a chart area.

C#

```
public float GetSeriesDepth
(System.Windows.Forms.DataVisualization.Charting.Series series);
```

Parameters

series Series

The Series for which the depth is retrieved.

Returns

Single

A `float` value that represents the depth of the specified Series, which is measured as a percentage of the chart area's depth.

Remarks

Call this method to get the depth of a specified series. The returned value is measured as a percentage of the chart area's depth, which is the same unit of measurement used in the Z property.

All series have a depth, regardless of their chart type. For example, even point and bubble series have a depth, even though the entire depth is not used to draw the points or bubbles.

The depth of a series can be used to determine a Z value for a point that is used in a custom drawing routine.

> **Note**
>
> Any points used in custom 3D drawing must be transformed from relative 3D coordinates to relative 2D coordinates using the **TransformPoints** method.

To get relative X and Y-values from an axis, use the ValueToPosition method for both the X and Y axes.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.GetSeriesZPosition(Series) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the Z position of a series, which is useful for custom drawing.

C#

```
public float GetSeriesZPosition
(System.Windows.Forms.DataVisualization.Charting.Series series);
```

Parameters

`series` Series

The Series for which the Z position is retrieved.

Returns

Single

A `float` value that represents the Z position of the specified series, which is measured as a percentage of the depth of a chart area.

Remarks

Call this method to get the Z position of a series. The Z position is measured from the back wall of the chart area to the center of a data series, and is represented as a percentage of the chart area's depth. For example, a series that is drawn halfway along the Z-axis has a Z position of 50.

The Z position of a series is normally used for a custom drawing routine.

Note

Any points used in custom 3D drawing must be transformed from relative 3D coordinates to relative 2D coordinates using the **TransformPoints** method.

To get relative X and Y-values from an axis, use the ValueToPosition method for both the X and Y axes.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.RecalculateAxesScale Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Recalculates the properties of the chart area after changes have been made to the chart in the Customize event.

C#

```
public void RecalculateAxesScale ();
```

Remarks

If you use the Customize event to make changes, it is sometimes necessary to recalculate chart area properties so that the chart is rendered correctly. For example, if an axis range is changed, the labels for that axis must be recalculated.

Call this method after you use the Customize event to make ANY changes to a chart area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea.TransformPoints(Point3D[]) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Applies 3D transformations to an array of 3D points, which must be done before GDI+ drawing methods can be called.

C#

```
public void TransformPoints
(System.Windows.Forms.DataVisualization.Charting.Point3D[] points);
```

Parameters

points Point3D[]

An array of 3D points.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Manages the display attributes for three-dimensional (3D) chart areas.

C#

```
public class ChartArea3DStyle
```

Inheritance Object → ChartArea3DStyle

Remarks

This class provides the styles for 3D chart areas, and is exposed using the Area3DStyle property.

To enable 3D for a chart area, set the Enable3D property to `true`. This causes 3D to be applied to all data series drawn in a chart area.

An isometric projection of a chart area can be displayed by setting the IsRightAngleAxes property to `true`, which is the default. Isometric views are not actually 3D, because the displayed angles of rotation may not match the actual angles of rotation around the vertical and horizontal axes. The actual angles are controlled using the Rotation and Inclination properties, respectively.

For more information about isometric projections, see the Remarks section of the IsRightAngleAxes property.

You can also use this class to:

- Cluster bar and column chart data series using the IsClustered property.
- Control the lighting effects using the LightStyle property.
- Set the width of chart area walls using the WallWidth property.
- Set the amount of perspective with the Perspective property.

- Control the depth of data points and the distance between data rows using the PointDepth and PointGapDepth properties, respectively.
- Rotate a chart area using the Inclination and Rotation properties.

To work with coordinates of 3D points, use the Point3D class.

When a chart is set to 3D, the following chart features are not valid:

- Scale breaks
- Gradients
- Scrolling and zooming

Constructors

ChartArea3DStyle()	Initializes a new instance of the ChartArea3DStyle class.
ChartArea3DStyle(ChartArea)	Initializes a new instance of the ChartArea3DStyle class using the `chartArea` parameter.

Properties

Enable3D	Gets or sets a flag that toggles the 3D on and off for a chart area.
Inclination	Gets or sets the angle of rotation around the horizontal axes for 3D chart areas.
IsClustered	Gets or sets a flag that determines whether data series for a bar or column chart are clustered; that is, displayed along distinct rows.
IsRightAngleAxes	Gets or sets a flag that determines whether a chart area is displayed using an isometric projection.
LightStyle	Gets or sets the style of lighting for a 3D chart area.
Perspective	Gets or sets the percent of perspective for a 3D chart area.
PointDepth	Gets or sets the depth of data points displayed in a 3D chart area.
PointGapDepth	Gets or sets the distance between series rows in a 3D chart area.
Rotation	Gets or sets the angle of rotation around the vertical axes for 3D chart areas.

WallWidth	Gets or sets the width of the walls displayed in a 3D chart area.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ChartArea3DStyle class.

Overloads

ChartArea3DStyle()	Initializes a new instance of the ChartArea3DStyle class.
ChartArea3DStyle(ChartArea)	Initializes a new instance of the ChartArea3DStyle class using the `chartArea` parameter.

ChartArea3DStyle()

Initializes a new instance of the ChartArea3DStyle class.

C#

```
public ChartArea3DStyle ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle(ChartArea)

Initializes a new instance of the ChartArea3DStyle class using the `chartArea` parameter.

C#

```
public ChartArea3DStyle
(System.Windows.Forms.DataVisualization.Charting.ChartArea chartArea);
```

Parameters

chartArea ChartArea

A ChartArea object that is used for initialization.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.Enable3D Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that toggles the 3D on and off for a chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public bool Enable3D { get; set; }
```

Property Value

Boolean

`true` the chart area is displayed using 3D, `false` if the chart area is displayed in 2D. The default value is `false`.

Attributes BindableAttribute

Remarks

Toggles the three-dimensional display of a chart area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.Inclination Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the angle of rotation around the horizontal axes for 3D chart areas.

C#

```
[System.ComponentModel.Bindable(true)]
public int Inclination { get; set; }
```

Property Value

Int32

An `integer` that represents the angle of rotation around the horizontal axes for 3D chart areas.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.IsClustered Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether data series for a bar or column chart are clustered; that is, displayed along distinct rows.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsClustered { get; set; }
```

Property Value

Boolean

`true` if whether data series for a bar or column chart are clustered; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.IsRightAngleAxes Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether a chart area is displayed using an isometric projection.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsRightAngleAxes { get; set; }
```

Property Value

Boolean

`true` if the chart area is displayed using an isometric projection; otherwise, `false`.

Attributes BindableAttribute

Remarks

Isometric views are not actually 3D, because the displayed angles of rotation may not match the actual angles of rotation around the vertical and horizontal axes. The actual angles are controlled using the Rotation and Inclination properties, respectively.

Note the following points concerning isometric projection:

- It is not "real" 3D. That is, the angle of rotation around axes doesn't use true rotational angles.
- It is ideal for displaying 3D charts with some depth.
- If your application allows end-users to rotate the charts, the rotation will not be smooth. In this case, using isometric views is not recommended.

- With isometric projection, perspective is not supported.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.LightStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of lighting for a 3D chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.LightStyle LightStyle
{ get; set; }
```

Property Value

LightStyle

A LightStyle enumeration value that specifies the style of lighting for a 3D chart area.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.Perspective Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the percent of perspective for a 3D chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public int Perspective { get; set; }
```

Property Value

Int32

An `integer` value that represents the percent of perspective for a 3D chart area. The allowable range is 0-100 percent, and the default is zero (0) percent.

Attributes BindableAttribute

Remarks

A perspective projection of any set of parallel lines that are not parallel to the projection plane converges to a vanishing point.

If you set the Perspective property, the IsRightAngleAxes property will be automatically set to `false`, because these properties are mutually exclusive.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.PointDepth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the depth of data points displayed in a 3D chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public int PointDepth { get; set; }
```

Property Value

Int32

An `integer` value that represents the depth of data points. The allowable range is 0-1000 percent. The default is 100 percent.

Attributes BindableAttribute

Remarks

This property determines the depth of data points, and is represented as a percentage of the data point width.

The PointDepth property can be applied to the depth of bar, column, line, pie and spline chart data points only. If you set this property for other chart types (for example, bubble and point charts) that cannot have their point depths "stretched", it will reserve extra space for the data points, but will not increase their depth.

You can display the chart as an isometric projection by setting the IsRightAngleAxes property to `true`, which is the default value. This will cause the point depth to appear to be more or less than the percentage set by this property, depending on the chart being displayed.

To control the spacing between rows of data, use the PointGapDepth property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.PointGapDepth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the distance between series rows in a 3D chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public int PointGapDepth { get; set; }
```

Property Value

Int32

An `integer` value that represents the distance between data rows (that is, the data series) in the 3D chart area. The allowable range is 0-1000 percent, and the default is 100 percent.

Attributes BindableAttribute

Remarks

This property determines the distance between data rows, with each row corresponding to a data series. The unit of measurement is expressed as a percentage of the distance between data points in one row.

Changing the gap between columns or bars for column and bar chart types using the `PointWidth` custom property will affect the PointGapDepth property value.

Set the IsRightAngleAxes property to `true` (the default value) to display the chart as an isometric projection. This will cause the point gap depth to appear to be more or less than the percentage set by this property, depending on the chart being displayed.

To control the depth of data points, use the PointDepth property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.Rotation Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the angle of rotation around the vertical axes for 3D chart areas.

C#

```
[System.ComponentModel.Bindable(true)]
public int Rotation { get; set; }
```

Property Value

Int32

An `integer` value that represents the angle of rotation around the vertical axes for 3D chart areas.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartArea3DStyle.WallWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of the walls displayed in a 3D chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public int WallWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the width, in pixels, of the walls displayed in a 3D chart area.

Attributes BindableAttribute

Remarks

This property determines the width of the walls (for example, the back wall, side walls, and so forth) that are displayed in 3D chart areas. The allowable range is 0-30 pixels.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartAreaCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of ChartArea objects.

C#

```
public class ChartAreaCollection :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<
System.Windows.Forms.DataVisualization.Charting.ChartArea>
```

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<ChartArea> → ChartAreaCollection

Remarks

The ChartAreaCollection class represents a collection of ChartArea objects, which in turn represent the chart areas that are displayed within the chart image.

This class is exposed as the ChartAreas property of the root Chart object.

The Count property indicates the total number of items in the collection and is commonly used to find the upper bound of the collection.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Item[String]	Gets or sets the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>.

	(Inherited from Collection<T>)
NamePrefix	Gets the name prefix that is used to create unique chart element names. (Inherited from ChartNamedElementCollection<T>)

Methods

Add(String)	Adds a ChartArea object with the specified name to the collection.
Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds the chart element by the specified name. (Inherited from ChartNamedElementCollection<T>)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance.

	(Inherited from Object)
IndexOf(String)	Finds the index of the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index. (Inherited from ChartNamedElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection. (Inherited from ChartNamedElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection. (Inherited from ChartNamedElementCollection<T>)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection. (Inherited from ChartNamedElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element. (Inherited from ChartNamedElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object.

	(Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.

Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified

	IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>,	Returns the first element of the sequence that satisfies a condition or a default value if no

Func<TSource,Boolean>)	such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.

GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Double>)	of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32

	value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>,	Adds a value to the beginning of the

TSource)	sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an

	exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.

Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>,	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key

Func<TSource,TElement>, IEqualityComparer<TKey>)	selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey>(IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey>(IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement>(IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement>(IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEqualityComparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult>(IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.

AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartAreaCollection.Add(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a ChartArea object with the specified name to the collection.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartArea Add (string name);
```

Parameters

name String

The name of the object to be added to the collection.

Returns

ChartArea

A `string` value that represents a reference to the newly added object.

Remarks

Objects are always added as the last item in the collection.

The `name` argument specifies the Name property of the object being added.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartColorPalette Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a palette when setting a Palette property.

C#

```
public enum ChartColorPalette
```

Inheritance Object → ValueType → Enum → ChartColorPalette

Fields

Berry	8	Palette utilizes blues and purples.
Bright	1	Palette utilizes bright colors.
BrightPastel	12	Palette utilizes bright pastel colors.
Chocolate	9	Palette utilizes shades of brown.
EarthTones	6	Palette utilizes earth tone colors such as green and brown.
Excel	3	Palette utilizes Excel-style colors.
Fire	10	Palette utilizes red, orange and yellow colors.
Grayscale	2	Palette utilizes grayscale colors, that is, shades of black and white.
Light	4	Palette utilizes LightStyle style colors; very light colors.
None	0	No palette is used.
Pastel	5	Palette utilizes pastel colors.
SeaGreen	11	Palette utilizes colors that range from green to blue.
SemiTransparent	7	Palette utilizes semi-transparent colors.

Remarks

The ChartColorPalette enumeration represents the palettes to be used for the root Chart object and any Series objects.

If no colors are assigned to data points and their associated series, the default colors will be assigned to the data. For example:

- If Palette is set to None, then the default palette is set to PaletteCustomColors or BrightPastel.

If.Palette is set to None, then the default palette for the chart is used (Palette).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartDashStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a line style.

C#

```
public enum ChartDashStyle
```

Inheritance Object → ValueType → Enum → ChartDashStyle

Fields

Dash	1	A dashed line.
DashDot	2	A line with a repeating dash-dot pattern.
DashDotDot	3	A line a repeating dash-dot-dot pattern.
Dot	4	A line with a repeating dot pattern.
NotSet	0	The line style is not set.
Solid	5	A solid line.

Remarks

The ChartDashStyle enumeration represents a line style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElement Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Base class for all the chart elements. The ChartElement class represents the most basic element of the chart element hierarchy.

C#

```
public abstract class ChartElement : IDisposable
```

Inheritance Object → ChartElement

Derived System.Windows.Forms.DataVisualization.Charting.AnnotationPathPoint
System.Windows.Forms.DataVisualization.Charting.BorderSkin
System.Windows.Forms.DataVisualization.Charting.ChartGraphics
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
System.Windows.Forms.DataVisualization.Charting.ElementPosition
More...

Implements IDisposable

Constructors

ChartElement()	Initializes a new instance of the ChartElement class.

Properties

Tag	Gets or sets an object associated with this chart element.

Methods

Dispose()	Releases the resources used by the ChartElement.

Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement.
GetHashCode()	Returns a hash function for a particular type.
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElement Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ChartElement class.

C#

```
protected ChartElement ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElement.Tag Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an object associated with this chart element.

C#

```
[System.ComponentModel.Browsable(false)]
public object Tag { get; set; }
```

Property Value

Object

An Object associated with this chart element.

Attributes BrowsableAttribute

Remarks

The Tag property can be used to store additional data with this chart element.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElement.Dispose Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the resources used by the ChartElement.

Overloads

Dispose()	Releases the resources used by the ChartElement.
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources.

Dispose()

Releases the resources used by the ChartElement.

C#

```
public void Dispose ();
```

Implements

Dispose()

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Dispose(Boolean)

Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources.

```
C#

protected virtual void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElement.Equals(Object) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Determines whether the specified Object is equal to the current ChartElement.

C#

```
public override bool Equals (object obj);
```

Parameters

obj Object

The Object to compare with the current ChartElement.

Returns

Boolean

`true` if the specified Object is equal to the current ChartElement; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElement.GetHashCode Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns a hash function for a particular type.

C#

```
public override int GetHashCode ();
```

Returns

Int32

An `integer` value that represents the hash code for the current Object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElement.ToString Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns a string that represents the current Object.

C#

```
public override string ToString ();
```

Returns

String

A `string` value that represents the current Object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T> Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the base class for all chart element collections.

C#

```
public abstract class ChartElementCollection<T> :
System.Collections.ObjectModel.Collection<T>, IDisposable where T :
ChartElement
```

Type Parameters

T

Inheritance Object → Collection<T> → ChartElementCollection<T>

Derived System.Windows.Forms.DataVisualization.Charting.AnnotationPathPointCollection
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<T>
System.Windows.Forms.DataVisualization.Charting.CustomLabelsCollection
System.Windows.Forms.DataVisualization.Charting.DataPointCollection
System.Windows.Forms.DataVisualization.Charting.LegendItemsCollection
More...

Implements IDisposable

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index.

	(Inherited from Collection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)

Methods

Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>.
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>.
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts an element into the Collection<T> at the specified index

	location.
Invalidate()	Forces the invalidation of the parent chart element.
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the element at the specified index location of the Collection<T>.
ResumeUpdates()	Resumes the invalidation of a chart element in the collection.
SetItem(Int32, T)	Replaces the element at the specified index location.
SuspendUpdates()	Suspends the invalidation of a chart element in the collection.
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)

IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.

All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by

	invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>,	Returns the element at a specified index in a

Int32)	sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult>	Groups the elements of a sequence according

(IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.

Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.

Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Single>)	of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a

	result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns

	the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a

	transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>,	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key

Func<TSource,TElement>)	selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.

Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T>.ClearItems Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Removes all elements from the Collection<T>.

C#

```
protected override void ClearItems ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T>.Dispose Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the resources used by the ChartElementCollection<T>.

Overloads

Dispose()	Releases the resources used by the ChartElementCollection<T>.
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources.

Dispose()

Releases the resources used by the ChartElementCollection<T>.

C#

```
public void Dispose ();
```

Implements

Dispose()

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Dispose(Boolean)

Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources.

```
C#

protected virtual void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T>.InsertItem(Int32, T) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Inserts an element into the Collection<T> at the specified index location.

C#

```
protected override void InsertItem (int index, T item);
```

Parameters

index Int32

The zero-based index location of the item will be inserted.

item T

The object to insert. The value can be `null` for reference types.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T>.Invalidate Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Forces the invalidation of the parent chart element.

C#

```
public virtual void Invalidate ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T>.RemoveItem(Int32) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Removes the element at the specified index location of the Collection<T>.

C#

```
protected override void RemoveItem (int index);
```

Parameters

`index` Int32

The zero-based index location of the element to remove.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T>.ResumeUpdates Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Resumes the invalidation of a chart element in the collection.

C#

```
public virtual void ResumeUpdates ();
```

Remarks

If you call the SuspendUpdates method several times, you will need to call the ResumeUpdates method an equal number of times.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T>.SetItem(Int32, T) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Replaces the element at the specified index location.

C#

```
protected override void SetItem (int index, T item);
```

Parameters

index Int32

The zero-based index location of the element to replace.

item T

The new value for the element at the specified index location. The value can be `null` for reference types.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementCollection<T>.SuspendUpdates Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Suspends the invalidation of a chart element in the collection.

C#

```
public virtual void SuspendUpdates ();
```

Remarks

A call to the Invalidate method will have no effect after the SuspendUpdates method is called.

If you call the SuspendUpdates method several times, you will need to call the ResumeUpdates method an equal number of times.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementOutline Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents an array of marker points and the outline path used for visual object selection in the chart.

C#

```
public class ChartElementOutline : IDisposable
```

Inheritance Object → ChartElementOutline

Implements IDisposable

Remarks

The value of the OutlinePath property can be `null` for complex objects or objects with two points or fewer.

Properties

Markers	Gets or sets the array of marker points.
OutlinePath	Gets or sets the outline path. The result can be `null` for complex objects and objects with two points or fewer.

Methods

Dispose()	Performs application-defined tasks that frees, releases, or resets unmanaged resources.
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
Equals(Object)	Determines whether the specified object is equal to the current

	object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementOutline.Markers Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the array of marker points.

C#

```
public
System.Collections.ObjectModel.ReadOnlyCollection<System.Drawing.PointF>
Markers { get; }
```

Property Value

ReadOnlyCollection<PointF>

An array of PointF values that represents the marker points.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementOutline.OutlinePath Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the outline path. The result can be `null` for complex objects and objects with two points or fewer.

C#

```
public System.Drawing.Drawing2D.GraphicsPath OutlinePath { get; }
```

Property Value

GraphicsPath

A GraphicsPath object that represents the outline path.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementOutline.Dispose Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs application-defined tasks that frees, releases, or resets unmanaged resources.

Overloads

Dispose()	Performs application-defined tasks that frees, releases, or resets unmanaged resources.
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.

Dispose()

Performs application-defined tasks that frees, releases, or resets unmanaged resources.

C#

```
public void Dispose ();
```

Implements

Dispose()

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Dispose(Boolean)

Releases unmanaged and, optionally, managed resources.

```
C#
protected virtual void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartElementType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a chart element type.

C#

```
public enum ChartElementType
```

Inheritance Object → ValueType → Enum → ChartElementType

Fields

Annotation	22	Any object derived from the Annotation class.
Axis	3	An Axis object.
AxisLabelImage	7	An Axis label image.
AxisLabels	8	An Axis label.
AxisTitle	9	An Axis title.
DataPoint	16	A DataPoint object.
DataPointLabel	17	A series DataPoint label.
Gridlines	5	Any major or minor grid line, either vertical or horizontal.
LegendArea	18	The area inside a Legend object. Does not include the space occupied by legend items.
LegendHeader	20	A Legend header.
LegendItem	21	A LegendItem object.
LegendTitle	19	A Legend title.
Nothing	0	No chart element.

PlottingArea	2	The plotting area, which is the chart area excluding axes, labels, and so on. Also excludes the areas that data points might occupy.
ScrollBarLargeDecrement	13	The background of a scrollbar that, when clicked, results in a large decrement in the view size. This background is located below the thumb for a vertical scrollbar, and to the left of the thumb for a horizontal scrollbar.
ScrollBarLargeIncrement	14	The background of a scrollbar that, when clicked, results in a large increment in the view size. This background is located above the thumb for a vertical scrollbar, and to the right of the thumb for a horizontal scrollbar.
ScrollBarSmallDecrement	11	A scrollbar small decrement button. A "down arrow" button for a vertical scrollbar, a "left arrow" button for a horizontal scrollbar.
ScrollBarSmallIncrement	12	A scrollbar small increment button. An "up arrow" button for a vertical scrollbar, a "right arrow" button for a horizontal scrollbar.
ScrollBarThumbTracker	10	A scrollbar tracking thumb.
ScrollBarZoomReset	15	The ZoomReset button of a scrollbar.
StripLines	6	A StripLine object.
TickMarks	4	Any major or minor TickMark object.
Title	1	The title of a Chart.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides numerous conversion functions and exposes a GDI+ Graphics object.

C#

```
public class ChartGraphics :
System.Windows.Forms.DataVisualization.Charting.ChartElement
```

Inheritance Object → ChartElement → ChartGraphics

Remarks

This class provides numerous conversion functions and also exposes a GDI+ Graphics object. Note that when rendering as Scalable Vector Graphics (SVG), use of the Graphics object for custom drawing will not be included as part of the SVG document.

Note also that, by default, Graphics functions usually take absolute coordinates. Therefore the conversion methods of this class should be useful if you perform drawing operations.

A ChartGraphics object is exposed in the ChartGraphics property of the ChartPaintEventArgs class, which makes it easy to perform custom drawing in the PrePaint and PostPaint events.

> **Important**
>
> If significant changes are made to the default behavior of the **Graphics** object, the chart image and its associated elements may be drawn erratically. For example, positioning may be altered. It is highly recommended that you become well acquainted with GDI+ before you make any significant changes to the properties of the **Graphics** object.

Properties

Graphics	Gets or sets a GDI+ Graphics object that can be used for drawing purposes.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetAbsolutePoint(PointF)	Takes a PointF object and converts its relative coordinates to absolute coordinates.
GetAbsolute Rectangle(RectangleF)	Takes a RectangleF object and converts its relative coordinates to absolute coordinates.
GetAbsoluteSize(SizeF)	Takes a SizeF object that uses relative coordinates and returns a SizeF object that uses absolute coordinates.
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetPositionFromAxis(String, AxisName, Double)	Takes a given axis value for a specified axis and returns the relative pixel value.
GetRelativePoint(PointF)	Takes a PointF object that uses absolute coordinates and returns a PointF object that uses relative coordinates.
GetRelative Rectangle(RectangleF)	Takes a RectangleF structure that is using absolute coordinates and returns a RectangleF object that uses relative coordinates.
GetRelativeSize(SizeF)	Takes a SizeF object in absolute coordinates and returns a SizeF object that uses relative coordinates.
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)

ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.Graphics Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a GDI+ Graphics object that can be used for drawing purposes.

C#

```
public System.Drawing.Graphics Graphics { get; set; }
```

Property Value

Graphics

A Graphics object used for drawing.

Remarks

You can use this property to perform any custom drawing that can be performed using GDI+.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.GetAbsolutePoint(PointF) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Takes a PointF object and converts its relative coordinates to absolute coordinates.

C#

```
public System.Drawing.PointF GetAbsolutePoint (System.Drawing.PointF point);
```

Parameters

`point` PointF

A PointF object in relative coordinates.

Returns

PointF

A PointF object in absolute coordinates.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.GetAbsoluteRectangle(RectangleF) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Takes a RectangleF object and converts its relative coordinates to absolute coordinates.

C#

```
public System.Drawing.RectangleF GetAbsoluteRectangle
(System.Drawing.RectangleF rectangle);
```

Parameters

`rectangle` RectangleF

A RectangleF object in relative coordinates.

Returns

RectangleF

A RectangleF object in absolute coordinates.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.GetAbsoluteSize(SizeF) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Takes a SizeF object that uses relative coordinates and returns a SizeF object that uses absolute coordinates.

C#

```
public System.Drawing.SizeF GetAbsoluteSize (System.Drawing.SizeF size);
```

Parameters

`size` SizeF

A SizeF object in relative coordinates.

Returns

SizeF

A SizeF object in absolute coordinates.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.GetPositionFromAxis(String, AxisName, Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Takes a given axis value for a specified axis and returns the relative pixel value.

C#

```
public double GetPositionFromAxis (string chartAreaName,
System.Windows.Forms.DataVisualization.Charting.AxisName axis, double
axisValue);
```

Parameters

`chartAreaName` String

Chart area name.

`axis` AxisName

An AxisName enumeration value that identifies the relevant axis.

`axisValue` Double

The axis value to be converted to a relative pixel value.

Returns

Double

A `string` value that represents the converted axis value, in relative pixel coordinates.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.GetRelativePoint(PointF) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Takes a PointF object that uses absolute coordinates and returns a PointF object that uses relative coordinates.

C#

```
public System.Drawing.PointF GetRelativePoint (System.Drawing.PointF point);
```

Parameters

`point` PointF

A PointF object in absolute coordinates.

Returns

PointF

A PointF object in relative coordinates.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.GetRelativeRectangle(RectangleF) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Takes a RectangleF structure that is using absolute coordinates and returns a RectangleF object that uses relative coordinates.

C#

```
public System.Drawing.RectangleF GetRelativeRectangle
(System.Drawing.RectangleF rectangle);
```

Parameters

`rectangle` RectangleF

A RectangleF structure in absolute coordinates.

Returns

RectangleF

A RectangleF structure in relative coordinates.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartGraphics.GetRelativeSize(SizeF) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Takes a SizeF object in absolute coordinates and returns a SizeF object that uses relative coordinates.

C#

```
public System.Drawing.SizeF GetRelativeSize (System.Drawing.SizeF size);
```

Parameters

`size` SizeF

A SizeF object in absolute coordinates.

Returns

SizeF

A SizeF object in relative coordinates.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartHatchStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a hatching style for a chart element.

C#

```
public enum ChartHatchStyle
```

Inheritance Object → ValueType → Enum → ChartHatchStyle

Fields

BackwardDiagonal	1	Backward diagonal style.
Cross	2	Cross style.
DarkDownwardDiagonal	3	Dark downward diagonal style.
DarkHorizontal	4	Dark horizontal style.
DarkUpwardDiagonal	5	Dark upward diagonal style.
DarkVertical	6	Dark vertical style.
DashedDownwardDiagonal	7	Dashed downward diagonal style.
DashedHorizontal	8	Dashed horizontal style.
DashedUpwardDiagonal	9	Dashed upward diagonal style.
DashedVertical	10	Dashed vertical style.
DiagonalBrick	11	Diagonal brick style.
DiagonalCross	12	Diagonal cross style.
Divot	13	Divot style.
DottedDiamond	14	Dotted diamond style.

DottedGrid	15	Dotted grid style.
ForwardDiagonal	16	Forward diagonal style.
Horizontal	17	Horizontal style.
HorizontalBrick	18	Horizontal brick style.
LargeCheckerBoard	19	Large checker board style.
LargeConfetti	20	Large confetti style.
LargeGrid	21	Large grid style.
LightDownwardDiagonal	22	Light downward diagonal style.
LightHorizontal	23	Light horizontal style.
LightUpwardDiagonal	24	Light upward diagonal style.
LightVertical	25	Light vertical style.
NarrowHorizontal	26	Narrow horizontal style.
NarrowVertical	27	Narrow vertical style.
None	0	No hatching style.
OutlinedDiamond	28	Outlined diamond style.
Percent05	29	Percent05 style.
Percent10	30	Percent10 style.
Percent20	31	Percent20 style.
Percent25	32	Percent25 style.
Percent30	33	Percent30 style.
Percent40	34	Percent40 style.
Percent50	35	Percent50 style.
Percent60	36	Percent60 style.
Percent70	37	Percent70 style.
Percent75	38	Percent75 style.
Percent80	39	Percent80 style.
Percent90	40	Percent90 style.
Plaid	41	Plaid style.

Shingle	42	Shingle style.
SmallCheckerBoard	43	Small checker board style.
SmallConfetti	44	Small confetti style.
SmallGrid	45	Small grid style.
SolidDiamond	46	Solid diamond style.
Sphere	47	Sphere style.
Trellis	48	Trellis style.
Vertical	49	Vertical style.
Wave	50	Wave style.
Weave	51	Weave style.
WideDownwardDiagonal	52	Wide downward diagonal style.
WideUpwardDiagonal	53	Wide upward diagonal style.
ZigZag	54	Zigzag style.

Remarks

The ChartHatchStyle enumeration represents the hatching styles that can be used for numerous chart elements, such as the ChartArea, Legend, and so forth.

Setting the `BackHatchStyle` property of a chart element object to any member of this enumeration except None will cause the corresponding hatching style to be applied to the chart element.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartImageAlignmentStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the image alignment for a chart background image.

C#

```
public enum ChartImageAlignmentStyle
```

Inheritance Object → ValueType → Enum → ChartImageAlignmentStyle

Fields

Bottom	5	The image is aligned to the bottom boundary of the chart element.
BottomLeft	6	The image is aligned to the bottom left corner of the chart element.
BottomRight	4	The image is aligned to the bottom right corner of the chart element.
Center	8	The image is aligned in the center of the chart element.
Left	7	The image is aligned to the left boundary of the chart element.
Right	3	The image is aligned to the right boundary of the chart element.
Top	1	The image is aligned to the top boundary of the chart element.
TopLeft	0	The mage is aligned to the top left corner of the chart element.
TopRight	2	The image is aligned to the top right corner of the chart element.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartImageFormat Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the image type of the chart.

C#

```
public enum ChartImageFormat
```

Inheritance Object → ValueType → Enum → ChartImageFormat

Fields

Bmp	2	A bitmap (BMP) image format.
Emf	5	A Windows Enhanced Metafile (EMF) image format.
EmfDual	6	A Windows Enhanced Metafile Dual (EMF-dual) image format.
EmfPlus	7	A Windows Enhanced Metafile Plus (EMF+) image format.
Gif	4	A GIF image format.
Jpeg	0	A JPEG image format.
Png	1	A PNG image format.
Tiff	3	A TIFF image format.

Remarks

The ChartImageFormat enumeration represents an image type that is used to save the chart to a file or stream.

Note

Only charts saved to disk can use the EmfDual or EmfDual format. Charts cannot be rendered on a Web page using EMF-dual or EMF+.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartImageWrapMode Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the drawing mode for a background image.

C#

```
public enum ChartImageWrapMode
```

Inheritance Object → ValueType → Enum → ChartImageWrapMode

Fields

Scaled	4	Background image is scaled to fit the entire chart element.
Tile	0	Background image is tiled to fit the entire chart element.
TileFlipX	1	Every other tiled image is reversed around the X-axis.
TileFlipXY	3	Every other tiled image is reversed around the X-axis and Y-axis.
TileFlipY	2	Every other tiled image is reversed around the Y-axis.
Unscaled	100	Background image is not scaled.

Remarks

The ChartImageWrapMode enumeration represents the drawing mode for a background image, as set by the BackImageWrapMode property of the Chart control. To set a background image for a chart element, use the BackImage property.

> **Note**
>
> If you specify the Unscaled value, the location where the image is drawn will depend on the **BackImageAlignment** property setting.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElement Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the base class for most chart elements. Chart elements such as series, chart areas, and legends must have unique names, and reuse the unique name generation and validation logic provided by the ChartNamedElementCollection<T> class.

C#

```
public abstract class ChartNamedElement :
System.Windows.Forms.DataVisualization.Charting.ChartElement
```

Inheritance Object → ChartElement → ChartNamedElement

Derived System.Windows.Forms.DataVisualization.Charting.Annotation
System.Windows.Forms.DataVisualization.Charting.Axis
System.Windows.Forms.DataVisualization.Charting.ChartArea
System.Windows.Forms.DataVisualization.Charting.CustomLabel
System.Windows.Forms.DataVisualization.Charting.DataPointCustomProperties
More...

Constructors

ChartNamedElement()	Initializes a new instance of the ChartNamedElement class.
ChartNamedElement(String)	Initializes a new instance of the ChartNamedElement class with the specified chart element name.

Properties

Name	Gets or sets the name of the chart element.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources. (Inherited from ChartElement)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElement Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ChartNamedElement class.

Overloads

ChartNamedElement()	Initializes a new instance of the ChartNamedElement class.
ChartNamedElement(String)	Initializes a new instance of the ChartNamedElement class with the specified chart element name.

ChartNamedElement()

Initializes a new instance of the ChartNamedElement class.

C#

```
protected ChartNamedElement ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElement(String)

Initializes a new instance of the ChartNamedElement class with the specified chart element name.

```
C#

protected ChartNamedElement (string name);
```

Parameters

`name` String

The name of the new chart element.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElement.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the chart element.

C#

```
public virtual string Name { get; set; }
```

Property Value

String

A `string` value that represents the name of the chart element.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElementCollection<T> Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the base class for all collections of named chart elements. Performs name management and enforces the uniqueness of the names.

C#

```
public abstract class ChartNamedElementCollection<T> :
System.Windows.Forms.DataVisualization.Charting.ChartElementCollection<T>
where T : ChartNamedElement
```

Type Parameters

T

A ChartNamedElement object.

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<T>

Derived System.Windows.Forms.DataVisualization.Charting.AnnotationCollection
System.Windows.Forms.DataVisualization.Charting.ChartAreaCollection
System.Windows.Forms.DataVisualization.Charting.LegendCellCollection
System.Windows.Forms.DataVisualization.Charting.LegendCellColumnCollection
System.Windows.Forms.DataVisualization.Charting.LegendCollection
More...

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)

Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Item[String]	Gets or sets the chart element with the specified name.
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)
NamePrefix	Gets the name prefix that is used to create unique chart element names.

Methods

Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds the chart element by the specified name.
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)

GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(String)	Finds the index of the chart element with the specified name.
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index.
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection.
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection.
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection.
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element.
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, Data	Copies DataRow objects to the specified

Table, LoadOption)	DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>,	Computes the average of a sequence of Int64

Func<TSource,Int64>)	values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>,	Returns a number that represents how many

Func<TSource,Boolean>)	elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey>	Groups the elements of a sequence according

(IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.

Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.

Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Nullable<Decimal>>)	of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>,	Projects each element of a sequence into a new form.

Func<TSource,TResult>)	
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this

	method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.

Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.

ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source

	collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElement Collection<T>.Item[String] Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the chart element with the specified name.

C#

```
public T this[string name] { get; set; }
```

Parameters

name String

The name of the chart element.

Property Value

T

The ChartNamedElement object of the specified name in the collection.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElementCollection<T>.NamePrefix Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the name prefix that is used to create unique chart element names.

C#

```
protected virtual string NamePrefix { get; }
```

Property Value

String

A `string` value that represents the default name prefix of the chart elements stored in the collection.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElementCollection<T>.FindByName(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Finds the chart element by the specified name.

C#

```
public virtual T FindByName (string name);
```

Parameters

name String

The name of the chart element.

Returns

T

The ChartNamedElement object of the specified name in the collection.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElement Collection<T>.IndexOf(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Finds the index of the chart element with the specified name.

C#

```
public int IndexOf (string name);
```

Parameters

name String

The name of the chart element.

Returns

Int32

An `integer` value that represents the position of the chart element in the zero-based index.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElement Collection<T>.InsertItem(Int32, T) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Inserts the specified item in the collection at the specified index.

```
C#
protected override void InsertItem (int index, T item);
```

Parameters

index Int32

The zero-based index where the item is to be inserted.

item T

The object to insert.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElementCollection<T>.IsUniqueName(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Determines whether the chart element with the specified name already exists in the collection.

C#

```
public virtual bool IsUniqueName (string name);
```

Parameters

name String

The new chart element name.

Returns

Boolean

`true` if the new chart element name is unique; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElementCollection<T>.NextUniqueName Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Finds the unique name for the new element to be added to the collection.

C#

```
public virtual string NextUniqueName ();
```

Returns

String

A `string` value that represents the next unique chart element name.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElementCollection<T>.RemoveItem(Int32) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Removes the chart element at the specified index of the collection.

```
C#
protected override void RemoveItem (int index);
```

Parameters

`index` Int32

The zero-based index of the chart element to remove.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartNamedElementCollection<T>.SetItem(Int32, T) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Replaces the chart element at the specified index with a new chart element.

C#

```
protected override void SetItem (int index, T item);
```

Parameters

`index` Int32

The zero-based index of the chart element to replace.

`item` T

The new chart element to insert at the specified index.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartPaintEventArgs Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides data for the paint events of the Chart control.

C#

```
public class ChartPaintEventArgs : EventArgs
```

Inheritance Object → EventArgs → ChartPaintEventArgs

Remarks

A ChartPaintEventArgs object is used to set chart properties during the PostPaint and PrePaint events of the Chart control.

The Position property allows you to position chart elements, and the ChartGraphics property facilitates GDI+ drawing operations.

The Chart property gives the developer access to the root Chart object, which can be used to set various properties of the chart.

Properties

Chart	Gets the root Chart object.
ChartElement	Gets an Object for the event.
ChartGraphics	Gets a ChartGraphics object.
Position	Gets an ElementPosition object.

Methods

Equals(Object)	Determines whether the specified object is equal to the current

	object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartPaintEventArgs.Chart Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the root Chart object.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Chart Chart { get; }
```

Property Value

Chart

The root Chart object.

Remarks

The Chart property allows the developer to set chart properties during the PostPaint and PrePaint events of the Chart control.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartPaintEventArgs.ChartElement Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets an Object for the event.

C#

```
public object ChartElement { get; }
```

Property Value

Object

An Object that represents the hart element.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartPaintEventArgs.ChartGraphics Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a ChartGraphics object.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartGraphics
ChartGraphics { get; }
```

Property Value

ChartGraphics

A ChartGraphics object.

Remarks

Use this object property during a painting event to:

- Help with the conversion to and from relative and/or absolute chart coordinates.
- Perform custom drawing using a GDI+ Graphics object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartPaintEventArgs.Position Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets an ElementPosition object.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ElementPosition
Position { get; }
```

Property Value

ElementPosition

An ElementPosition object.

Remarks

Use this object property during a painting event to position a chart element.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Enables serialization of the Chart control.

C#

```
public class ChartSerializer
```

Inheritance Object → ChartSerializer

Remarks

This class is exposed as the Serializer property of the root Chart class.

Serialization saves the state of the chart and also provides the option to load the serialized data into the chart. All chart properties can be persisted, including the chart's data. In addition, all chart properties marked for serialization can be reset.

This class can optionally be used when managing user-defined view state. One of its `Load` methods can be used to load the view state, and one of the `Save` methods can be used when setting the view state.

The XML or binary formats can be used to store serialized data. XML is used by default, but you can change the format with the Format property.

Chart properties are saved and loaded using the `Save` and `Load` methods, respectively. Note that only properties with non-default values will be saved. Data can be saved and/or loaded using a file, stream, TextWriter, TextReader, XmlWriter and XmlReader objects. When saving and/or loading data using an object derived from the TextReader, TextWriter, XmlReader or XmlWriter classes, the format of the data must be XML.

The following list describes three techniques, from basic to advanced, used to define serializable data. Note that it is also possible to define non-serializable data. This technique is described in the next paragraph.

- Serialize the Chart using the `Save` and `Load` methods, without specifying any serializable properties. All chart properties with non-default values, which includes chart data, will be serialized.

- Serialize the chart properties specified by the Content property. The Content property groups the chart properties to be serialized into categories such as Data, Appearance, and so forth. Note that the names of ChartArea and Series objects are automatically serialized; the serialized data is then applied to existing chart areas and series when loaded.

> ⓘ **Important**
>
> The **Content** property is implemented internally by the **Chart** control as the **SerializableContent** property

- Serialization of chart properties specified by the SerializableContent property. This is a comma-separated listing of all chart properties that are serializable. After this property is set to an explicit value, it is the responsibility of the developer to specify all properties to be saved. In other words, unlike the Content property, the names of chart areas and series will not be automatically persisted. The Content and SerializableContent properties can also be used in conjunction. However, make sure that SerializableContent is concatenated with itself when being set, otherwise the chart properties specified by the Content property will be overridden.

To prohibit certain properties from being serialized, use the NonSerializableContent property. When one property is set as serializable and non-serializable, which is common when wildcards are used, the SerializableContent property has precedence. However, precedence also depends on how many wildcards were used when a property was specified.

By default, serializable properties are reset to their defaults before they are loaded. This behavior can be changed using the IsResetWhenLoading property. The Reset method can also be used to explicitly reset chart properties to their default values.

Properties

Content	Gets or sets the type of chart content to be serialized.
Format	Gets or sets the format used when the chart is serialized.
IsResetWhenLoading	Gets or sets a flag that indicates whether object properties are reset to default values before loading.

IsTemplateMode	Gets or sets a flag that indicates whether the chart serializer is working in template-creation mode.
IsUnknownAttributeIgnored	Gets or sets a flag that indicates whether unknown XML properties and elements will be ignored without throwing an exception.
NonSerializableContent	Gets or sets the chart properties that will not be serialized.
SerializableContent	Gets or sets the chart properties that can be serialized.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetContentString(SerializationContents, Boolean)	Returns a serializable or non-serializable class and/or property names, depending on the specific flag value.
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
Load(Stream)	Loads serialized data into the control from an object derived from the Stream class.
Load(String)	Loads serialized data saved to disk into the Chart control.
Load(TextReader)	Loads serialized data into the control from a reader object that derives from the TextReader class.
Load(XmlReader)	Loads serialized data into the control from an object derived from the XmlReader class.
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Reset()	Resets all chart properties marked for serialization.
Save(Stream)	Saves chart data and properties with non-default values to an object that derives from the Stream class.
Save(String)	Saves chart data and properties with non-default values to the given file.
Save(TextWriter)	Saves chart data and properties with non-default values to the given object that is derived from the TextWriter class.

Save(XmlWriter)	Saves chart data and properties with non-default values to the given object that is derived from the XmlWriter class.
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.Content Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the type of chart content to be serialized.

C#

```
public System.Windows.Forms.DataVisualization.Charting.SerializationContents
Content { get; set; }
```

Property Value

SerializationContents

A SerializationContents enumeration value that represents the chart content that will be serialized during a load, save or reset operation. By default, all non-default chart properties, including data, are persisted.

Remarks

By default, all chart properties will be serialized when any of the `Save` and `Load` methods are called. To provide more control over what is serialized, use the Content property. Note that the SerializableContent and NonSerializableContent properties can also be used along with the Content property.

> ⓘ **Important**
>
> The **Content** property determines types of chart characteristics, such as "appearance", to be serialized, and is actually implemented internally as the **SerializableContent** property. This means that setting the **Content** property and then setting the **SerializableContent** property will result in the **Content** property value being overridden, unless **SerializableContent** is concatenated. See the sample code below for an example of this procedure.

If you set the Content property, all ChartArea and Series objects names will automatically be serialized; this means that serialized data that pertains to chart areas and series will be applied to existing objects when loaded.

Setting this property affects all load, save and reset operations.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.Format Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the format used when the chart is serialized.

C#

```
public System.Windows.Forms.DataVisualization.Charting.SerializationFormat
Format { get; set; }
```

Property Value

SerializationFormat

A SerializationFormat enumeration value that represents the format of the serialized data. The default value is XML.

Remarks

Data can be serialized in either an XML or binary format. Binary format is compact, and takes less storage than XML. XML, however, is a text format that is human-readable.

When you save or load serialized data, the format must be XML.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.IsResetWhenLoading Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether object properties are reset to default values before loading.

C#

```
public bool IsResetWhenLoading { get; set; }
```

Property Value

Boolean

`true` if object properties are reset to default values before loading; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.IsTemplateMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the chart serializer is working in template-creation mode.

```
C#

public bool IsTemplateMode { get; set; }
```

Property Value

Boolean

`true` if the chart serializer is working in template-creation mode; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.IsUnknownAttributeIgnored Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether unknown XML properties and elements will be ignored without throwing an exception.

C#

```
public bool IsUnknownAttributeIgnored { get; set; }
```

Property Value

Boolean

`true` if unknown XML properties and elements are ignored without throwing an exception; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.NonSerializableContent Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the chart properties that will not be serialized.

```
C#
public string NonSerializableContent { get; set; }
```

Property Value

String

A comma-separated `string` expression that represents the chart properties that will not be serialized. The syntax is "Class.Property[,Class.Property]".

Remarks

By default all chart properties will be serialized when the `Save` and `Load` methods are called. You can use the NonSerializableContent property to limit chart characteristics to be serialized, along with the SerializableContent and Content properties.

This property determines which chart characteristics will not be serialized. Wildcards can be used in the string expression, in the same manner as the SerializableContent property. For example, to exclude all chart `BackColor` properties from serialization, set this property to "*.BackColor".

Sometimes, a property can be set to be both serialized and not serialized, which is common when wildcards are used. The NonSerializableContent expression has a lower priority when compared to the SerializableContent expression. However, note that less weight is given to string expressions that use wildcards. For example, if the SerializableContent property is set to "*.BackColor" and the NonSerializableContent

property is set to BackColor, all `BackColor` properties except for ChartArea objects will be serialized.

If the Content property has been set, the priority depends on how many wild cards are used in the NonSerializableContent expression. For example, setting the Content property to Appearance will result in the SerializableContent property containing a "*.Back*" expression. If the NonSerializableContent property is set to "*.Back*", all `Back\*` chart properties, such as `BackColor`, `BackGradient`, and so forth, will still be serialized. This is because the SerializableContent property value has higher priority when compared to the NonSerializableContent property.

This property affects all save, load and reset operations.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.SerializableContent Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the chart properties that can be serialized.

C#

```
public string SerializableContent { get; set; }
```

Property Value

String

A comma-separated `string` expression that represents the chart properties to be serialized. The syntax is "Class.Property[,Class.Property]".

Remarks

By default all chart properties will be serialized when the `Save` and `Load` methods are called. You can use the NonSerializableContent property to limit chart characteristics to be serialized, along with the SerializableContent and Content properties.

The SerializableContent property determines which chart characteristics will be serialized. Wildcards can be used in the string expression, in the same manner as the NonSerializableContent property. For example, to serialize all chart `BackColor` properties, set this property to "*.BackColor".

If ChartArea and/or Series child properties are specified, then the names of all ChartArea and Series objects must be also specified to enable the persisted data to be applied to existing ChartArea and Series objects; that is, you must specify "Series.Name" and "ChartArea.Name" expressions. See the sample code below for an example of this procedure. Similarly, if multiple chart areas are being used, the ChartArea property of all plotted Series must also be saved.

Sometimes, a property can be set to be both serialized and not serialized, which is common when wildcards are used. The NonSerializableContent expression has a lower priority when compared to the SerializableContent expression. However, note that less weight is given to string expressions that use wildcards. For example, if the SerializableContent property is set to "*.BackColor" and the NonSerializableContent property is set to BackColor, all B`ackColor` properties except for ChartArea objects will be serialized.

If you set the Content property, the SerializableContent property will be set internally. If the Content property is already set but another property needs to be serialized, make sure to concatenate the SerializableContent property when specifying that extra property. See the sample code below for an example of this procedure.

This property affects all save, load and reset operations.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.GetContentString(SerializationContents, Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns a serializable or non-serializable class and/or property names, depending on the specific flag value.

C#

```
protected string GetContentString
(System.Windows.Forms.DataVisualization.Charting.SerializationContents
content, bool serializable);
```

Parameters

content SerializationContents

The serializable content.

serializable Boolean

`true` to get a serializable string; `false` to get a non-serializable string.

Returns

String

A `string` value that represents the serializable or non-serializable class and/or property names.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.Load Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Loads serialized data into the Chart control.

Overloads

Load(XmlReader)	Loads serialized data into the control from an object derived from the XmlReader class.
Load(String)	Loads serialized data saved to disk into the Chart control.
Load(Stream)	Loads serialized data into the control from an object derived from the Stream class.
Load(TextReader)	Loads serialized data into the control from a reader object that derives from the TextReader class.

Load(XmlReader)

Loads serialized data into the control from an object derived from the XmlReader class.

C#

```
public void Load (System.Xml.XmlReader reader);
```

Parameters

`reader` XmlReader

An object from which to read the persisted data. The object is derived from the XmlReader class, and can be a .NET Framework object or a user-defined object.

Remarks

Call this method to load serialized data into the Chart control.

When a load operation occurs, by default serializable properties are first reset to their default values before they are set to the persisted values, if saved. To change this behavior, use the IsResetWhenLoading property.

By default, all serialized chart properties will be loaded. If you set the Content, SerializableContent or NonSerializableContent properties, a specific set of properties can be loaded.

When you use this definition, the format must be XML; the format can be changed with the Format property.

When you work with XML format text files, for simplicity use the Load method instead.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Load(String)

Loads serialized data saved to disk into the Chart control.

C#

```
public void Load (string fileName);
```

Parameters

`fileName` String

The relative or absolute path of the file used to store the serialized data. If a relative path is specified, the path will be relative to the current directory.

Remarks

Call this method to load serialized data into the Chart control.

When a load operation occurs, serializable properties are, by default, first reset to their default values before they are set to the persisted values, if saved. To change this behavior, use the IsResetWhenLoading property.

By default, all serialized chart properties will be loaded. To load specific serialized properties, set the Content , SerializableContent or NonSerializableContent properties before loading the data.

Note that when data is saved using a `Save` method, only those properties that have non-default values are actually serialized.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Load(Stream)

Loads serialized data into the control from an object derived from the Stream class.

C#

```
public void Load (System.IO.Stream stream);
```

Parameters

`stream` Stream

An object derived from the Stream class, for example, a MemoryStream object.

Remarks

Call this method to load serialized data into the Chart control.

When a load operation occurs, serializable properties are, by default, first reset to their default values before they are set to the persisted values, if saved. To change this behavior, use the IsResetWhenLoading property.

By default, all serialized chart properties will be loaded. To load specific serialized properties, set the Content , SerializableContent or NonSerializableContent

properties before loading the data.

Note that when data is saved using a `Save` method, only those properties that have non-default values are actually serialized.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Load(TextReader)

Loads serialized data into the control from a reader object that derives from the TextReader class.

C#

```
public void Load (System.IO.TextReader reader);
```

Parameters

reader TextReader

A reader, such as StringReader, that is derived from the abstract TextReader class.

Remarks

Call this method to load serialized data into the Chart control.

When a load operation occurs, by default the serializable properties are first reset to their default values before they are set to the persisted values, if saved. To change this behavior, use the IsResetWhenLoading property.

By default, all serialized chart properties will be loaded. By setting the Content, SerializableContent or NonSerializableContent properties, a specific set of properties can be loaded.

If you use this definition, note that the format, which is set by the Format property, must be XML.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.Reset Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Resets all chart properties marked for serialization.

C#

```
public void Reset ();
```

Remarks

By default, this method resets all chart properties to their default values.

To limit the chart properties that are reset to their defaults, use the Content, SerializableContent and NonSerializableContent properties.

Note that this method is also called automatically when a `Load` method is used and the IsResetWhenLoading property is set to `true`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartSerializer.Save Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Saves chart properties that have non-default values. All chart properties can be serialized; this includes data.

Overloads

Save(XmlWriter)	Saves chart data and properties with non-default values to the given object that is derived from the XmlWriter class.
Save(Stream)	Saves chart data and properties with non-default values to an object that derives from the Stream class.
Save(TextWriter)	Saves chart data and properties with non-default values to the given object that is derived from the TextWriter class.
Save(String)	Saves chart data and properties with non-default values to the given file.

Save(XmlWriter)

Saves chart data and properties with non-default values to the given object that is derived from the XmlWriter class.

C#

```
public void Save (System.Xml.XmlWriter writer);
```

Parameters

`writer` XmlWriter

An object derived from XmlWriter used to save Chart properties.

Remarks

By default, all chart properties with non-default values, which includes data points, will be serialized when this method is called. To control which chart characteristics are serialized, use the Content, SerializableContent and NonSerializableContent properties.

When you use this definition, the format must be XML; the format can be changed with the Format property.

When you work with XML format text files, for simplicity use the Save method instead.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Save(Stream)

Saves chart data and properties with non-default values to an object that derives from the Stream class.

C#

```
public void Save (System.IO.Stream stream);
```

Parameters

`stream` Stream

An object derived from the Stream class, such as MemoryStream, that is used to save chart properties.

Remarks

By default, all chart properties with non-default values, which includes data points, will be serialized when this method is called. To control chart characteristics to be serialized, use the Content, SerializableContent and NonSerializableContent properties.

If this method is used to save appearance properties as a template, make sure that the IsTemplateMode property of the ChartSerializer object is first set to `true` before calling this method.

Serialization of a chart as a template occurs differently from non-template serialization; template serialization concentrates on Appearance properties.

When a template is serialized, the LoadTemplate method can be used to load the template, to provide the Appearance properties for a chart.

Templates can be used to simulate custom palettes.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Save(TextWriter)

Saves chart data and properties with non-default values to the given object that is derived from the TextWriter class.

C#

```
public void Save (System.IO.TextWriter writer);
```

Parameters

`writer` TextWriter

An object derived from TextWriter class, for example, StringWriter, that is used to save chart properties.

Remarks

By default, all chart properties with non-default values, which includes data points, will be serialized when this method is called. To control which chart characteristics are serialized, use the Content, SerializableContent and NonSerializableContent properties.

When you use this definition, the format must be XML; the format can be set with the Format property.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Save(String)

Saves chart data and properties with non-default values to the given file.

C#

```
public void Save (string fileName);
```

Parameters

`fileName` String

The relative or absolute path of the file used to store the serialized data. Note that if a relative path is specified, the path will be relative to the current directory.

Remarks

By default, all chart properties with non-default values, which includes data points, will be serialized when this method is called. To control chart characteristics to be serialized use the Content, SerializableContent and NonSerializableContent properties.

By default data is saved in an XML format. To save the data in a binary format, use the Format property.

Note that Write permissions must be enabled for the directory in which the file is stored. By default, this is the Bin folder of the application.

If this method is being used to save Appearance properties as a template, make sure that the IsTemplateMode property is first set to `true` before calling this method.

Serialization of a chart as a template proceeds differently from non-template serialization, primarily because template serialization concentrates on Appearance properties.

When a template is serialized, the LoadTemplate method can be used to load the template, to provide the Appearance properties for a chart.

Templates can be used to simulate custom palettes.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ChartValueType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies value types for various chart properties.

C#

```
public enum ChartValueType
```

Inheritance Object → ValueType → Enum → ChartValueType

Fields

Auto	0	Property type is set automatically by the Chart control.
Date	9	The Date portion of a DateTime value.
DateTime	8	A DateTime value.
DateTimeOffset	11	A DateTime value with offset.
Double	1	A Double value.
Int32	3	A Int32 value.
Int64	4	A Int64 value.
Single	2	A Single value.
String	7	A String value.
Time	10	The Time portion of the DateTime value.
UInt32	5	A UInt32 value.
UInt64	6	A UInt64 value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CompareMethod Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the type of comparison to be used when filtering data. This enumeration is used by the `Filter` method definitions in the DataManipulator class that take the `CompareMethod` argument.

```
C#

public enum CompareMethod
```

Inheritance Object → ValueType → Enum → CompareMethod

Fields

EqualTo	2	If the data point value is equal to the compared value, the point is filtered.
LessThan	1	If the data point value is less than the compared value, the point is filtered.
LessThanOrEqualTo	4	If the data point value is less than or equal to the compared value, the point is filtered.
MoreThan	0	If the data point value is more than the compared value, the point is filtered.
MoreThanOrEqualTo	3	If the data point value more than or equal to the compared value, the point is filtered.
NotEqualTo	5	If the data point value is not equal to the compared value, the point is filtered.

Remarks

The CompareMethod enumeration represents a comparison type for certain filtering operations.

> **Note**
>
> By default, if the filtering criteria are met, a data point is either removed or marked as empty. However, data points can be filtered even if they do not match the filtering criteria, by setting the **FilterMatchedPoints** property to `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a horizontal or vertical line that defines a position along an axis.

C#

```
public class Cursor : IDisposable
```

Inheritance Object → Cursor

Implements IDisposable

Remarks

A cursor is a horizontal or vertical line that defines a position along an axis. A range selection is a range along an axis that is defined by a beginning and end position, and is displayed using a semi-transparent color.

Cursors and range selections are both implemented by the Cursor class, which is exposed as the CursorX and CursorY properties of the ChartArea object. The CursorX object is for the X axis of a chart area, and the CursorY object is for the Y axis. The AxisType property of these objects determines whether the associated axis is primary or secondary.

The only cursor and range selection functionality found outside the Cursor class are the events related to the cursor and range selection that are in the root Chart object.

Constructors

Cursor()	Initializes a new instance of the Cursor class.

Properties

AutoScroll	Gets or sets a flag that determines whether scrolling will occur if a range selection operation extends beyond a boundary of the chart area.
AxisType	Gets or sets the type of axis that the cursor is attached to.
Interval	Gets or sets the cursor interval.
IntervalOffset	Gets or sets the interval offset, which affects where the cursor and range selection can be drawn when they are set by a user.
IntervalOffsetType	Gets or sets the interval offset type of a cursor and selected range for an axis.
IntervalType	Gets or sets the interval type for the cursor and selected range of an axis.
IsUserEnabled	Gets or sets a flag that enables or disables the cursor user interface.
IsUserSelectionEnabled	Gets or sets a flag that enables or disables the range selection user interface.
LineColor	Gets or sets the cursor line color.
LineDashStyle	Gets or sets the style of the cursor line.
LineWidth	Gets or sets the line width, in pixels, of a cursor.
Position	Gets or sets the position of a cursor.
SelectionColor	Gets or sets a semi-transparent color that highlights a range of data.
SelectionEnd	Gets or sets the end position of a range selection.
SelectionStart	Gets or sets the start position of a cursor's selected range.

Methods

Dispose()	Performs application-defined tasks that free, release, or reset unmanaged resources.
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function.

	(Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
SetCursorPixelPosition(PointF, Boolean)	Displays a cursor at the specified position. Measured in pixels.
SetCursorPosition(Double)	Sets the position of a cursor within a chart area at the specified axis value.
SetSelectionPixel Position(PointF, PointF, Boolean)	Sets the start and end positions of a range selection.
SetSelectionPosition(Double, Double)	Sets the position of a selected range within a chart area at the specified axis values.
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Cursor class.

C#

```
public Cursor ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.AutoScroll Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether scrolling will occur if a range selection operation extends beyond a boundary of the chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public bool AutoScroll { get; set; }
```

Property Value

Boolean

`true` if the data view can be scrolled automatically; otherwise, `false`.

Attributes BindableAttribute

Remarks

If a data view does not display all data points in a plotted series and a range is being selected by the user, automatic scrolling of the view will occur if this property is `true` and the mouse - and therefore the cursor - is moved outside the chart area boundary.

Scrolling will only occur along an axis if there is more data available to be viewed.

A data view that does not display all data usually results if the end-user zooms into a subset of data. Views and zooming are implemented by the AxisScaleView class.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.AxisType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the type of axis that the cursor is attached to.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.AxisType AxisType {
get; set; }
```

Property Value

AxisType

An AxisType enumeration value that represents whether the axis is attached to the primary or secondary axis.

Attributes BindableAttribute

Remarks

There are two sets of axes: primary and secondary. Primary axes for all charts except the bar charts are located at the bottom (primary X-axis) and left (primary Y-axis) border of the chart area.

The secondary axes are located at the right (secondary X-axis) and top (secondary Y-axis) border of the chart area.

> **Note**
>
> If all series plotted within a chart area use only one type of axis (for example, the primary X-axis), then setting this property to an axis type not used by the series (in this case, the secondary X-axis) will have no effect; the cursor will automatically be attached to the only axis being used to plot the series.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.Interval Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the cursor interval.

C#

```
[System.ComponentModel.Bindable(true)]
public double Interval { get; set; }
```

Property Value

Double

The cursor interval.

Attributes BindableAttribute

Remarks

The Interval property determines how often and where a cursor can be drawn by an end user, as well as where the beginning and end points of a range selection can be drawn. A value of zero, which is the default value, signifies that a cursor or range selection can be drawn anywhere within a chart area. A nonzero value signifies that the cursor or beginning and/or end points of a selected range will only be drawn at the specified interval when set by user interaction.

The unit of measurement is determined by the IntervalType property.

This property can also be used in conjunction with the IntervalOffset property, which allows you to specify an offset from the specified interval. For example, to allow a cursor to only be displayed every Saturday, an interval of 1 week should be specified, along with an interval offset of 6 days. In .NET Framework, Sunday is considered to be the first day of the week.

If the cursor user interface is enabled by setting the IsUserEnabled property, and the mouse is clicked inside a chart area, the position of the cursor will be set to the nearest allowable location.

> **Important**
>
> The **Interval** property has no effect if you set the position of a cursor or range selection programmatically.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.IntervalOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval offset, which affects where the cursor and range selection can be drawn when they are set by a user.

C#

```
[System.ComponentModel.Bindable(true)]
public double IntervalOffset { get; set; }
```

Property Value

Double

The offset from the interval. The default value is zero, which signifies no limitations, and negative values are allowed.

Attributes BindableAttribute

Remarks

The IntervalOffset property determines the interval offset of a cursor and the beginning and/or end points of a range selection. In conjunction with the Interval property, this property determines where a cursor and range can be displayed within a chart area. A value of zero, which is the default, signifies that there is no offset.

The unit of measurement is determined by the IntervalOffsetType property.

The offset affects the locations at which a cursor and the beginning and/or end points of a range selection can be drawn. For example, to allow a cursor to only be displayed every Saturday, an interval of 1 week should be specified, along with an interval offset of 6 days. In .NET Framework, Sunday is considered to be the first day of the week.

If the cursor user interface is enabled by setting the IsUserEnabled property, and the mouse is clicked inside a chart area, the position of the cursor will be set to the nearest

allowable location.

> **Important**
>
> Interval-related properties have no effect when a range selection or cursor position are set programmatically.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.IntervalOffsetType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval offset type of a cursor and selected range for an axis.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalOffsetType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that determines the unit of measurement for the interval offset.

Attributes BindableAttribute

Remarks

The IntervalOffsetType property determines the type of interval offset, which is specified using the IntervalOffset property.

If the IntervalOffsetType property is set to Auto, the IntervalType property setting will be used.

> **Note**
>
> Interval-related properties have no effect when a range selection or cursor position are set programmatically.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.IntervalType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval type for the cursor and selected range of an axis.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalType { get; set; }
```

Property Value

DateTimeIntervalType

The interval type for the cursor and selected range of an axis.

Attributes BindableAttribute

Remarks

The IntervalType property determines the type of interval, which is specified using the Interval property.

If the interval type is Auto, the unit of measurement for the interval is treated as a `double`, which is equivalent to one day when DateTime values are specified.

> **Note**
>
> Interval-related properties have no effect when a range selection or cursor position is set programmatically.

If the axis interval type represents a time - for example, hours, minutes, and so forth - then in order to correctly enable the zooming function, the IntervalType must also be set

to a time value. For example, if the ChartArea1.AxisX.IntervalType is expressed in hours, then the ChartArea1.CursorX.IntervalType could be set to minutes.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.IsUserEnabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that enables or disables the cursor user interface.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsUserEnabled { get; set; }
```

Property Value

Boolean

`true` if the cursor user interface is enabled; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.IsUserSelectionEnabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that enables or disables the range selection user interface.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsUserSelectionEnabled { get; set; }
```

Property Value

Boolean

`true` if the range selection user interface is enabled; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the cursor line color.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

The line color of the cursor. The default value is Red.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the cursor line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
LineDashStyle { get; set; }
```

Property Value

ChartDashStyle

The style of the cursor line.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line width, in pixels, of a cursor.

C#

```
[System.ComponentModel.Bindable(true)]
public int LineWidth { get; set; }
```

Property Value

Int32

The line width, in pixels. The default value is one (1).

Attributes BindableAttribute

Remarks

Use this property to specify the line width, in pixels, of a cursor.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.Position Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the position of a cursor.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleDateNanValueConverter))]
public double Position { get; set; }
```

Property Value

Double

The position of a cursor.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property can be used to get the position of a cursor, as well as to programmatically set the cursor position.

The position is measured as a value of the axis that the cursor belongs to, which is determined by the AxisType property. For example, if the cursor belongs to the primary X axis, then setting this property to a value of January 1, 2000 will result in the X-axis cursor being drawn at the January 1, 2000 position along the axis. If this axis value does not exist, the cursor will not be displayed.

Note that if you set this property, the Interval and IntervalOffset properties will have no effect; that is, the cursor will be displayed at this specified location.

Setting this property will not raise the CursorPositionChanged or CursorPositionChanging events.

Alternatively, the SetCursorPosition and SetCursorPixelPosition methods can be used to set the cursor.

To programmatically remove a cursor, use this property to set the position of the cursor to NaN.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.SelectionColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a semi-transparent color that highlights a range of data.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color SelectionColor { get; set; }
```

Property Value

Color

The color of the highlighted range. The default value is LightGray, with an alpha value of 120.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property determines the color of the range selection, and can be set to any valid ARGB (alpha, red, green, blue) value. If an alpha value, which determines the level of transparency, is not provided, a default alpha value of 120 is used.

The selected color can be set for both the X and Y axis cursors, which is implemented using the CursorX and CursorY properties of the ChartArea object, respectively. If a range is only selected along the X-axis, the SelectionColor value for the X-axis cursor is used; if a range is only selected along the Y-axis, the SelectionColor value for the Y-axis cursor is used. If a range is selected along both axes, the SelectionColor value for the Y-axis cursor is also used for the selected color.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.SelectionEnd Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the end position of a range selection.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleDateNanValueConverter))]
public double SelectionEnd { get; set; }
```

Property Value

Double

The end position of a range selection.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

A user can select a range within a chart area by clicking on the left mouse button and then dragging the mouse. Note that the IsUserSelectionEnabled property must also be set to `true`. In addition, a range selection can be set programmatically using either the SelectionEnd and SelectionStart properties, or the SetSelectionPosition or SetSelectionPixelPosition methods.

The end position of a range selection is calculated as a value of the axis to which the cursor belongs. For example, if the cursor belongs to the primary X-axis, and you set the SelectionEnd property to a value of December 31, 2000, the end of the selected area will occur at December 31, 2000 along the primary X axis.

To programmatically remove a range selection, set the beginning and end points of the range to NaN using the SelectionEnd and SelectionStart properties, or use the SetSelectionPosition method.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.SelectionStart Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the start position of a cursor's selected range.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleDateNanValueConverter))]
public double SelectionStart { get; set; }
```

Property Value

Double

The start position of a cursor's selected range.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

A user can select a range by clicking on the left mouse button and then moving the cursor position by dragging the mouse. Note that the IsUserSelectionEnabled property must also be set to `true`. In addition, a range selection can be set programmatically using either the SelectionEnd and SelectionStart properties, or the SetSelectionPosition or SetSelectionPixelPosition methods.

The start position of a range selection is calculated as a value of the axis to which the cursor belongs. For example, if the cursor belongs to the primary X-axis, and you set the SelectionStart property to a value of January 1, 2000, the beginning of the selected area will occur at January 1, 2000 along the primary X axis.

To programmatically remove a range selection, set the beginning and end points of the range to NaN using the SelectionEnd and SelectionStart properties, or use the SetSelectionPosition method.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.Dispose Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the resources used by the Cursor class.

Overloads

Dispose()	Performs application-defined tasks that free, release, or reset unmanaged resources.
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.

Dispose()

Performs application-defined tasks that free, release, or reset unmanaged resources.

C#

```
public void Dispose ();
```

Implements

Dispose()

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Dispose(Boolean)

Releases unmanaged and, optionally, managed resources.

```
C#

protected virtual void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.SetCursorPixelPosition(PointF, Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Displays a cursor at the specified position. Measured in pixels.

C#

```
public void SetCursorPixelPosition (System.Drawing.PointF point, bool
roundToBoundary);
```

Parameters

`point` PointF

A PointF structure that specifies where the cursor will be drawn.

`roundToBoundary` Boolean

If `true`, the cursor will be drawn along the nearest chart area boundary, if the specified position does not fall within a ChartArea object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.SetCursorPosition(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets the position of a cursor within a chart area at the specified axis value.

C#

```
public void SetCursorPosition (double newPosition);
```

Parameters

`newPosition` Double

The new position of the cursor. Measured as a value along the relevant axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.SetSelectionPixelPosition(PointF, PointF, Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets the start and end positions of a range selection.

C#

```
public void SetSelectionPixelPosition (System.Drawing.PointF startPoint,
System.Drawing.PointF endPoint, bool roundToBoundary);
```

Parameters

`startPoint` PointF

A PointF structure that specifies where the range selection begins.

`endPoint` PointF

A PointF structure that specifies where the range selection ends.

`roundToBoundary` Boolean

If `true`, the start and end points will be rounded to the nearest chart area boundary if the specified positions do not fall within a ChartArea object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Cursor.SetSelectionPosition(Double, Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets the position of a selected range within a chart area at the specified axis values.

C#

```
public void SetSelectionPosition (double newStart, double newEnd);
```

Parameters

`newStart` Double

The new start position of the range selection. Measured as a value along the relevant axis.

`newEnd` Double

The new end position of the range selection. Measured as a value along the relevant axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CursorEventArgs Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides properties that are used with the following events in the root Chart object: CursorPositionChanged, CursorPositionChanging, SelectionRangeChanged and SelectionRangeChanging.

C#

```
public class CursorEventArgs : EventArgs
```

Inheritance Object → EventArgs → CursorEventArgs

Remarks

The CursorEventArgs class is exposed as the `e` parameter in the CursorPositionChanged, CursorPositionChanging, SelectionRangeChanged and SelectionRangeChanging events of the root Chart object.

This class contains the following properties:

- The ChartArea property, which is used to get the ChartArea object that the event was raised for. The Name property gets the name of the chart area; other chart area properties can also be set.
- The Axis property, which is used to get the Axis object that the event was raised for. To determine the type of axis - X, X2, Y or Y2 - use the AxisName property. Other axis properties can also be set.
- The NewPosition property, which represents the position of a cursor along an axis, measured as an axis value, at the time the event was raised. The NewPosition property can be used to get the position of a cursor. In the CursorPositionChanging event, this property can also be used to reset the cursor position.

- The NewSelectionStart property, which represents the starting position of a range selection along an axis, measured as an axis value, at the time the event was raised. The NewSelectionStart property can be used to get the selected range start position. In the SelectionRangeChanging event, this property can also be used to reset the position.

- The NewSelectionEnd property, which represents the end position of a range selection along an axis, measured as an axis value, at the time the event was raised. The NewSelectionEnd property can be used to get the selected range end position. In the SelectionRangeChanging event, this property can also be used to reset the position.

Constructors

CursorEventArgs(ChartArea, Axis, Double)	Initializes a new instance of the CursorEventArgs class with the specified chart area, axis, and cursor position.
CursorEventArgs(ChartArea, Axis, Double, Double)	Initializes a new instance of the CursorEventArgs class with the specified chart area, axis, start and end positions for the new range.

Properties

Axis	Gets the Axis object that a cursor or range selection belongs to.
ChartArea	Gets the ChartArea object that a cursor or range selection belongs to.
NewPosition	Gets or sets the position of a cursor. Depending on the event, also sets the cursor's position.
NewSelectionEnd	Gets or sets the end position of a selected range. Depending on the event, also sets the end position of the range.
NewSelectionStart	Gets or sets the start position of a selected range. Depending on the event, also sets the start position of the range.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)

GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CursorEventArgs Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the CursorEventArgs class.

Overloads

CursorEventArgs(ChartArea, Axis, Double)	Initializes a new instance of the CursorEventArgs class with the specified chart area, axis, and cursor position.
CursorEventArgs(ChartArea, Axis, Double, Double)	Initializes a new instance of the CursorEventArgs class with the specified chart area, axis, start and end positions for the new range.

CursorEventArgs(ChartArea, Axis, Double)

Initializes a new instance of the CursorEventArgs class with the specified chart area, axis, and cursor position.

C#

```
public CursorEventArgs
(System.Windows.Forms.DataVisualization.Charting.ChartArea chartArea,
System.Windows.Forms.DataVisualization.Charting.Axis axis, double
newPosition);
```

Parameters

chartArea ChartArea

The ChartArea of the cursor.

axis Axis

The axis of the cursor.

`newPosition` Double

The new cursor position.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CursorEventArgs(ChartArea, Axis, Double, Double)

Initializes a new instance of the CursorEventArgs class with the specified chart area, axis, start and end positions for the new range.

C#

```
public CursorEventArgs
(System.Windows.Forms.DataVisualization.Charting.ChartArea chartArea,
System.Windows.Forms.DataVisualization.Charting.Axis axis, double
newSelectionStart, double newSelectionEnd);
```

Parameters

`chartArea` ChartArea

The ChartArea of the cursor.

`axis` Axis

The axis of the cursor.

`newSelectionStart` Double

The start position of the new range selection.

`newSelectionEnd` Double

The end position of the new range selection.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CursorEventArgs.Axis Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Axis object that a cursor or range selection belongs to.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Axis Axis { get; }
```

Property Value

Axis

An Axis object that represents the axis that a cursor or range selection belongs to.

Remarks

The Axis property is exposed in the following events of the root Chart object: CursorPositionChanged, CursorPositionChanging, SelectionRangeChanged and SelectionRangeChanging.

This property can be used to:

- determine the type of axis - X, X2, Y or Y2 - that the event is being raised for, using the AxisName property.
- get and set axis properties, such as LineColor.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CursorEventArgs.ChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the ChartArea object that a cursor or range selection belongs to.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartArea ChartArea {
get; }
```

Property Value

ChartArea

An object of type ChartArea that represents the chart area that a range selection or cursor belongs to.

Remarks

The ChartArea property is exposed in the following events of the root Chart object: CursorPositionChanged, CursorPositionChanging, SelectionRangeChanged and SelectionRangeChanging.

This property can be used to:

- determine which chart area the event is being raised for; this is useful to know when there are multiple chart areas within a chart image.
- get and set chart area properties, such as BackColor.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CursorEventArgs.NewPosition Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the position of a cursor. Depending on the event, also sets the cursor's position.

```
C#
public double NewPosition { get; set; }
```

Property Value

Double

A `double` value that represents the position of a cursor along an axis.

Remarks

This property represents the new position of a cursor, and can be used in the CursorPositionChanging and CursorPositionChanged events to get the cursor position. It can also be used in the SelectionRangeChanging event to set the cursor position.

Note that setting this property in the SelectionRangeChanging event will have no effect. The NewPosition property will have a value of NaN in this event.

To determine the chart area that the event is being raised for, use the ChartArea property.

Use the Axis property to determine if this end position is for a range selected along the primary or secondary X or Y axis.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CursorEventArgs.NewSelectionEnd Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the end position of a selected range. Depending on the event, also sets the end position of the range.

```
C#

public double NewSelectionEnd { get; set; }
```

Property Value

Double

A `double` value that represents the end position of a selected range.

Remarks

The NewSelectionEnd property represents the new position of a cursor, and can be used in the CursorPositionChanging and CursorPositionChanged events to get the cursor position. It can also be used in the SelectionRangeChanging event to set the cursor position.

Note that setting this property in the SelectionRangeChanging event will have no effect. The NewSelectionEnd property will have a value of NaN in this event.

The start position of a selected range is not necessarily less than the end position of a selected range. The start position is the position where the user fist clicked within the chart area, and the end position is the position of the mouse at the time the event is raised.

To determine the chart area that the event is being raised for use the ChartArea property.

Use the Axis property to determine if this end position is for a range selected along the primary or secondary X or Y axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CursorEventArgs.NewSelectionStart Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the start position of a selected range. Depending on the event, also sets the start position of the range.

C#

```
public double NewSelectionStart { get; set; }
```

Property Value

Double

A `double` value that represents the start position of a selected range.

Remarks

The NewSelectionStart property represents the new position of a cursor, and can be used in the CursorPositionChanging and CursorPositionChanged events to get the cursor position. It can also be used in the CursorPositionChanging event to set the cursor position.

Note that setting this property in the CursorPositionChanging event will have no effect; the NewSelectionStart property will have a value of NaN in this event.

The start position of a selected range is not necessarily less than the end position of a selected range. The start position is the position where the user fist clicked within the chart area, and the end position is the position of the mouse at the time the event is raised.

To determine the chart area that the event is being raised for, use the ChartArea property.

Use the Axis property to determine if this end position is for a range selected along the primary or secondary X or Y axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomizeLegendEventArgs Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides data for the CustomizeLegend event of the Chart control.

C#

```
public class CustomizeLegendEventArgs : EventArgs
```

Inheritance Object → EventArgs → CustomizeLegendEventArgs

Remarks

The CustomizeLegendEventArgs class is used in the CustomizeLegend event.

There are only two properties in this class. The LegendItems property allows you to delete or modify both default and custom legend items, and to add custom legend items. This property can also be used to change the order of legend items.

The LegendName property is used to identify the legend that raised this event.

Constructors

CustomizeLegendEventArgs(LegendItemsCollection)	Initializes a new instance of the CustomizeLegendEventArgs class with the specified LegendItemsCollection.
CustomizeLegendEventArgs(LegendItemsCollection, String)	Initializes a new instance of the CustomizeLegendEventArgs class with the specified LegendItemsCollection and legend name.

Properties

LegendItems	Used to remove, modify or change the order of both custom and default legend items in the CustomizeLegend event. Can also be used to add custom legend items.

LegendName	Gets the name of the Legend object, which can be used to determine which legend was the sender of the CustomizeLegend event.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomizeLegendEventArgs Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the CustomizeLegendEventArgs class.

Overloads

CustomizeLegendEventArgs(LegendItemsCollection)	Initializes a new instance of the CustomizeLegendEventArgs class with the specified LegendItemsCollection.
CustomizeLegendEventArgs(LegendItemsCollection, String)	Initializes a new instance of the CustomizeLegendEventArgs class with the specified LegendItemsCollection and legend name.

CustomizeLegendEventArgs(LegendItemsCollection)

Initializes a new instance of the CustomizeLegendEventArgs class with the specified LegendItemsCollection.

C#

```
public CustomizeLegendEventArgs
(System.Windows.Forms.DataVisualization.Charting.LegendItemsCollection
legendItems);
```

Parameters

`legendItems` LegendItemsCollection

The LegendItemsCollection.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomizeLegendEventArgs(LegendItemsCollection, String)

Initializes a new instance of the CustomizeLegendEventArgs class with the specified LegendItemsCollection and legend name.

C#

```
public CustomizeLegendEventArgs
(System.Windows.Forms.DataVisualization.Charting.LegendItemsCollection
legendItems, string legendName);
```

Parameters

`legendItems` LegendItemsCollection

The LegendItemsCollection.

`legendName` String

The legend name.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomizeLegendEventArgs.LegendItems Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Used to remove, modify or change the order of both custom and default legend items in the CustomizeLegend event. Can also be used to add custom legend items.

C#

```
public System.Windows.Forms.DataVisualization.Charting.LegendItemsCollection
LegendItems { get; }
```

Property Value

LegendItemsCollection

A LegendItemsCollection object.

Remarks

The LegendItems property allows you to delete or modify both default and custom legend items in the CustomizeLegend event, and to add custom legend items. The order of legend items can also be changed.

> **Note**
>
> The **SeriesPointIndex** and **SeriesName** properties can be used to determine the data element that is associated with a default legend item. These properties will return zero-length strings for custom legend items, because custom items are not associated with chart data.

This property enables you to determine which Legend the event was raised for, which enables you to customize an individual Legend.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomizeLegendEventArgs.Legend Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the name of the Legend object, which can be used to determine which legend was the sender of the CustomizeLegend event.

C#

```
public string LegendName { get; }
```

Property Value

String

A `string` value that represents the name of the associated Legend object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a custom label for an axis.

C#

```
public class CustomLabel :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → CustomLabel

Remarks

CustomLabel objects are stored in the CustomLabelsCollection class, which is exposed as the CustomLabels property of the Axis class.

Axis labels for the first label row are automatically generated unless custom labels are specified using the CustomLabels property. If you specify a custom label along the first label row, no axis labels will be automatically generated for the entire first row.

Only custom labels may be used for the second label row.

The ToPosition and FromPosition properties determine the axis range that the label applies to.

A grid line and/or tick mark can be displayed for a custom axis label by using the GridTicks property; this property specifies whether a custom grid line or tick mark will be drawn in the middle of the label range.

Constructors

CustomLabel()	Creates a new instance of the CustomLabel class.
CustomLabel(Double, Double, String, Int32, LabelMarkStyle)	Initializes a new instance of the CustomLabel class with the specified `fromPosition`, `toPosition`, `text`, `labelRow`, and

	`markStyle` parameters.
CustomLabel(Double, Double, String, Int32, LabelMarkStyle, GridTickTypes)	Initializes a new instance of the CustomLabel class with the specified `fromPosition`, `toPosition`, `text`, `labelRow`, `markStyle` and `gridTick` parameters.

Properties

Axis	Gets the axis that this CustomLabel object belongs to.
ForeColor	Gets or sets the text color of the custom label.
FromPosition	Gets or sets the beginning position, in axis coordinates, of the custom label.
GridTicks	Gets or sets a property that specifies whether custom tick marks and grid lines will be drawn in the center of the label.
Image	Gets or sets the custom label image.
ImageTransparentColor	Gets or sets a color that will be replaced with a transparent color when a custom label image is drawn.
LabelMark	Gets or sets the label mark for a custom label. This applies to labels in rows other than the first row.
MarkColor	Gets or sets the marker color for the custom label.
Name	Gets or sets the name of the custom label.
RowIndex	Gets or sets the index of the custom label row.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the custom label text.
ToolTip	Gets or sets the custom label tooltip text.
ToPosition	Gets or sets the starting position of the custom label, in axis coordinates.

Methods

Clone()	Returns a cloned label object.
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)

Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources. (Inherited from ChartElement)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Creates a new instance of the CustomLabel class.

Overloads

CustomLabel()	Creates a new instance of the CustomLabel class.
CustomLabel(Double, Double, String, Int32, LabelMarkStyle)	Initializes a new instance of the CustomLabel class with the specified `fromPosition`, `toPosition`, `text`, `labelRow`, and `markStyle` parameters.
CustomLabel(Double, Double, String, Int32, LabelMarkStyle, GridTickTypes)	Initializes a new instance of the CustomLabel class with the specified `fromPosition`, `toPosition`, `text`, `labelRow`, `markStyle` and `gridTick` parameters.

CustomLabel()

Creates a new instance of the CustomLabel class.

C#

```
public CustomLabel ();
```

Remarks

This constructor can be used to create and initialize a new instance of the CustomLabel class.

However, it is recommended that you use one of the `Add` methods in the CustomLabelsCollection class to add custom labels.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel(Double, Double, String, Int32, LabelMarkStyle)

Initializes a new instance of the CustomLabel class with the specified `fromPosition`, `toPosition`, `text`, `labelRow`, and `markStyle` parameters.

```
C#

public CustomLabel (double fromPosition, double toPosition, string text,
int labelRow,
System.Windows.Forms.DataVisualization.Charting.LabelMarkStyle
markStyle);
```

Parameters

`fromPosition` Double

The beginning of the axis range the label applies to.

`toPosition` Double

The end of the axis range the label applies to.

`text` String

The label text.

`labelRow` Int32

The label row index.

`markStyle` LabelMarkStyle

The label mark style.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel(Double, Double, String, Int32, LabelMarkStyle, GridTickTypes)

Initializes a new instance of the CustomLabel class with the specified `fromPosition`, `toPosition`, `text`, `labelRow`, `markStyle` and `gridTick` parameters.

```
C#

public CustomLabel (double fromPosition, double toPosition, string text,
int labelRow,
System.Windows.Forms.DataVisualization.Charting.LabelMarkStyle markStyle,
System.Windows.Forms.DataVisualization.Charting.GridTickTypes gridTick);
```

Parameters

`fromPosition` Double

The beginning of the axis range the label applies to.

`toPosition` Double

The end of the axis range the label applies to.

`text` String

The label text.

`labelRow` Int32

The label row index.

`markStyle` LabelMarkStyle

The label mark style.

`gridTick` GridTickTypes

The custom grid line and/or tick marks flag.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.Axis Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the axis that this CustomLabel object belongs to.

C#

```
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.Axis Axis { get; }
```

Property Value

Axis

An Axis object that represents the axis that this CustomLabel object belongs to.

Attributes BrowsableAttribute

Remarks

Use this method to determine which Axis object the CustomLabel object belongs to.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color of the custom label.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value that represents the text color of a custom label.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The ForeColor property determines the text color for a custom label.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.FromPosition Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the beginning position, in axis coordinates, of the custom label.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisLabelDateValueConverter))]
public double FromPosition { get; set; }
```

Property Value

Double

A `double` value that represents the beginning of the axis range that the label is applied to.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The FromPosition property determines the start position of the axis range the label is applied to.

The difference between the values of the ToPosition and FromPosition properties determines where the label text will be displayed. For example, suppose the X-axis has a scale that ranges from 1 to 5 (1, 2 ,3 , 4, 5). In order to display a custom label just underneath the second point, you would set the ToPosition property to 1.5 and the FromPosition property to 2.5.

If the XValueType - or YValueType, depending on the axis in question - is set to a DateTime value, then this FromPosition property will be displayed as a DateTime value in the designer.

This property should not be set to the same value as the ToPosition property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.GridTicks Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a property that specifies whether custom tick marks and grid lines will be drawn in the center of the label.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GridTickTypes
GridTicks { get; set; }
```

Property Value

GridTickTypes

A GridTickTypes enumeration value that specifies whether custom tick marks and grid lines will be drawn in the center of the label.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.Image Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the custom label image.

C#

```
[System.ComponentModel.Bindable(true)]
public string Image { get; set; }
```

Property Value

String

A `string` value that represents the name of the custom label image.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.ImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color that will be replaced with a transparent color when a custom label image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ImageTransparentColor { get; set; }
```

Property Value

Color

A Color value which is replaced with a transparent color when a custom label image is drawn. The default value is Magenta.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.LabelMark Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the label mark for a custom label. This applies to labels in rows other than the first row.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.LabelMarkStyle
LabelMark { get; set; }
```

Property Value

LabelMarkStyle

A LabelMarkStyle enumeration value that determines the type of label mark, if any, to be used.

Attributes BindableAttribute

Remarks

Label marks can only be used for the second row of labels, and are usually used to group labels together from the first row.

The major tick mark's line style of the relevant Axis object is used for the line pattern of the mark, for example, the AxisX.MajorTickMark.LineDashStyle property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.MarkColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the marker color for the custom label.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color MarkColor { get; set; }
```

Property Value

Color

A Color value that represents the marker color of a custom label.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The MarkColor property determines the color of a custom label's marker.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the custom label.

C#

```
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the text name of the custom label.

Remarks

A string value that represents the text name of the item.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.RowIndex Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the index of the custom label row.

C#

```
[System.ComponentModel.Bindable(true)]
public int RowIndex { get; set; }
```

Property Value

Int32

An `integer` value that specifies the index, 0 to 10, of the custom label row.

Attributes BindableAttribute

Remarks

The RowIndex property specifies the index of the row label where zero is the row label closest to the axis to ten, the further most axis label.

Only custom labels can be used for the label rows beyond the first row.

If a custom label is used for the first row, labels for that entire axis will not be generated automatically; in this case, all labels for the first row must then be custom labels.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.Text Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the custom label text.

C#

```
[System.ComponentModel.Bindable(true)]
public string Text { get; set; }
```

Property Value

String

A `string` value that represents the label text.

Attributes BindableAttribute

Remarks

The Text property gets or sets the label text of the CustomLabel object.

Note that axis labels for the first label row are generated automatically, but can be overridden using CustomLabel objects. If you specify a custom label along the first axis row, no axis labels will be automatically generated for the entire first row.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the custom label tooltip text.

C#

```
[System.ComponentModel.Bindable(true)]
public string ToolTip { get; set; }
```

Property Value

String

A `string` that represents the tooltip text.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.ToPosition Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the starting position of the custom label, in axis coordinates.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisLabelDateValueConverter))]
public double ToPosition { get; set; }
```

Property Value

Double

A `double` value that represents the starting position.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabel.Clone Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns a cloned label object.

C#

```
public System.Windows.Forms.DataVisualization.Charting.CustomLabel Clone ();
```

Returns

CustomLabel

A CustomLabel object that represents a copy of the current custom label.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabelsCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of CustomLabel objects.

C#

```
public class CustomLabelsCollection :
System.Windows.Forms.DataVisualization.Charting.ChartElementCollection<Syste
m.Windows.Forms.DataVisualization.Charting.CustomLabel>
```

Inheritance Object → Collection<T> → ChartElementCollection<CustomLabel> → CustomLabelsCollection

Remarks

The CustomLabelsCollection class represents a collection of CustomLabel objects, which in turn represent the custom labels that an Axis object can use. This class is exposed as the CustomLabels collection property of an Axis object.

> **Note**
>
> You can automatically populate the entire label row of an axis with custom **DateTime** labels using any **Add** method definition that includes a `DateTimeIntervalType` argument.

The Count property indicates the total number of items in the collection and is commonly used to find the upper bound of the collection.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)

Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)

Methods

Add(Double, DateTimeIntervalType)	Adds multiple CustomLabel objects of type DateTime to the collection. Populates the entire first row of an axis with labels.
Add(Double, DateTimeIntervalType, Double, Double, String, Int32, LabelMarkStyle)	Adds multiple custom labels to the collection. The labels will be DateTime labels with the specified interval type, and will be generated for the axis range that is determined by the `min` and `max` arguments.
Add(Double, DateTimeIntervalType, String)	Adds multiple DateTimeCustomLabelsCollection objects to the collection. Populates the entire first row of an axis with labels with the specified format.
Add(Double, DateTimeIntervalType, String, Int32, LabelMarkStyle)	Adds multiple custom labels to the collection. The labels will be DateTime labels with the specified interval type, text format, row index and marking style.
Add(Double, Double, String)	Adds a CustomLabel object to the collection, using the given axis range and text.
Add(Double, Double, String, Int32, LabelMarkStyle)	Adds a custom label to the collection, using the given axis range, text, row index and marking style.
Add(Double, Double, String, Int32, LabelMarkStyle, GridTickTypes)	Adds a custom label to the collection, using the given axis range, text, row index, marking style and grid tick type.
Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array.

	(Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts an element into the Collection<T> at the specified index location. (Inherited from ChartElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the element at the specified index location of the Collection<T>. (Inherited from ChartElementCollection<T>)

ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the element at the specified index location. (Inherited from ChartElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)

IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.

Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the

	specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.

First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>,	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its

Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>,	Returns an Int64 that represents how many

Func<TSource,Boolean>)	elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.

Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>,	Sorts the elements of a sequence in ascending order by using a specified comparer.

IComparer<TKey>)	
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource>	Determines whether two sequences are equal

(IEnumerable<TSource>, IEnumerable<TSource>, IEqualityComparer<TSource>)	by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.

Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.

ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.

Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.

Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomLabelsCollection.Add Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a CustomLabel object to the collection.

Overloads

Add(Double, DateTimeIntervalType)	Adds multiple CustomLabel objects of type DateTime to the collection. Populates the entire first row of an axis with labels.
Add(Double, Double, String)	Adds a CustomLabel object to the collection, using the given axis range and text.
Add(Double, DateTimeIntervalType, String)	Adds multiple DateTimeCustomLabelsCollection objects to the collection. Populates the entire first row of an axis with labels with the specified format.
Add(Double, Double, String, Int32, LabelMarkStyle)	Adds a custom label to the collection, using the given axis range, text, row index and marking style.
Add(Double, DateTimeIntervalType, String, Int32, LabelMarkStyle)	Adds multiple custom labels to the collection. The labels will be DateTime labels with the specified interval type, text format, row index and marking style.
Add(Double, Double, String, Int32, LabelMarkStyle, GridTickTypes)	Adds a custom label to the collection, using the given axis range, text, row index, marking style and grid tick type.
Add(Double, DateTimeIntervalType, Double, Double, String, Int32, LabelMarkStyle)	Adds multiple custom labels to the collection. The labels will be DateTime labels with the specified interval type, and will be generated for the axis range that is determined by the `min` and `max` arguments.

Add(Double, DateTimeIntervalType)

Adds multiple CustomLabel objects of type DateTime to the collection. Populates the entire first row of an axis with labels.

```
C#

public void Add (double labelsStep,
System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
intervalType);
```

Parameters

`labelsStep` Double

A Double value that determines how often the custom labels will be drawn.

`intervalType` DateTimeIntervalType

A DateTimeIntervalType enumeration value that determines the interval type.

Remarks

The Add method creates and initializes CustomLabel objects for the entire first label row of an axis with DateTime labels of the specified type.

The `labelsStep` parameter determines how often the custom labels will be drawn.

> ⓘ **Note**
>
> The **RecalculateAxesScale** method must be called prior to calling this particular method definition.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Add(Double, Double, String)

Adds a CustomLabel object to the collection, using the given axis range and text.

```
C#
```

```
public System.Windows.Forms.DataVisualization.Charting.CustomLabel Add
(double fromPosition, double toPosition, string text);
```

Parameters

`fromPosition` Double

The minimum value of the axis range to which the label is applied.

`toPosition` Double

The maximum value of the axis range to which the label is applied.

`text` String

The label text.

Returns

CustomLabel

The newly added CustomLabel object.

Remarks

Adds one custom label to the first label row of the relevant axis. No label mark is used.

If a custom label is used anywhere in the first label row, no labels are automatically generated for that first row.

> ⓘ **Note**
>
> The **RecalculateAxesScale** method must be called prior to calling this particular method definition.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Add(Double, DateTimeIntervalType, String)

Adds multiple DateTimeCustomLabelsCollection objects to the collection. Populates the entire first row of an axis with labels with the specified format.

```
C#

public void Add (double labelsStep,
System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
intervalType, string format);
```

Parameters

`labelsStep` Double

A Double value that determines how often custom labels will be drawn.

`intervalType` DateTimeIntervalType

A DateTimeIntervalType enumeration value that determines the unit of measure for the `labelsStep` parameter.

`format` String

A value that determines the format of the custom label.

Remarks

This method creates and initializes CustomLabel objects for an axis' entire first label row with DateTime labels of the specified type.

The step determines how often the custom labels will be drawn, and optional label formatting can be specified as well.

> ⓘ **Note**
>
> The **RecalculateAxesScale** method must be called prior to calling this particular method definition.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Add(Double, Double, String, Int32, LabelMarkStyle)

Adds a custom label to the collection, using the given axis range, text, row index and marking style.

C#

```
public System.Windows.Forms.DataVisualization.Charting.CustomLabel Add
(double fromPosition, double toPosition, string text, int rowIndex,
System.Windows.Forms.DataVisualization.Charting.LabelMarkStyle
markStyle);
```

Parameters

`fromPosition` Double

The minimum value of the axis range the label is applied to.

`toPosition` Double

The maximum value of the axis range the label is applied to.

`text` String

The label text.

`rowIndex` Int32

The label row index.

`markStyle` LabelMarkStyle

A LabelMarkStyle enumeration value that specifies mark style for the label.

Returns

CustomLabel

The newly added CustomLabel object.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Add(Double, DateTimeIntervalType, String, Int32, LabelMarkStyle)

Adds multiple custom labels to the collection. The labels will be DateTime labels with the specified interval type, text format, row index and marking style.

C#

```
public void Add (double labelsStep,
System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
intervalType, string format, int rowIndex,
System.Windows.Forms.DataVisualization.Charting.LabelMarkStyle
markStyle);
```

Parameters

`labelsStep` Double

A Double value that determines how often custom labels will be drawn.

`intervalType` DateTimeIntervalType

A DateTimeIntervalType enumeration value that determines the unit of measure for the `labelsStep` parameter.

`format` String

The label text format.

`rowIndex` Int32

The label row index.

`markStyle` LabelMarkStyle

A LabelMarkStyle enumeration value that specifies mark style for the label.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Add(Double, Double, String, Int32, LabelMarkStyle, GridTickTypes)

Adds a custom label to the collection, using the given axis range, text, row index, marking style and grid tick type.

C#

```
public System.Windows.Forms.DataVisualization.Charting.CustomLabel Add
(double fromPosition, double toPosition, string text, int rowIndex,
System.Windows.Forms.DataVisualization.Charting.LabelMarkStyle markStyle,
System.Windows.Forms.DataVisualization.Charting.GridTickTypes gridTick);
```

Parameters

`fromPosition` Double

The minimum value of the axis range the label is applied to.

`toPosition` Double

The maximum value of the axis range the label is applied to.

`text` String

The label text.

`rowIndex` Int32

The label row index.

`markStyle` LabelMarkStyle

A LabelMarkStyle enumeration value that specifies mark style for the label.

`gridTick` GridTickTypes

A GridTickTypes enumeration value that specifies the grid tick for the label.

Returns

CustomLabel

The newly added CustomLabel object.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Add(Double, DateTimeIntervalType, Double, Double, String, Int32, LabelMarkStyle)

Adds multiple custom labels to the collection. The labels will be DateTime labels with the specified interval type, and will be generated for the axis range that is determined by the `min` and `max` arguments.

C#

```
public void Add (double labelsStep,
System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
intervalType, double min, double max, string format, int rowIndex,
System.Windows.Forms.DataVisualization.Charting.LabelMarkStyle
markStyle);
```

Parameters

`labelsStep` Double

A Double value that determines how often custom labels will be drawn.

`intervalType` DateTimeIntervalType

A DateTimeIntervalType enumeration value that determines the unit of measure for the `labelsStep` parameter.

`min` Double

The minimum value for the axis range.

`max` Double

The maximum value for the axis range.

`format` String

The label text format.

`rowIndex` Int32

The label row index.

`markStyle` LabelMarkStyle

A LabelMarkStyle enumeration value that specifies mark style for the label.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

CustomProperties Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the custom properties for the data visualization charting.

This API supports the product infrastructure and is not intended to be used directly from your code.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.CustomPropertiesTypeConverter))]
public class CustomProperties
```

Inheritance Object → CustomProperties

Attributes TypeConverterAttribute

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataFormula Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides properties and methods to prepare series data for technical analysis, apply formulas on the series data and prepare output data to be displayed as a chart. The actual calculations are performed in the StatisticFormula class.

C#

```
public class DataFormula
```

Inheritance Object → DataFormula

Derived System.Windows.Forms.DataVisualization.Charting.DataManipulator

Constructors

DataFormula()	Initializes a new instance of the DataFormula class.

Properties

IsEmptyPointIgnored	Gets or sets a flag that indicates whether empty points are ignored while performing calculations.
IsStartFromFirst	Gets or sets a flag that indicates whether to start formulas such as "rolling average" from zero.
Statistics	Gets a reference to the StatisticFormula class.

Methods

CopySeriesValues(String, String)	Copies a series or the specific X or Y value.

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FinancialFormula(Financial Formula, Series)	Calls a method from a formula module with the specified formula name and input series.
FinancialFormula(Financial Formula, Series, Series)	Calls a method from a formula module with the specified formula name, input and output series.
FinancialFormula(Financial Formula, String)	Calls a method from a formula module with the specified formula name and input series.
FinancialFormula(Financial Formula, String, Series, Series)	Calls a method from a formula module with the specified formula name and parameters, using the specified input and output series.
FinancialFormula(Financial Formula, String, String)	Calls a method from a formula module with the specified formula name, and the names of the input and output series.
FinancialFormula(Financial Formula, String, String, String)	Calls a method from a formula module with the specified formula name and parameters, and the names of the input and output series.
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataFormula Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the DataFormula class.

C#

```
public DataFormula ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataFormula.IsEmptyPointIgnored Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether empty points are ignored while performing calculations.

C#

```
public bool IsEmptyPointIgnored { get; set; }
```

Property Value

Boolean

`true` if empty points are ignored while performing calculations; `false` if empty points are treated as zeroes.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataFormula.IsStartFromFirst Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether to start formulas such as "rolling average" from zero.

```
C#
public bool IsStartFromFirst { get; set; }
```

Property Value

Boolean

`true` if formulas such as "rolling average" start from zero; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataFormula.Statistics Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a reference to the StatisticFormula class.

C#

```
public System.Windows.Forms.DataVisualization.Charting.StatisticFormula
Statistics { get; }
```

Property Value

StatisticFormula

A reference to the StatisticFormula object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataFormula.CopySeriesValues(String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Copies a series or the specific X or Y value.

C#

```
public void CopySeriesValues (string inputSeries, string outputSeries);
```

Parameters

`inputSeries` String

A comma-separated list of input data series names and optional X and Y value names.

`outputSeries` String

A comma-separated list of output data series names and optional X and Y value names.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataFormula.FinancialFormula Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Calls a method from a formula.

Overloads

FinancialFormula(FinancialFormula, String)	Calls a method from a formula module with the specified formula name and input series.
FinancialFormula(FinancialFormula, Series)	Calls a method from a formula module with the specified formula name and input series.
FinancialFormula(FinancialFormula, String, String)	Calls a method from a formula module with the specified formula name, and the names of the input and output series.
FinancialFormula(FinancialFormula, Series, Series)	Calls a method from a formula module with the specified formula name, input and output series.
FinancialFormula(FinancialFormula, String, String, String)	Calls a method from a formula module with the specified formula name and parameters, and the names of the input and output series.
FinancialFormula(FinancialFormula, String, Series, Series)	Calls a method from a formula module with the specified formula name and parameters, using the specified input and output series.

FinancialFormula(FinancialFormula, String)

Calls a method from a formula module with the specified formula name and input series.

C#

```
public void FinancialFormula
(System.Windows.Forms.DataVisualization.Charting.FinancialFormula
formulaName, string inputSeries);
```

Parameters

`formulaName` FinancialFormula

The name of the formula to call.

`inputSeries` String

The input series in which calculation is performed.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FinancialFormula(FinancialFormula, Series)

Calls a method from a formula module with the specified formula name and input series.

C#

```
public void FinancialFormula
(System.Windows.Forms.DataVisualization.Charting.FinancialFormula
formulaName, System.Windows.Forms.DataVisualization.Charting.Series
inputSeries);
```

Parameters

`formulaName` FinancialFormula

The name of the formula to call.

`inputSeries` Series

A comma-separated list of input series names and optional X and Y value names.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FinancialFormula(FinancialFormula, String, String)

Calls a method from a formula module with the specified formula name, and the names of the input and output series.

C#

```
public void FinancialFormula
(System.Windows.Forms.DataVisualization.Charting.FinancialFormula
formulaName, string inputSeries, string outputSeries);
```

Parameters

`formulaName` FinancialFormula

The name of the formula to call.

`inputSeries` String

The input series in which calculation is performed.

`outputSeries` String

The output series in which the result of the calculation is stored.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FinancialFormula(FinancialFormula, Series, Series)

Calls a method from a formula module with the specified formula name, input and

output series.

```
C#

public void FinancialFormula
(System.Windows.Forms.DataVisualization.Charting.FinancialFormula
formulaName, System.Windows.Forms.DataVisualization.Charting.Series
inputSeries, System.Windows.Forms.DataVisualization.Charting.Series
outputSeries);
```

Parameters

`formulaName` FinancialFormula

The name of the formula to call.

`inputSeries` Series

A comma-separated list of input series names and optional X and Y value names.

`outputSeries` Series

A comma-separated list of output series names and optional X and Y value names.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FinancialFormula(FinancialFormula, String, String, String)

Calls a method from a formula module with the specified formula name and parameters, and the names of the input and output series.

```
C#

public void FinancialFormula
(System.Windows.Forms.DataVisualization.Charting.FinancialFormula
formulaName, string parameters, string inputSeries, string outputSeries);
```

Parameters

`formulaName` FinancialFormula

The name of the formula to call.

`parameters` String

The parameters associated with the formula.

`inputSeries` String

The input series in which calculation is performed.

`outputSeries` String

The output series in which the result of the calculation is stored.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FinancialFormula(FinancialFormula, String, Series, Series)

Calls a method from a formula module with the specified formula name and parameters, using the specified input and output series.

C#

```
public void FinancialFormula
(System.Windows.Forms.DataVisualization.Charting.FinancialFormula
formulaName, string parameters,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
System.Windows.Forms.DataVisualization.Charting.Series outputSeries);
```

Parameters

`formulaName` FinancialFormula

The name of the formula to call.

`parameters` String

The parameters associated with the formula.

`inputSeries` Series

A comma-separated list of input series names and optional X and Y value names.

`outputSeries` Series

A comma-separated list of output series names and optional X and Y value names.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides method that performs data manipulation operations at run time. This class is exposed by the DataManipulator property of the root Chart object.

C#

```
public class DataManipulator :
System.Windows.Forms.DataVisualization.Charting.DataFormula
```

Inheritance Object → DataFormula → DataManipulator

Constructors

DataManipulator()	Initializes a new instance of the DataManipulator class.

Properties

FilterMatchedPoints	Gets or sets a value that determines whether points are filtered if they match criteria specified in arguments in the `Filter` methods of this class.
FilterSetEmptyPoints	Gets or sets a flag that indicates whether points filtered by the `Filter` or `FilterTopN` methods will be removed or marked as empty.
IsEmptyPointIgnored	Gets or sets a flag that indicates whether empty points are ignored while performing calculations. (Inherited from DataFormula)
IsStartFromFirst	Gets or sets a flag that indicates whether to start formulas such as "rolling average" from zero. (Inherited from DataFormula)
Statistics	Gets a reference to the StatisticFormula class. (Inherited from DataFormula)

Methods

CopySeriesValues(String, String)	Copies a series or the specific X or Y value. (Inherited from DataFormula)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
ExportSeriesValues()	Exports the data for every series stored in the SeriesCollection to the DataSet object.
ExportSeriesValues(Series)	Exports the data from the specified series to the DataSet object.
ExportSeriesValues(String)	Exports the data from one or more series to the DataSet object.
Filter(CompareMethod, Double, Series)	Filters the data points in a series by applying a filtering rule to the first Y-value of each data point. The Series object that is filtered also stores the modified data.
Filter(CompareMethod, Double, Series, Series)	Filters the data points in a series by applying a filtering rule to the first Y-value of each of the data points.
Filter(CompareMethod, Double, Series, Series, String)	Filters the data points in a series by applying a filtering rule to the specified data point values for comparison.
Filter(CompareMethod, Double, String)	Filters one or more series by applying a filtering rule to the first Y-value of the data points in the first series. The filtered Series objects are used to store the modified data.
Filter(CompareMethod, Double, String, String)	Filters one or more series by applying a filtering rule to the first Y-value of each of the data points in the first series.
Filter(CompareMethod, Double, String, String, String)	Filters one or more series by applying a filtering rule to the specified data point values in the first series.
Filter(DateRangeType, String, Series)	Filters the data points in a series, either by removing the specified points or by marking them as empty for the specified date/time ranges. The Series object that is filtered is used to store the modified data.
Filter(DateRangeType, String, Series, Series)	Filters the data points in a series, either by removing the specified points or by marking them as empty for the specified date/time ranges.
Filter(DateRangeType, String, String)	Filters the data points of one or more series, either by removing the specified points or by marking them as empty for the given date/time ranges. The filtered Series objects are used to store the modified data.

Filter(DateRangeType, String, String, String)	Filters the data points of one or more series, either by removing the specified points or by marking them as empty for the given date/time ranges.
Filter(IDataPointFilter, Series)	Performs custom filtering on the data points in a series. The Series object that is filtered also stores the modified data.
Filter(IDataPointFilter, Series, Series)	Performs custom filtering on the data points in a series.
Filter(IDataPointFilter, String)	Performs custom filtering on the data points in one or more series, based on the points of the first series. The filtered Series objects also store the modified data.
Filter(IDataPointFilter, String, String)	Performs custom filtering on the data points in one or more series, based on the points of the first series.
FilterTopN(Int32, Series)	Filters all data points in a series except for a specified number of points with the largest first Y-values. The Series object that is filtered is used to store the modified data.
FilterTopN(Int32, Series, Series)	Filters all data points in a series except for a specified number of points with the largest first Y-values.
FilterTopN(Int32, Series, Series, String)	Filters all data points in a series except for a specified number of points with the largest values.
FilterTopN(Int32, Series, Series, String, Boolean)	Filters all data points in a series except for a specified number of points with the smallest or largest values.
FilterTopN(Int32, String)	Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first series that have the largest first Y-values. The Series objects that are filtered are used to store the modified data.
FilterTopN(Int32, String, String)	Filters all data points in one or more series except for a specified number of points. The points that aren't filtered correspond to points in the first series that have the largest first Y-values.
FilterTopN(Int32, String, String, String)	Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first series that have the largest values.
FilterTopN(Int32, String, String, String, Boolean)	Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first input series that have the largest or smallest values.
FinancialFormula(Financial Formula, Series)	Calls a method from a formula module with the specified formula name and input series. (Inherited from DataFormula)

FinancialFormula(FinancialFormula, Series, Series)	Calls a method from a formula module with the specified formula name, input and output series. (Inherited from DataFormula)
FinancialFormula(FinancialFormula, String)	Calls a method from a formula module with the specified formula name and input series. (Inherited from DataFormula)
FinancialFormula(FinancialFormula, String, Series, Series)	Calls a method from a formula module with the specified formula name and parameters, using the specified input and output series. (Inherited from DataFormula)
FinancialFormula(FinancialFormula, String, String)	Calls a method from a formula module with the specified formula name, and the names of the input and output series. (Inherited from DataFormula)
FinancialFormula(FinancialFormula, String, String, String)	Calls a method from a formula module with the specified formula name and parameters, and the names of the input and output series. (Inherited from DataFormula)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
Group(String, Double, IntervalType, Double, IntervalType, Series)	Groups data points in a series using one or more formulas. The series that is grouped is cleared of its original data, and used to store the new data points.
Group(String, Double, IntervalType, Double, IntervalType, Series, Series)	Groups data points in a series using one or more formulas. An output series is used to store the grouped data points, and an offset can be used for intervals.
Group(String, Double, IntervalType, Double, IntervalType, String)	Groups data points for one or more series, using one or more formulas. Series are cleared of their original data and used to store the new data points.
Group(String, Double, IntervalType, Double, IntervalType, String, String)	Groups data points for one or more series, using one or more formulas. Output series are used to store the grouped data points, and an offset can be used for intervals.
Group(String, Double, IntervalType, Series)	Groups data points in a series using one or more formulas. The series that is grouped is cleared of its original data, and used to store the new data points.
Group(String, Double, IntervalType, Series, Series)	Groups data points in a series using one or more formulas. An output series is used to store the grouped data points.

Group(String, Double, IntervalType, String)	Groups data points for one or more series, using one or more formulas. Series are cleared of their original data and used to store the new data points.
Group(String, Double, IntervalType, String, String)	Groups data points for one or more series, using one or more formulas. Output series are used to store the grouped data points.
GroupByAxisLabel(String, Series)	Groups data points in a series by axis labels, using one or more formulas. Original data point values are cleared, and the series then stores the new grouped data points.
GroupByAxisLabel(String, Series, Series)	Groups data points in a series by axis labels, using one or more formulas. An output series is used to store the grouped data points.
GroupByAxisLabel(String, String)	Groups data points in one or more series by axis labels, using one or more formulas. Original data point values are cleared, and each of the series then stores its new grouped data points.
GroupByAxisLabel(String, String, String)	Groups data points in one or more series by axis labels, using one or more formulas. Output series are used to store the grouped data points.
InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Double, Double, Series)	Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments. The series range that is checked is determined by the `toXValue` and `fromXValue` arguments.
InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Double, Double, String)	Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments. The series range that is checked is determined by the `toXValue` and `fromXValue` arguments.
InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Series)	Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments.
InsertEmptyPoints(Double, IntervalType, Double, IntervalType, String)	Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments.
InsertEmptyPoints(Double, IntervalType, Series)	Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval` and `intervalType` arguments.

InsertEmptyPoints(Double, IntervalType, String)	Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval` and `intervalType` arguments.
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Sort(IComparer<DataPoint>, Series)	Performs a custom sort on the data points in a series.
Sort(IComparer<DataPoint>, String)	Performs a custom sort on the data points in one or more series.
Sort(PointSortOrder, Series)	Performs a sort on the data points in a series using the specified order. The sort is based on the first Y-value of each of the data points.
Sort(PointSortOrder, String)	Performs a sort on the data points in one or more series, using the specified order. The sort is based on the first Y-value of each of the data points.
Sort(PointSortOrder, String, Series)	Performs a sort on the data points in a series using the specified order. The sort can be based on any available data point value.
Sort(PointSortOrder, String, String)	Performs a sort on the data points in one or more series, using the specified order. The sort can be based on any available data point value.
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the DataManipulator class.

C#

```
public DataManipulator ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.FilterMatchedPoints Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that determines whether points are filtered if they match criteria specified in arguments in the `Filter` methods of this class.

C#

```
public bool FilterMatchedPoints { get; set; }
```

Property Value

Boolean

`true` if points that match specified criteria are filtered; `false` if points that do not match the criteria are filtered. The default value is `true`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.FilterSetEmptyPoints Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether points filtered by the `Filter` or `FilterTopN` methods will be removed or marked as empty.

C#

```
public bool FilterSetEmptyPoints { get; set; }
```

Property Value

Boolean

`true` if filtered points will be marked as empty; `false` if they will be removed. The default value is `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.ExportSeriesValues Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Exports the data from the a series or collection of series to the DataSet object.

Overloads

ExportSeriesValues()	Exports the data for every series stored in the SeriesCollection to the DataSet object.
ExportSeriesValues(String)	Exports the data from one or more series to the DataSet object.
ExportSeriesValues(Series)	Exports the data from the specified series to the DataSet object.

ExportSeriesValues()

Exports the data for every series stored in the SeriesCollection to the DataSet object.

C#

```
public System.Data.DataSet ExportSeriesValues ();
```

Returns

DataSet

A DataSet object that contains the exported series data.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ExportSeriesValues(String)

Exports the data from one or more series to the DataSet object.

C#

```
public System.Data.DataSet ExportSeriesValues (string seriesNames);
```

Parameters

`seriesNames` String

A comma-separated list of names of the Series to be exported.

Returns

DataSet

A DataSet object that contains the exported series data.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ExportSeriesValues(Series)

Exports the data from the specified series to the DataSet object.

C#

```
public System.Data.DataSet ExportSeriesValues
(System.Windows.Forms.DataVisualization.Charting.Series series);
```

Parameters

`series` Series

The series to be exported.

Returns

DataSet

A DataSet object that contains the exported series data.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.Filter Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Filters one or more series, or filters the data point in a series or a collection of series, by applying a standard filtering rule or a custom filtering rule.

Overloads

Filter(CompareMethod, Double, Series, Series, String)	Filters the data points in a series by applying a filtering rule to the specified data point values for comparison.
Filter(CompareMethod, Double, String, String, String)	Filters one or more series by applying a filtering rule to the specified data point values in the first series.
Filter(DateRangeType, String, Series, Series)	Filters the data points in a series, either by removing the specified points or by marking them as empty for the specified date/time ranges.
Filter(DateRangeType, String, String, String)	Filters the data points of one or more series, either by removing the specified points or by marking them as empty for the given date/time ranges.
Filter(CompareMethod, Double, Series, Series)	Filters the data points in a series by applying a filtering rule to the first Y-value of each of the data points.
Filter(CompareMethod, Double, String, String)	Filters one or more series by applying a filtering rule to the first Y-value of each of the data points in the first series.
Filter(IDataPointFilter, Series, Series)	Performs custom filtering on the data points in a series.
Filter(DateRangeType, String, Series)	Filters the data points in a series, either by removing the specified points or by marking them as empty for the specified date/time ranges. The Series object that is filtered is used to store the modified data.
Filter(DateRangeType, String, String)	Filters the data points of one or more series, either by removing the specified points or by marking them as empty for the given date/time ranges. The filtered Series objects are used to store the modified data.

Filter(CompareMethod, Double, Series)	Filters the data points in a series by applying a filtering rule to the first Y-value of each data point. The Series object that is filtered also stores the modified data.
Filter(CompareMethod, Double, String)	Filters one or more series by applying a filtering rule to the first Y-value of the data points in the first series. The filtered Series objects are used to store the modified data.
Filter(IDataPointFilter, Series)	Performs custom filtering on the data points in a series. The Series object that is filtered also stores the modified data.
Filter(IDataPointFilter, String)	Performs custom filtering on the data points in one or more series, based on the points of the first series. The filtered Series objects also store the modified data.
Filter(IDataPointFilter, String, String)	Performs custom filtering on the data points in one or more series, based on the points of the first series.

Filter(CompareMethod, Double, Series, Series, String)

Filters the data points in a series by applying a filtering rule to the specified data point values for comparison.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.CompareMethod
compareMethod, double compareValue,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
System.Windows.Forms.DataVisualization.Charting.Series outputSeries,
string usingValue);
```

Parameters

compareMethod CompareMethod

A CompareMethod enumeration value that determines the comparison method for the filtering rule. Can be LessThan, EqualTo, and so forth.

compareValue Double

The value part of the filtering rule that is applied to the first Y-values of data points. For example, in the filtering rule "<6", 6 is the value.

inputSeries Series

The Series object to be filtered.

`outputSeries` Series

A Series object that stores the filtered data. If the series does not exist, an exception will be thrown.

`usingValue` String

The data point value that the filtering rule is applied to. Can be X, Y, Y2, Y3, and so forth.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(CompareMethod, Double, String, String, String)

Filters one or more series by applying a filtering rule to the specified data point values in the first series.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.CompareMethod
compareMethod, double compareValue, string inputSeriesNames, string
outputSeriesNames, string usingValue);
```

Parameters

`compareMethod` CompareMethod

A CompareMethod enumeration value that determines the comparison method for the filtering rule. Can be LessThan, EqualTo, and so forth.

`compareValue` Double

The value part of the filtering rule that is applied to the first Y-values of data points. For example, in the filtering rule "<6", 6 is the value.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

`outputSeriesNames` String

A comma-separated list of names of the Series that store the filtered data. The number of output series must equal the number of input series, otherwise an exception will be thrown.

`usingValue` String

The data point value that the filtering rule is applied to. Can be X, Y, Y2, Y3, and so forth.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(DateRangeType, String, Series, Series)

Filters the data points in a series, either by removing the specified points or by marking them as empty for the specified date/time ranges.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.DateRangeType dateRange,
string rangeElements,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
System.Windows.Forms.DataVisualization.Charting.Series outputSeries);
```

Parameters

`dateRange` DateRangeType

A DateRangeType enumeration value that represents a date/time interval, such as days of the week, days of the month, and so forth. Determines the range over which the given elements will be filtered.

`rangeElements` String

Specifies the elements within the date/time range designated by the `dateRange` parameter that will be filtered. Can be a single value, for example, "7"; comma-separated values, for example, "5,6"; a range of values, such as "9-11"; or any variation thereof, such as "5,6,9-11".

`inputSeries` Series

The Series object to be filtered.

`outputSeries` Series

A Series object that stores the filtered data. If the series does not exist, an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(DateRangeType, String, String, String)

Filters the data points of one or more series, either by removing the specified points or by marking them as empty for the given date/time ranges.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.DateRangeType dateRange,
string rangeElements, string inputSeriesNames, string outputSeriesNames);
```

Parameters

`dateRange` DateRangeType

A DateRangeType enumeration value that represents a date/time interval, such as days of the week, days of the month, and so forth. Determines the range over which the given elements will be filtered.

`rangeElements` String

Specifies the elements within the date/time range designated by the `dateRange` parameter that will be filtered. Can be a single value, for example, "7"; comma-separated values, for example, "5,6"; a range of values, such as "9-11"; or any variation thereof, such as "5,6,9-11".

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

`outputSeriesNames` String

A comma-separated list of names of the Series that store the filtered data. Make sure that the number of output series matches the number of input series, otherwise an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(CompareMethod, Double, Series, Series)

Filters the data points in a series by applying a filtering rule to the first Y-value of each of the data points.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.CompareMethod
compareMethod, double compareValue,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
System.Windows.Forms.DataVisualization.Charting.Series outputSeries);
```

Parameters

`compareMethod` CompareMethod

A CompareMethod enumeration value that determines the comparison method for the filtering rule. Can be LessThan, EqualTo, and so forth.

`compareValue` Double

The value part of the filtering rule that is applied to the first Y-values of data points. For example, in the filtering rule "<6", 6 is the value.

`inputSeries` Series

The Series object to be filtered.

`outputSeries` Series

A Series object that stores the filtered data. If the series does not exist, an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(CompareMethod, Double, String, String)

Filters one or more series by applying a filtering rule to the first Y-value of each of the data points in the first series.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.CompareMethod
compareMethod, double compareValue, string inputSeriesNames, string
outputSeriesNames);
```

Parameters

`compareMethod` CompareMethod

A CompareMethod enumeration value that determines the comparison method for the filtering rule. Can be LessThan, EqualTo, and so forth.

`compareValue` Double

The value part of the filtering rule that is applied to the first Y-values of data points. For example, in the filtering rule "<6", 6 is the value.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

`outputSeriesNames` String

A comma-separated list of names of the Series that store the filtered data. The number of output series must match the number of input series, otherwise an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(IDataPointFilter, Series, Series)

Performs custom filtering on the data points in a series.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.IDataPointFilter
filterInterface, System.Windows.Forms.DataVisualization.Charting.Series
inputSeries, System.Windows.Forms.DataVisualization.Charting.Series
outputSeries);
```

Parameters

`filterInterface` IDataPointFilter

An object that implements the IDataPointFilter interface.

`inputSeries` Series

The Series object to be filtered.

`outputSeries` Series

A Series object that stores the filtered data. If the series does not exist, an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(DateRangeType, String, Series)

Filters the data points in a series, either by removing the specified points or by marking them as empty for the specified date/time ranges. The Series object that is filtered is used to store the modified data.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.DateRangeType dateRange,
string rangeElements,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries);
```

Parameters

`dateRange` DateRangeType

A DateRangeType enumeration value that represents a date/time interval, such as days of the week, days of the month, and so forth. Determines the range over which the given elements will be filtered.

`rangeElements` String

Specifies the elements within the date/time range designated by the `dateRange` parameter that will be filtered. Can be a single value, for example, "7"; comma-separated values, for example, "5,6"; a range of values, such as "9-11"; or any variation thereof, such as "5,6,9-11".

`inputSeries` Series

The Series object to be filtered.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(DateRangeType, String, String)

Filters the data points of one or more series, either by removing the specified points or by marking them as empty for the given date/time ranges. The filtered Series objects are used to store the modified data.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.DateRangeType dateRange,
string rangeElements, string inputSeriesNames);
```

Parameters

`dateRange` DateRangeType

A DateRangeType enumeration value that represents a date/time interval, such as days of the week, days of the month, and so forth. Determines the range over which the given elements will be filtered.

`rangeElements` String

Specifies the elements within the date/time range designated by the `dateRange` parameter that will be filtered. Can be a single value, for example, "7"; comma-separated values, for example, "5,6"; a range of values, such as "9-11"; or any variation thereof, such as "5,6,9-11".

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(CompareMethod, Double, Series)

Filters the data points in a series by applying a filtering rule to the first Y-value of each data point. The Series object that is filtered also stores the modified data.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.CompareMethod
compareMethod, double compareValue,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries);
```

Parameters

`compareMethod` CompareMethod

A CompareMethod enumeration value that determines the comparison method for the filtering rule. Can be LessThan, EqualTo, and so forth.

`compareValue` Double

The value part of the filtering rule that is applied to the first Y-values of data points. For example, in the filtering rule "<6", 6 is the value.

`inputSeries` Series

The Series object that is filtered.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(CompareMethod, Double, String)

Filters one or more series by applying a filtering rule to the first Y-value of the data points in the first series. The filtered Series objects are used to store the modified data.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.CompareMethod
compareMethod, double compareValue, string inputSeriesNames);
```

Parameters

`compareMethod` CompareMethod

A CompareMethod enumeration value that determines the comparison method for the filtering rule. Can be LessThan, EqualTo, and so forth.

`compareValue` Double

The value part of the filtering rule that is applied to the first Y-values of data points. For example, in the filtering rule "<6", 6 is the value.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(IDataPointFilter, Series)

Performs custom filtering on the data points in a series. The Series object that is filtered also stores the modified data.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.IDataPointFilter
filterInterface, System.Windows.Forms.DataVisualization.Charting.Series
inputSeries);
```

Parameters

`filterInterface` IDataPointFilter

An object that implements the IDataPointFilter interface.

`inputSeries` Series

The Series object that is filtered.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(IDataPointFilter, String)

Performs custom filtering on the data points in one or more series, based on the points of the first series. The filtered Series objects also store the modified data.

C#

```
public void Filter
(System.Windows.Forms.DataVisualization.Charting.IDataPointFilter
filterInterface, string inputSeriesNames);
```

Parameters

`filterInterface` IDataPointFilter

An object that implements the IDataPointFilter interface.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Filter(IDataPointFilter, String, String)

Performs custom filtering on the data points in one or more series, based on the points of the first series.

```
C#

public void Filter
(System.Windows.Forms.DataVisualization.Charting.IDataPointFilter
filterInterface, string inputSeriesNames, string outputSeriesNames);
```

Parameters

`filterInterface` IDataPointFilter

An object that implements the IDataPointFilter interface.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

`outputSeriesNames` String

A comma-separated list of names of the Series that store the filtered data. Make sure that the number of output series matches the number of input series, otherwise an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.FilterTopN Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Filters all data points in one or more series except for a specified number of points.

Overloads

FilterTopN(Int32, String, String, String, Boolean)	Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first input series that have the largest or smallest values.
FilterTopN(Int32, Series, Series, String, Boolean)	Filters all data points in a series except for a specified number of points with the smallest or largest values.
FilterTopN(Int32, String, String, String)	Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first series that have the largest values.
FilterTopN(Int32, Series, Series, String)	Filters all data points in a series except for a specified number of points with the largest values.
FilterTopN(Int32, String, String)	Filters all data points in one or more series except for a specified number of points. The points that aren't filtered correspond to points in the first series that have the largest first Y-values.
FilterTopN(Int32, Series)	Filters all data points in a series except for a specified number of points with the largest first Y-values. The Series object that is filtered is used to store the modified data.
FilterTopN(Int32, String)	Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first series that have the largest first Y-values. The Series objects that are filtered are used to store the modified data.
FilterTopN(Int32, Series, Series)	Filters all data points in a series except for a specified number of points with the largest first Y-values.

FilterTopN(Int32, String, String, String, Boolean)

Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first input series that have the largest or smallest values.

```
C#

public void FilterTopN (int pointCount, string inputSeriesNames, string
outputSeriesNames, string usingValue, bool getTopValues);
```

Parameters

`pointCount` Int32

The number of data points that the filtering operation will not remove.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

`outputSeriesNames` String

A comma-separated list of names of the Series that store the filtered data. Make sure that the number of output series matches the number of input series, otherwise an exception will be thrown.

`usingValue` String

The data point value that the filtering rule is applied to. Can be X, Y, Y2, Y3, and so forth.

`getTopValues` Boolean

`true` if the largest values are kept; `false` if the smallest values are kept.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FilterTopN(Int32, Series, Series, String, Boolean)

Filters all data points in a series except for a specified number of points with the smallest or largest values.

C#

```
public void FilterTopN (int pointCount,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
System.Windows.Forms.DataVisualization.Charting.Series outputSeries,
string usingValue, bool getTopValues);
```

Parameters

`pointCount` Int32

The number of data points that the filtering operation will not remove.

`inputSeries` Series

The Series object to be filtered.

`outputSeries` Series

A Series object that stores the filtered data. If the series does not exist, an exception will be thrown.

`usingValue` String

The data point value that the filtering rule is applied to. Can be X, Y, Y2, Y3, and so forth.

`getTopValues` Boolean

`true` if the largest values are kept; `false` if the smallest values are kept.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FilterTopN(Int32, String, String, String)

Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first series that have the

largest values.

```
C#

public void FilterTopN (int pointCount, string inputSeriesNames, string
outputSeriesNames, string usingValue);
```

Parameters

`pointCount` Int32

The number of data points that the filtering operation will not remove.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

`outputSeriesNames` String

A comma-separated list of names of the Series that store the filtered data. Make sure that the number of output series matches the number of input series, otherwise an exception will be thrown.

`usingValue` String

The data point value that the filtering rule is applied to. Can be X, Y, Y2, Y3, and so forth.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FilterTopN(Int32, Series, Series, String)

Filters all data points in a series except for a specified number of points with the largest values.

```
C#

public void FilterTopN (int pointCount,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
```

```
System.Windows.Forms.DataVisualization.Charting.Series outputSeries,
string usingValue);
```

Parameters

`pointCount` Int32

The number of data points that the filtering operation will not remove.

`inputSeries` Series

The Series object to be filtered.

`outputSeries` Series

A Series object that stores the filtered data. If the series does not exist, an exception will be thrown.

`usingValue` String

The data point value that the filtering rule is applied to. Can be X, Y, Y2, Y3, and so forth.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FilterTopN(Int32, String, String)

Filters all data points in one or more series except for a specified number of points. The points that aren't filtered correspond to points in the first series that have the largest first Y-values.

C#

```
public void FilterTopN (int pointCount, string inputSeriesNames, string
outputSeriesNames);
```

Parameters

`pointCount` Int32

The number of data points that the filtering operation will not remove.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

`outputSeriesNames` String

A comma-separated list of names of the Series that store the filtered data. Make sure that the number of output series matches the number of input series, otherwise an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FilterTopN(Int32, Series)

Filters all data points in a series except for a specified number of points with the largest first Y-values. The Series object that is filtered is used to store the modified data.

C#

```
public void FilterTopN (int pointCount,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries);
```

Parameters

`pointCount` Int32

The number of data points that the filtering operation will not remove.

`inputSeries` Series

The Series object to be filtered.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FilterTopN(Int32, String)

Filters all data points in one or more series except for a specified number of points. The points that are not filtered correspond to points in the first series that have the largest first Y-values. The Series objects that are filtered are used to store the modified data.

C#

```
public void FilterTopN (int pointCount, string inputSeriesNames);
```

Parameters

`pointCount` Int32

The number of data points that the filtering operation will not remove.

`inputSeriesNames` String

A comma-separated list of names of the Series to be filtered.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FilterTopN(Int32, Series, Series)

Filters all data points in a series except for a specified number of points with the largest first Y-values.

C#

```
public void FilterTopN (int pointCount,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
System.Windows.Forms.DataVisualization.Charting.Series outputSeries);
```

Parameters

`pointCount` Int32

The number of data points that the filtering operation will not remove.

`inputSeries` Series

The Series object to be filtered.

`outputSeries` Series

A Series object that stores the filtered data. If the series does not exist, an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.Group Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Groups data points for one or more series, using one or more formulas.

Overloads

Group(String, Double, IntervalType, String)	Groups data points for one or more series, using one or more formulas. Series are cleared of their original data and used to store the new data points.
Group(String, Double, IntervalType, Series)	Groups data points in a series using one or more formulas. The series that is grouped is cleared of its original data, and used to store the new data points.
Group(String, Double, IntervalType, String, String)	Groups data points for one or more series, using one or more formulas. Output series are used to store the grouped data points.
Group(String, Double, IntervalType, Series, Series)	Groups data points in a series using one or more formulas. An output series is used to store the grouped data points.
Group(String, Double, IntervalType, Double, IntervalType, String)	Groups data points for one or more series, using one or more formulas. Series are cleared of their original data and used to store the new data points.
Group(String, Double, IntervalType, Double, IntervalType, Series)	Groups data points in a series using one or more formulas. The series that is grouped is cleared of its original data, and used to store the new data points.
Group(String, Double, IntervalType, Double, IntervalType, String, String)	Groups data points for one or more series, using one or more formulas. Output series are used to store the grouped data points, and an offset can be used for intervals.
Group(String, Double, IntervalType, Double, IntervalType, Series, Series)	Groups data points in a series using one or more formulas. An output series is used to store the grouped data points, and an offset can be used for intervals.

Group(String, Double, IntervalType, String)

Groups data points for one or more series, using one or more formulas. Series are cleared of their original data and used to store the new data points.

```
C#

public void Group (string formula, double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, string inputSeriesName);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`inputSeriesName` String

A comma-separated list of names of the Series to be grouped.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Group(String, Double, IntervalType, Series)

Groups data points in a series using one or more formulas. The series that is grouped is cleared of its original data, and used to store the new data points.

```
C#

public void Group (string formula, double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
```

```
intervalType, System.Windows.Forms.DataVisualization.Charting.Series
inputSeries);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`inputSeries` Series

The Series object to be grouped.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Group(String, Double, IntervalType, String, String)

Groups data points for one or more series, using one or more formulas. Output series are used to store the grouped data points.

C#

```
public void Group (string formula, double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, string inputSeriesName, string outputSeriesName);
```

Parameters

formula String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

interval Double

The size of the interval checked for missing data points.

intervalType IntervalType

The unit of measurement for the `interval` parameter.

inputSeriesName String

A comma-separated list of names of the Series to be grouped.

outputSeriesName String

A comma-separated list of names of the Series that will store the grouped data points. If a series does not exist, it will be created; if it does exist, it will first be cleared of all data before storing grouped data points. The number of output series must equal the number of input series, or an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Group(String, Double, IntervalType, Series, Series)

Groups data points in a series using one or more formulas. An output series is used to store the grouped data points.

C#

```
public void Group (string formula, double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, System.Windows.Forms.DataVisualization.Charting.Series
inputSeries, System.Windows.Forms.DataVisualization.Charting.Series
outputSeries);
```

Parameters

formula String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

interval Double

The size of the interval checked for missing data points.

intervalType IntervalType

The unit of measurement for the `interval` parameter.

inputSeries Series

The Series object to be grouped.

outputSeries Series

The Series that will store the new data points created by the grouping operation. If it does not exist, an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Group(String, Double, IntervalType, Double, IntervalType, String)

Groups data points for one or more series, using one or more formulas. Series are cleared of their original data and used to store the new data points.

C#

```
public void Group (string formula, double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, double intervalOffset,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalOffsetType, string inputSeriesName);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`intervalOffset` Double

The interval offset size.

`intervalOffsetType` IntervalType

The unit of measurement for the interval offset.

`inputSeriesName` String

A comma-separated list of names of the Series to be grouped.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Group(String, Double, IntervalType, Double, IntervalType, Series)

Groups data points in a series using one or more formulas. The series that is grouped is cleared of its original data, and used to store the new data points.

C#

```
public void Group (string formula, double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, double intervalOffset,
System.Windows.Forms.DataVisualization.Charting.IntervalType
```

```
intervalOffsetType,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`intervalOffset` Double

The interval offset size.

`intervalOffsetType` IntervalType

The unit of measurement for the interval offset.

`inputSeries` Series

The Series object to be grouped.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Group(String, Double, IntervalType, Double, IntervalType, String, String)

Groups data points for one or more series, using one or more formulas. Output series are used to store the grouped data points, and an offset can be used for intervals.

C#

```
public void Group (string formula, double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, double intervalOffset,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalOffsetType, string inputSeriesName, string outputSeriesName);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`intervalOffset` Double

The interval offset size.

`intervalOffsetType` IntervalType

The unit of measurement for the interval offset.

`inputSeriesName` String

A comma-separated list of names of the Series to be grouped.

`outputSeriesName` String

A comma-separated list of names of the Series that will store the grouped data points. If a series does not exist, it will be created; if it does exist, it will first be cleared of all data before storing grouped data points. The number of output series must equal the number of input series, or an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Group(String, Double, IntervalType, Double, IntervalType, Series, Series)

Groups data points in a series using one or more formulas. An output series is used to store the grouped data points, and an offset can be used for intervals.

C#

```
public void Group (string formula, double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, double intervalOffset,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalOffsetType,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
System.Windows.Forms.DataVisualization.Charting.Series outputSeries);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`intervalOffset` Double

The interval offset size.

`intervalOffsetType` IntervalType

The unit of measurement for the interval offset.

`inputSeries` Series

The Series object to be grouped.

`outputSeries` Series

The Series that will store the new data points created by the grouping operation. If it does not exist, an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.GroupByAxisLabel Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Groups data points in one or more series by axis labels, using one or more formulas.

Overloads

GroupByAxisLabel(String, Series, Series)	Groups data points in a series by axis labels, using one or more formulas. An output series is used to store the grouped data points.
GroupByAxisLabel(String, String, String)	Groups data points in one or more series by axis labels, using one or more formulas. Output series are used to store the grouped data points.
GroupByAxisLabel(String, Series)	Groups data points in a series by axis labels, using one or more formulas. Original data point values are cleared, and the series then stores the new grouped data points.
GroupByAxisLabel(String, String)	Groups data points in one or more series by axis labels, using one or more formulas. Original data point values are cleared, and each of the series then stores its new grouped data points.

GroupByAxisLabel(String, Series, Series)

Groups data points in a series by axis labels, using one or more formulas. An output series is used to store the grouped data points.

C#

```
public void GroupByAxisLabel (string formula,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries,
System.Windows.Forms.DataVisualization.Charting.Series outputSeries);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`inputSeries` Series

The input Series to be grouped.

`outputSeries` Series

The Series that will store the new data points created by the grouping operation. If it does not exist, an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

GroupByAxisLabel(String, String, String)

Groups data points in one or more series by axis labels, using one or more formulas. Output series are used to store the grouped data points.

C#

```
public void GroupByAxisLabel (string formula, string inputSeriesName,
string outputSeriesName);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`inputSeriesName` String

A comma-separated list of names of the input Series.

`outputSeriesName` String

A comma-separated list of names of the Series that will store the grouped data points. If a series does not exist, it will be created; if it does exist, it will first be cleared of all data before storing grouped data points. The number of output series must equal the number of input series, or an exception will be thrown.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

GroupByAxisLabel(String, Series)

Groups data points in a series by axis labels, using one or more formulas. Original data point values are cleared, and the series then stores the new grouped data points.

C#

```
public void GroupByAxisLabel (string formula,
System.Windows.Forms.DataVisualization.Charting.Series inputSeries);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`inputSeries` Series

The input Series.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

GroupByAxisLabel(String, String)

Groups data points in one or more series by axis labels, using one or more formulas. Original data point values are cleared, and each of the series then stores its new grouped data points.

```
C#
public void GroupByAxisLabel (string formula, string inputSeriesName);
```

Parameters

`formula` String

A comma-separated list of one or more formulas to be used for calculating the grouped data point values.

`inputSeriesName` String

A comma-separated list of names of the input Series.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.InsertEmptyPoints Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Replaces the missing data points in a series or a collection of series with empty points.

Overloads

InsertEmptyPoints(Double, IntervalType, String)	Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval` and `intervalType` arguments.
InsertEmptyPoints(Double, IntervalType, Series)	Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval` and `intervalType` arguments.
InsertEmptyPoints(Double, IntervalType, Double, IntervalType, String)	Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments.
InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Series)	Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments.
InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Double, Double, String)	Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments. The series range that is checked is determined by the `toXValue` and `fromXValue` arguments.
InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Double, Double, Series)	Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments. The series range that is checked is determined by the `toXValue` and `fromXValue` arguments.

InsertEmptyPoints(Double, IntervalType, String)

Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval` and `intervalType` arguments.

C#

```
public void InsertEmptyPoints (double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, string seriesName);
```

Parameters

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`seriesName` String

A comma-separated list of names of the Series to be checked.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

InsertEmptyPoints(Double, IntervalType, Series)

Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval` and `intervalType` arguments.

C#

```
public void InsertEmptyPoints (double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
```

```
intervalType, System.Windows.Forms.DataVisualization.Charting.Series
series);
```

Parameters

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`series` Series

The Series object that is checked.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

InsertEmptyPoints(Double, IntervalType, Double, IntervalType, String)

Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments.

C#

```
public void InsertEmptyPoints (double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, double intervalOffset,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalOffsetType, string seriesName);
```

Parameters

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`intervalOffset` Double

The interval offset size.

`intervalOffsetType` IntervalType

The unit of measurement for the interval offset.

`seriesName` String

A comma-separated list of names of the Series to be checked.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Series)

Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments.

C#

```
public void InsertEmptyPoints (double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, double intervalOffset,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalOffsetType,
System.Windows.Forms.DataVisualization.Charting.Series series);
```

Parameters

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`intervalOffset` Double

The interval offset size.

`intervalOffsetType` IntervalType

The unit of measurement for the interval offset.

`series` Series

The Series object in which to insert the empty points.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Double, Double, String)

Replaces the missing data points in one or more series with empty points. Each series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments. The series range that is checked is determined by the `toXValue` and `fromXValue` arguments.

C#

```
public void InsertEmptyPoints (double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, double intervalOffset,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalOffsetType, double fromXValue, double toXValue, string
seriesName);
```

Parameters

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`intervalOffset` Double

The interval offset size.

`intervalOffsetType` IntervalType

The unit of measurement for the interval offset.

`fromXValue` Double

The starting point for the series range to be checked. Uses the X-values of the data points.

`toXValue` Double

The end point for the series range to be checked. Uses the X-values of the data points.

`seriesName` String

A comma-separated list of names of the Series to be checked.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

InsertEmptyPoints(Double, IntervalType, Double, IntervalType, Double, Double, Series)

Replaces the missing data points in a series with empty points. The series is checked for a given interval that is determined by the `interval`, `intervalOffset`, `intervalOffsetType` and `intervalType` arguments. The series range that is checked is determined by the `toXValue` and `fromXValue` arguments.

```C#
public void InsertEmptyPoints (double interval,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalType, double intervalOffset,
System.Windows.Forms.DataVisualization.Charting.IntervalType
intervalOffsetType, double fromXValue, double toXValue,
System.Windows.Forms.DataVisualization.Charting.Series series);
```

Parameters

`interval` Double

The size of the interval checked for missing data points.

`intervalType` IntervalType

The unit of measurement for the `interval` parameter.

`intervalOffset` Double

The interval offset size.

`intervalOffsetType` IntervalType

The unit of measurement for the interval offset.

`fromXValue` Double

The starting point for the series range to be checked. Uses the X-values of the data points.

`toXValue` Double

The end point for the series range to be checked. Uses the X-values of the data points.

`series` Series

The Series in which to insert the empty points.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataManipulator.Sort Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs a custom sort on the data points in one or more series.

Overloads

Sort(PointSortOrder, String, String)	Performs a sort on the data points in one or more series, using the specified order. The sort can be based on any available data point value.
Sort(IComparer<DataPoint>, String)	Performs a custom sort on the data points in one or more series.
Sort(IComparer<DataPoint>, Series)	Performs a custom sort on the data points in a series.
Sort(PointSortOrder, String)	Performs a sort on the data points in one or more series, using the specified order. The sort is based on the first Y-value of each of the data points.
Sort(PointSortOrder, Series)	Performs a sort on the data points in a series using the specified order. The sort is based on the first Y-value of each of the data points.
Sort(PointSortOrder, String, Series)	Performs a sort on the data points in a series using the specified order. The sort can be based on any available data point value.

Sort(PointSortOrder, String, String)

Performs a sort on the data points in one or more series, using the specified order. The sort can be based on any available data point value.

C#

```
public void Sort
(System.Windows.Forms.DataVisualization.Charting.PointSortOrder
pointSortOrder, string sortBy, string seriesName);
```

Parameters

`pointSortOrder` PointSortOrder

A PointSortOrder enumeration value that determines if the sort is ascending or descending.

`sortBy` String

A `string` value that represents the data point values to be sorted. Possible values are: "X", "Y", "Y2", "Y3", "AxisLabel", and so forth.

`seriesName` String

A comma-separated list of names of the Series to be sorted.

Remarks

This method performs an ascending or descending sort on data points in a series, and is based on the value that is set in the `sortBy` parameter such as "X", "AxisLabel", "Y", "Y1", and so forth.

Note that if multiple series are sorted, the series must be aligned, otherwise an exception will be thrown.

> **Important**
>
> All series will be sorted using the specified Y-value from the FIRST listed series; that is, all data points, regardless of the series they belong to, are sorted based on the first series.

To sort points using a custom sort routine, use a different definition of this overloaded method.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Sort(IComparer<DataPoint>, String)

Performs a custom sort on the data points in one or more series.

C#

```
public void Sort
(System.Collections.Generic.IComparer<System.Windows.Forms.DataVisualizat
ion.Charting.DataPoint> comparer, string seriesName);
```

Parameters

comparer IComparer<DataPoint>

An object that implements the IComparer interface and provides the custom sorting logic.

seriesName String

A comma-separated list of names of the Series to be sorted.

Remarks

This method performs a custom sort on the data points in one or more series; the sort logic is defined by the `comparer` parameter. Override and implement the Compare method of the IComparer interface, perform the comparison(s) and then return `true` or `false`. The sort order is determined by the return value.

Note that if multiple series are sorted, the series must be aligned, otherwise an exception will be thrown.

The two object parameters of the Compare method must always be of type DataPoint.

> **Important**
>
> All data points, regardless of the series they belong to, are sorted based on the first series; if the position of a data point in the first series changes, all corresponding points in the other series change as well.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Sort(IComparer<DataPoint>, Series)

Performs a custom sort on the data points in a series.

C#

```
public void Sort
(System.Collections.Generic.IComparer<System.Windows.Forms.DataVisualizat
ion.Charting.DataPoint> comparer,
System.Windows.Forms.DataVisualization.Charting.Series series);
```

Parameters

`comparer` IComparer<DataPoint>

An object that implements the IComparer interface and provides the custom sorting logic.

`series` Series

The Series object to be sorted.

Remarks

This method performs a custom sort on the data points in a series; the sort logic is defined by the `comparer` parameter. Override and implement the Compare method of the IComparer interface, perform the comparison(s) and then return `true` or `false`. The sort order is determined by the return value.

The two object parameters of the Compare method must always be of type DataPoint.

To perform a custom sort for more than one series per function call, use the Sort method.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Sort(PointSortOrder, String)

Performs a sort on the data points in one or more series, using the specified order. The sort is based on the first Y-value of each of the data points.

C#

```
public void Sort
(System.Windows.Forms.DataVisualization.Charting.PointSortOrder
pointSortOrder, string seriesName);
```

Parameters

`pointSortOrder` PointSortOrder

A PointSortOrder enumeration value that determines if the sort is ascending or descending.

`seriesName` String

A comma-separated list of names of the Series to be sorted.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Sort(PointSortOrder, Series)

Performs a sort on the data points in a series using the specified order. The sort is based on the first Y-value of each of the data points.

C#

```
public void Sort
(System.Windows.Forms.DataVisualization.Charting.PointSortOrder
```

```
pointSortOrder, System.Windows.Forms.DataVisualization.Charting.Series
series);
```

Parameters

`pointSortOrder` PointSortOrder

A PointSortOrder enumeration value that determines if the sort is ascending or descending.

`series` Series

The Series object to be sorted.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Sort(PointSortOrder, String, Series)

Performs a sort on the data points in a series using the specified order. The sort can be based on any available data point value.

C#

```
public void Sort
(System.Windows.Forms.DataVisualization.Charting.PointSortOrder
pointSortOrder, string sortBy,
System.Windows.Forms.DataVisualization.Charting.Series series);
```

Parameters

`pointSortOrder` PointSortOrder

A PointSortOrder enumeration value that determines if the sort is ascending or descending.

`sortBy` String

A `string` value that represents the data point values to be sorted. Possible values are: "X", "Y", "Y2", "Y3", "AxisLabel", and so forth.

`series` Series

The Series object to be sorted.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a data point that is stored in the DataPointCollection class.

C#

```
[System.ComponentModel.TypeConverter("System.Windows.Forms.Design.DataVisual
ization.Charting.DataPointConverter,
System.Windows.Forms.DataVisualization.Design, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35")]
public class DataPoint :
System.Windows.Forms.DataVisualization.Charting.DataPointCustomProperties
```

Inheritance Object → ChartElement → ChartNamedElement → DataPointCustomProperties → DataPoint

Attributes TypeConverterAttribute

Remarks

The DataPoint class stores properties associated with data as well as data point values. For example, a data point has a value, but it also has a color property, background image property, background gradient property, and so forth. For more information about data point properties, see the DataPointCustomProperties class overview topic.

Each data point consists of an X-value and one or more Y-values. The X-value can be zero, or you can set this explicitly.

Only one Y-value per point is required for all chart types except bubble, candlestick and stock charts. These chart types require more than one Y-value because one data point consists of multiple values. For example, to plot one stock chart column, four values are required: high, low, open and close values.

Data can be added at either design time or run time; you can also use data-binding at run time.

Constructors

DataPoint()	Initializes a new instance of the DataPoint class.
DataPoint(Double, Double)	Initializes a new instance of the DataPoint class with the specified X and Y-value.
DataPoint(Double, Double[])	Initializes a new instance of the DataPoint class with the specified X-value and an array of Y-values.
DataPoint(Double, String)	Initializes a new instance of the DataPoint class with the specified X-value and Y-values.
DataPoint(Series)	Initializes a new instance of the DataPoint class, and sets the Series that the data point belongs to.

Properties

AxisLabel	Gets or sets the text of the X-axis label for the data point, series or an empty point. This property is only used if a custom label has not been specified for the relevant Axis object. (Inherited from DataPointCustomProperties)
BackGradientStyle	Gets or sets the background gradient style. (Inherited from DataPointCustomProperties)
BackHatchStyle	Gets or sets the background hatching style. (Inherited from DataPointCustomProperties)
BackImage	Gets or sets the background image of the data point. (Inherited from DataPointCustomProperties)
BackImageAlignment	Gets or sets the alignment of the background image, which is used with the Unscaled drawing mode. (Inherited from DataPointCustomProperties)
BackImageTransparentColor	Gets or sets a color that will be replaced with a transparent color when the background image is drawn. (Inherited from DataPointCustomProperties)
BackImageWrapMode	Gets or sets the drawing mode of the background image. (Inherited from DataPointCustomProperties)
BackSecondaryColor	Gets or sets the secondary background color. (Inherited from DataPointCustomProperties)
BorderColor	Gets or sets the border color of the data point. (Inherited from DataPointCustomProperties)

BorderDashStyle	Gets or sets the border style of the data point. (Inherited from DataPointCustomProperties)
BorderWidth	Gets or sets the border width of the data point. (Inherited from DataPointCustomProperties)
Color	Gets or sets the color of the data point. (Inherited from DataPointCustomProperties)
CustomProperties	Gets or sets the custom properties of the data point. Custom properties can be specified in the following format: AttrName1=Value1, AttrName2=Value2 ... (Inherited from DataPointCustomProperties)
CustomPropertiesExtended	Gets or sets the extended custom properties of the data point. Extended custom properties can be specified in the following format: AttrName1=Value1, AttrName2=Value2... (Inherited from DataPointCustomProperties)
Font	Gets or sets the font of the data point. (Inherited from DataPointCustomProperties)
IsEmpty	Gets or sets a flag that indicates whether a data point is marked as empty.
IsValueShownAsLabel	Gets or sets a flag that indicates whether to show the value of the data point on the label. (Inherited from DataPointCustomProperties)
IsVisibleInLegend	Gets or sets a flag that indicates whether the item is shown in the legend. (Inherited from DataPointCustomProperties)
Item[Int32]	Gets the DataPointCustomProperties object by its position in the index. (Inherited from DataPointCustomProperties)
Item[String]	Gets or sets the DataPointCustomProperties object by name. (Inherited from DataPointCustomProperties)
Label	Gets or sets the text of the data point label. (Inherited from DataPointCustomProperties)
LabelAngle	Gets or sets the angle of the data point label. (Inherited from DataPointCustomProperties)
LabelBackColor	Gets or sets the background color of the data point label. (Inherited from DataPointCustomProperties)
LabelBorderColor	Gets or sets the border color of the data point label. (Inherited from DataPointCustomProperties)

LabelBorderDashStyle	Gets or sets the border style of the label. (Inherited from DataPointCustomProperties)
LabelBorderWidth	Gets or sets the border width of the label. (Inherited from DataPointCustomProperties)
LabelForeColor	Gets or sets the text color of the label. (Inherited from DataPointCustomProperties)
LabelFormat	Gets or sets the format of the data point label. (Inherited from DataPointCustomProperties)
LabelToolTip	Gets or sets the tooltip for the data point label. (Inherited from DataPointCustomProperties)
LegendText	Gets or sets the text of the item in the legend. (Inherited from DataPointCustomProperties)
LegendToolTip	Gets or sets the tooltip of the item in the legend. (Inherited from DataPointCustomProperties)
MarkerBorderColor	Gets or sets the border color of the marker. (Inherited from DataPointCustomProperties)
MarkerBorderWidth	Gets or sets the border width of the marker. (Inherited from DataPointCustomProperties)
MarkerColor	Gets or sets the marker color. (Inherited from DataPointCustomProperties)
MarkerImage	Gets or sets the marker image. (Inherited from DataPointCustomProperties)
MarkerImageTransparentColor	Gets or sets the color that will be replaced with a transparent color when the marker image is drawn. (Inherited from DataPointCustomProperties)
MarkerSize	Gets or sets the size of the marker. (Inherited from DataPointCustomProperties)
MarkerStyle	Gets or sets the marker style. (Inherited from DataPointCustomProperties)
Name	Gets or sets the name of the data point.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
ToolTip	Gets or sets the tooltip. (Inherited from DataPointCustomProperties)
XValue	Gets or sets the X-value of a data point.

YValues	Gets or sets the Y-value(s) of a data point.

Methods

Clone()	Returns a new instance that is an exact copy of the data point.
DeleteCustomProperty(String)	Deletes the data point custom property with the specified name. (Inherited from DataPointCustomProperties)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources. (Inherited from ChartElement)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetCustomProperty(String)	Returns the data point custom property with the specified name. (Inherited from DataPointCustomProperties)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
GetValueByName(String)	Returns a data point that is specified by its name. This is a helper function.
IsCustomPropertySet(String)	Determines whether a custom property with the specified name was set. (Inherited from DataPointCustomProperties)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResetIsValueShownAsLabel()	Resets a custom property to its default value. (Inherited from DataPointCustomProperties)
ResetIsVisibleInLegend()	Resets a custom property to its default value. (Inherited from DataPointCustomProperties)
SetCustomProperty(String, String)	Sets a custom property of the data point. (Inherited from DataPointCustomProperties)
SetDefault(Boolean)	Sets the default properties of the data point. Indicates whether the previous property values must be cleared.

	(Inherited from DataPointCustomProperties)
SetValueXY(Object, Object[])	Sets the X-value and one or more Y-values of the data point.
SetValueY(Object[])	Sets the Y-value(s) of a single data point.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the DataPoint class.

Overloads

DataPoint()	Initializes a new instance of the DataPoint class.
DataPoint(Series)	Initializes a new instance of the DataPoint class, and sets the Series that the data point belongs to.
DataPoint(Double, Double)	Initializes a new instance of the DataPoint class with the specified X and Y-value.
DataPoint(Double, Double[])	Initializes a new instance of the DataPoint class with the specified X-value and an array of Y-values.
DataPoint(Double, String)	Initializes a new instance of the DataPoint class with the specified X-value and Y-values.

DataPoint()

Initializes a new instance of the DataPoint class.

C#

```
public DataPoint ();
```

Remarks

This constructor can be used to create and initialize a new instance of the DataPoint class.

However, it is highly recommended that you instead add data points at run time, using the Points collection property.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint(Series)

Initializes a new instance of the DataPoint class, and sets the Series that the data point belongs to.

C#

```
public DataPoint (System.Windows.Forms.DataVisualization.Charting.Series
series);
```

Parameters

`series` Series

The Series object that the data point belongs to.

Remarks

This constructor can be used to create and initialize a new instance of the DataPoint class.

However, it is highly recommended that you instead add data points at run time, using the Points collection property.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint(Double, Double)

Initializes a new instance of the DataPoint class with the specified X and Y-value.

```
C#

public DataPoint (double xValue, double yValue);
```

Parameters

xValue Double

The X-value of the data point.

yValue Double

The Y-value of the data point.

Remarks

This constructor can be used to create and initialize a new instance of the DataPoint class.

However, it is highly recommended that you instead add data points at run time, using the Points collection property.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint(Double, Double[])

Initializes a new instance of the DataPoint class with the specified X-value and an array of Y-values.

```
C#

public DataPoint (double xValue, double[] yValues);
```

Parameters

`xValue` Double

The X-value of the data point.

`yValues` Double[]

An array of Y-values of the data point.

Remarks

This constructor can be used to create and initialize a new instance of the DataPoint class.

However, it is highly recommended that you instead add data points at run time, using the Points collection property.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint(Double, String)

Initializes a new instance of the DataPoint class with the specified X-value and Y-values.

C#

```
public DataPoint (double xValue, string yValues);
```

Parameters

`xValue` Double

The X-value of the data point.

`yValues` String

A comma-separated `string` of Y-values of the data point.

Remarks

This constructor can be used to create and initialize a new instance of the DataPoint class.

However, it is highly recommended that you instead add data points at run time, using the Points collection property.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint.IsEmpty Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether a data point is marked as empty.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsEmpty { get; set; }
```

Property Value

Boolean

`true` if the point is marked as an empty point; otherwise, `false`. The default value is `false`.

Attributes BindableAttribute

Remarks

Set this property to `true` to mark a data point as an empty point.

Alternatively one of the `InsertEmptyPoints` methods in the DataManipulator class can be used to mark missing data as empty points.

The EmptyPointStyle property determines how empty points are displayed.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the data point.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the text name of the data point.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint.XValue Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the X-value of a data point.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DataPointValueConverter))]
public double XValue { get; set; }
```

Property Value

Double

A `double` that represents the X-value of a data point.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

All chart types can be displayed as either scatter plots or as non-scatter plots, because both X-values and Y-values can be set. Setting an X-value will result in a scatter plot, whereas not setting this property - which is equivalent to setting it to zero - will result in a non-scatter plot. For example, to draw a point chart type, add data points that have X-values of zero. To display a traditional scatter chart, add data points that have X-values other than zero.

The X-values of data points are not required to be set. If they are not set, data points are plotted sequentially, using their index values from the DataPointCollection.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint.YValues Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Y-value(s) of a data point.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.DoubleArrayConverter))]
public double[] YValues { get; set; }
```

Property Value

Double[]

An array of `double` values that represent the Y-value(s) of a data point.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The YValues property is used to set the Y-values of data points.

Only one Y-value per point is required for all chart types except bubble, candlestick and stock charts. These chart types require more than one Y-value because one data point consists of multiple values. For example, to plot one stock chart column, four values are required: high, low, open and close values.

The YValues property returns an array of `double` values when used to retrieve the Y-values.

> **Important**
>
> The **YValuesPerPoint** property determines the maximum number of Y-values that all data points in a **Series** can have. If you specify more than the allowable number

of Y-values, an exception will be raised.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint.Clone Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns a new instance that is an exact copy of the data point.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint Clone ();
```

Returns

DataPoint

A cloned DataPoint object that is an exact copy of the data point.

Remarks

Use the Clone method to obtain an exact copy of the data point.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint.GetValueByName(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns a data point that is specified by its name. This is a helper function.

C#

```
public double GetValueByName (string valueName);
```

Parameters

`valueName` String

Point value names. X, Y, Y2...

Returns

Double

A `string` value that represents a data point.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint.SetValueXY(Object, Object[]) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets the X-value and one or more Y-values of the data point.

C#

```
public void SetValueXY (object xValue, params object[] yValue);
```

Parameters

xValue Object

The X-value of the data point.

yValue Object[]

A list of Y-values of the data point.

Remarks

Use the SetValueXY method to set the value(s) of a data point at run-time.

Note that if you set the X-value to a value other than zero (0), a scatter plot will be generated. For further details, see the Remarks section of the XValue property.

Refer to the following table for a complete listing of valid .NET Framework types that can be used for the object type parameter.

String	DateTime	Double
Decimal	Single	Int32
UInt32	Int64	UInt64

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPoint.SetValueY(Object[]) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets the Y-value(s) of a single data point.

C#

```
public void SetValueY (params object[] yValue);
```

Parameters

yValue Object[]

The Y-value(s) of a DataPoint object in the collection. Formatted as one or more values separated by commas.

Remarks

The SetValueY property is used to set the Y-value(s) of a data point at run time.

Only one Y-value per point is required for all chart types except for bubble, candlestick and stock charts. These chart types require more than one Y-value because one data point consists of multiple values. For example, to plot one stock chart column, four values are required: high, low, open and close values.

This property returns an array of `double` values when used to retrieve multiple Y-values.

> **Important**
>
> The **YValuesPerPoint** property determines the maximum number of Y-values that all data points in a **Series** can have. If you specify more than the allowable number of Y-values, an exception will be raised.
>
> Refer to the following table for a complete listing of valid .NET Framework types that can be used for the object type parameter:

String	DateTime	Double
Decimal	Single	Int32
UInt32	Int64	UInt64

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of DataPoint objects.

C#

```
public class DataPointCollection :
System.Windows.Forms.DataVisualization.Charting.ChartElementCollection<Syste
m.Windows.Forms.DataVisualization.Charting.DataPoint>
```

Inheritance Object → Collection<T> → ChartElementCollection<DataPoint> → DataPointCollection

Remarks

The DataPointCollection class represents a collection of DataPoint objects, which in turn represent the data points that are part of a Series object.

This collection class is implemented as the Points property of a Series object.

DataPoint objects can be appended or inserted into the collection with their X-value and Y-value(s) already specified as parameters. Note that they are also added automatically when a series is bound to a data source using either the DataBindY or DataBindXY method.

Data points are plotted in the order that they occur in this collection only if the series that the collection belongs to has its IsXValueIndexed property set to `true` (the default value is `false`), or if all X-values for all data points are zero. Otherwise data points are plotted using their X-values.

Data points can be located, and manipulated, based on maximum, minimum or specified X or Y-values using the overloaded `FindMaxValue`, `FindMinValue` or `FindValue` methods, respectively.

The Count property indicates the total number of items in the collection and is commonly used to find the upper bound of the collection.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)

Methods

Add(Double[])	Adds a new DataPoint object to the collection and sets its Y value.
Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
AddXY(Double, Double)	Adds a DataPoint object to the end of the collection, with the specified X-value and Y-value.
AddXY(Object, Object[])	Adds a DataPoint object to the end of the collection, with the specified X-value and Y-value(s).
AddY(Double)	Adds a DataPoint object to the end of the collection, with the specified Y-value.
AddY(Object[])	Adds a DataPoint object to the end of the collection, with the specified Y-value(s).
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>.
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)

DataBind(IEnumerable, String, String, String)	Data binds the X-value, Y-value(s) and property values of the data points, such as Tooltip or LabelStyle, to the data source.
DataBindXY(IEnumerable, IEnumerable[])	Data binds the X-value and Y-values of the collection's data points to the first columns of the specified data source.
DataBindXY(IEnumerable, String, IEnumerable, String)	Data binds the X-value and Y-values of the data points in the collection to the specified columns of the specified data sources.
DataBindY(IEnumerable, String)	Data binds the Y-value(s) of the data points to the specified column(s) of the specified data source.
DataBindY(IEnumerable[])	Data binds the Y-value(s) of the collection's data points to the first column of the specified data source(s).
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindAllByValue(Double)	Finds all the data points that are equal to the specified value.
FindAllByValue(Double, String)	Finds all the data points that are equal to the specified value.
FindAllByValue(Double, String, Int32)	Finds all the data points that are equal to the specified value, starting from the specified index location.
FindByValue(Double)	Finds the first data point that is equal to the specified value.
FindByValue(Double, String)	Finds the first data point that is equal to the specified value.
FindByValue(Double, String, Int32)	Finds the first data point that is equal to the specified value, starting from the specified index location.
FindMaxByValue()	Finds the data point with the maximum value.
FindMaxByValue(String)	Finds the data point value with the maximum value.
FindMaxByValue(String, Int32)	Finds the data point with the maximum value, starting from the specified index location.
FindMinByValue()	Finds the data point with the minimum value.
FindMinByValue(String)	Finds the data point with the minimum value.
FindMinByValue(String, Int32)	Finds the data point with the minimum value, starting from the

	specified index location.
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts an element into the Collection<T> at the specified index location. (Inherited from ChartElementCollection<T>)
InsertXY(Int32, Object, Object[])	Inserts a data point with the specified X value and one or more specified Y values.
InsertY(Int32, Object[])	Inserts a data point with one or more specified Y values.
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the element at the specified index location of the Collection<T>. (Inherited from ChartElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the element at the specified index location. (Inherited from ChartElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection.

	(Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a

	transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.

Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource>(IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource>(IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEqualityComparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.

FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>,	Correlates the elements of two sequences based on equality of keys and groups the

Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum

	Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.

Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.

Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.

Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by

	invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.

ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEqualityComparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.

AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement>(IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.Add(Double[]) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a new DataPoint object to the collection and sets its Y value.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint Add (params double[] y);
```

Parameters

y Double[]

The Y-value of the data point.

Returns

DataPoint

The newly added DataPoint object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.AddXY Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a DataPoint object to the end of the collection, with the specified X-value and Y-value.

Overloads

AddXY(Double, Double)	Adds a DataPoint object to the end of the collection, with the specified X-value and Y-value.
AddXY(Object, Object[])	Adds a DataPoint object to the end of the collection, with the specified X-value and Y-value(s).

AddXY(Double, Double)

Adds a DataPoint object to the end of the collection, with the specified X-value and Y-value.

C#

```
public int AddXY (double xValue, double yValue);
```

Parameters

xValue Double

The X-value of the data point.

yValue Double

The Y-value of the data point.

Returns

Int32

An `integer` that represents the zero-based index where the item was inserted into the data point collection.

Remarks

This method adds DataPoint object to the DataPointCollection; the object is always added to the end of the collection.

Use the AddXY method overload if your data points require more than one Y-value.

If your data points do not need an X-value - that is, if you are creating non-scatter plots - use the AddY method instead.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AddXY(Object, Object[])

Adds a DataPoint object to the end of the collection, with the specified X-value and Y-value(s).

C#

```
public int AddXY (object xValue, params object[] yValue);
```

Parameters

`xValue` Object

The X value of the data point.

`yValue` Object[]

One or more comma-separated values that represent the Y-value(s) of the data point.

Returns

Int32

An `integer` value that represents the zero-based index where the item was inserted into the collection.

Remarks

This method adds one DataPoint object to the DataPointCollection; the data point is always added to the end of the collection.

You must provide at least one (1) Y-value, otherwise an exception will be thrown. This method also checks the YValueType property of the DataPointCollection object that this data belongs to; if too many Y-values are specified, an exception will be thrown.

In order for DateTime formatting to have an effect, a value must be a DateTime object.

If your data points do not need an X-value - that is, if you are creating non-scatter plots - use the AddY method instead

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.AddY Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a DataPoint object to the end of the collection, with the specified Y-value.

Overloads

AddY(Double)	Adds a DataPoint object to the end of the collection, with the specified Y-value.
AddY(Object[])	Adds a DataPoint object to the end of the collection, with the specified Y-value(s).

AddY(Double)

Adds a DataPoint object to the end of the collection, with the specified Y-value.

C#

```
public int AddY (double yValue);
```

Parameters

`yValue` Double

The Y-value of the data point.

Returns

Int32

An `integer` that represents the zero-based index where the item was inserted into the data point collection.

Remarks

This method adds one DataPoint object to the DataPointCollection; the data point is always added to the end of the collection.

Use the method definition that allows for an array of Y-values if your data points require more than one Y-value.

If your data points need an X-value - that is, if you are creating scatter plots - use one of the `AddXY` methods instead.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

AddY(Object[])

Adds a DataPoint object to the end of the collection, with the specified Y-value(s).

```
C#

public int AddY (params object[] yValue);
```

Parameters

`yValue` Object[]

A comma-separated list of Y-value(s) of the DataPoint object added to the collection.

Returns

Int32

An `integer` that represents the location in zero-based index where the item was inserted into the collection.

Remarks

This method adds one DataPoint object to the DataPointCollection; the data point is always added to the end of the collection.

You must provide at least one (1) Y-value, otherwise an exception will be thrown. This method also checks the ChartType property of the DataPoint object this data belongs to; if too many Y-values are specified, an exception will be thrown.

The X-value will always be set to zero (0), resulting in non-scatter plots. If you want the data points to use an X-value, call one of the `AddXY` methods instead.

In order for DateTime formatting to have an effect, a value must be a DateTime object.

Refer to the following for a complete listing of valid .NET Framework types that can be used for the object type parameters:

String	DateTime	Double
Decimal	Single	Int32
UInt32	Int64	UInt64

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.ClearItems Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Removes all elements from the Collection<T>.

C#

```
protected override void ClearItems ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.DataBind(IEnumerable, String, String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Data binds the X-value, Y-value(s) and property values of the data points, such as Tooltip or LabelStyle, to the data source.

C#

```
public void DataBind (System.Collections.IEnumerable dataSource, string
xField, string yFields, string otherFields);
```

Parameters

`dataSource` IEnumerable

The data source to bind with X-value and Y-value(s).

`xField` String

The name of the field for X-values.

`yFields` String

A comma separated names of the fields for Y-values.

`otherFields` String

Other data point properties with binding rules, in the format: PointProperty=Field[{Format}] [,PointProperty=Field[{Format}]].

For example: "Tooltip=Price{C1},Url=WebSiteName".

Remarks

The data source can be the Ole(SQL)DataReader, DataView, DataSet, DataTable or DataRow.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.DataBindXY Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Data binds the X-value and Y-values of the data points in the collection to the specified columns of the specified data sources.

Overloads

DataBindXY(IEnumerable, IEnumerable[])	Data binds the X-value and Y-values of the collection's data points to the first columns of the specified data source.
DataBindXY(IEnumerable, String, IEnumerable, String)	Data binds the X-value and Y-values of the data points in the collection to the specified columns of the specified data sources.

DataBindXY(IEnumerable, IEnumerable[])

Data binds the X-value and Y-values of the collection's data points to the first columns of the specified data source.

C#

```
public void DataBindXY (System.Collections.IEnumerable xValue, params
System.Collections.IEnumerable[] yValues);
```

Parameters

xValue IEnumerable

The data source that will supply the X-values for the data points.

yValues IEnumerable[]

A comma-separated list of the Y-value(s) of the DataPoint object added to the collection.

Remarks

This method binds all DataPoint objects in a Series, and uses separate data sources for the X and Y-value(s).

Note that the first available column in the data source will be used if data-binding to a table. To bind to a column other than the first column, use the DataBindXY method.

You can bind to multiple Y-values by providing a comma-separated list of IEnumerable<T> objects for the `yValues` parameter.

If Y-values are not provided by the data source, or if the wrong number of values are provided, an exception will be thrown.

If you do not want to display scatter-type plots, in which both X and Y-values are used, use one of the `DataBindY` methods instead.

The following is a list of objects that you can use as the data source parameter:

- DataView
- Data readers (SQL, OleDB)
- Arrays
- Lists
- All other objects that use the IEnumerable<T> interface.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataBindXY(IEnumerable, String, IEnumerable, String)

Data binds the X-value and Y-values of the data points in the collection to the specified columns of the specified data sources.

```
C#
```

```
public void DataBindXY (System.Collections.IEnumerable xValue, string
xField, System.Collections.IEnumerable yValue, string yFields);
```

Parameters

`xValue` IEnumerable

The data source that will supply the X-values for the data points.

`xField` String

The name of the column that will supply the X-values for the data points.

`yValue` IEnumerable

A comma-separated list of the Y-value(s) of the DataPoint object added to the collection.

`yFields` String

A comma-separated list of column name(s) that will supply the Y-values for the data points. Note that a comma can be embedded as part of a column name, by using a double comma.

Remarks

Data binds the X-value and Y-values of the data points in the collection to the specified data source's first columns.

You can bind to several Y-values by setting the `yFields` parameter to multiple column names. Separate the column names with commas.

If Y-values are not provided by the data source, or if the wrong number of values are provided, an exception will be thrown.

If you do not want to display scatter-type plots, in which both X and Y-values are used, use one of the `DataBindY` methods instead.

The following is a list of objects that you can use as the data source parameter:

- DataView
- Data readers (SQL, OleDB)
- Arrays

- Lists
- All other objects that use the IEnumerable<T> interface.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.DataBindY Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Data binds the Y-value(s) of the data points to the specified column(s) of the specified data source.

Overloads

DataBindY(IEnumerable[])	Data binds the Y-value(s) of the collection's data points to the first column of the specified data source(s).
DataBindY(IEnumerable, String)	Data binds the Y-value(s) of the data points to the specified column(s) of the specified data source.

DataBindY(IEnumerable[])

Data binds the Y-value(s) of the collection's data points to the first column of the specified data source(s).

C#

```
public void DataBindY (params System.Collections.IEnumerable[] yValue);
```

Parameters

yValue IEnumerable[]

One or more comma-separated IEnumerable<T> data sources.

Remarks

This method binds the Y-value(s) of the data points in the collection to the first column of the specified data source(s).

To bind Y-values to a data source column that is not the first available column, use the DataBindY method.

You can bind several Y-values by specifying multiple data sources, using the `yValue` parameter. Note that the first available column in each data source will be used for each data point's successive Y-values. For example, the first column of the first IEnumerable<T> data source will be bound to the first Y-value of data points, the first column of the second IEnumerable<T> data source will be bound to the second Y-value of data points, and so forth.

If Y-values are not provided by the data source, or if the wrong number of values are provided, an exception will be thrown.

The following is a list of objects that you can use as the data source parameter:

The following is a list of objects that you can use as the data source:

- DataView
- Data readers (SQL, OleDB)
- Arrays
- Lists
- All other objects that use the IEnumerable<T> interface.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataBindY(IEnumerable, String)

Data binds the Y-value(s) of the data points to the specified column(s) of the specified data source.

C#

```
public void DataBindY (System.Collections.IEnumerable yValue, string
yFields);
```

Parameters

`yValue` IEnumerable

An IEnumerable<T> data source object that provides the Y-value(s).

`yFields` String

The data source field(s) to which to bind data point(s). Note that a comma can be embedded as part of a column name, by using a double comma.

Remarks

This method binds the Y-values of the data points in the collection to the specified fields(s) of the data source.

To bind only to the first field in a data source, use the DataBindY method.

You can bind to several fields by setting the `yFields` parameter to multiple field names that are comma-separated. Note that the first field will be used for the data point's first Y-value, the second named field will be used for the data point's second Y-value, and so forth.

If Y-values are not provided by the data source, or the wrong number of values are provided, an exception will be thrown.

The following is a list of objects that you can use as the data source:

- DataView
- Data readers (SQL, OleDB)
- Arrays
- Lists
- All other objects that use the IEnumerable<T> interface.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.FindAllByValue Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Finds all the data points that are equal to the specified value.

Overloads

FindAllByValue(Double, String, Int32)	Finds all the data points that are equal to the specified value, starting from the specified index location.
FindAllByValue(Double, String)	Finds all the data points that are equal to the specified value.
FindAllByValue(Double)	Finds all the data points that are equal to the specified value.

FindAllByValue(Double, String, Int32)

Finds all the data points that are equal to the specified value, starting from the specified index location.

C#

```
public
System.Collections.Generic.IEnumerable<System.Windows.Forms.DataVisualiza
tion.Charting.DataPoint> FindAllByValue (double valueToFind, string
useValue, int startIndex);
```

Parameters

`valueToFind` Double

The data point value to find.

`useValue` String

The data point value to use (X, Y1, Y2...).

`startIndex` Int32

The index location of the data point to start searching from.

Returns

IEnumerable<DataPoint>

An IEnumerable<T> interface that represents the enumerator of the matching data points.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FindAllByValue(Double, String)

Finds all the data points that are equal to the specified value.

C#

```
public
System.Collections.Generic.IEnumerable<System.Windows.Forms.DataVisualiza
tion.Charting.DataPoint> FindAllByValue (double valueToFind, string
useValue);
```

Parameters

`valueToFind` Double

The data point value to find.

`useValue` String

The data point value to use (X, Y1, Y2...).

Returns

IEnumerable<DataPoint>

An IEnumerable<T> interface that represents the enumerator of the matching data points.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FindAllByValue(Double)

Finds all the data points that are equal to the specified value.

C#

```
public
System.Collections.Generic.IEnumerable<System.Windows.Forms.DataVisualiza
tion.Charting.DataPoint> FindAllByValue (double valueToFind);
```

Parameters

`valueToFind` Double

The data point value to find.

Returns

IEnumerable<DataPoint>

An IEnumerable<T> interface that represents the enumerator of the matching data points.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.FindByValue Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Finds the first data point that is equal to the specified value.

Overloads

FindByValue(Double)	Finds the first data point that is equal to the specified value.
FindByValue(Double, String)	Finds the first data point that is equal to the specified value.
FindByValue(Double, String, Int32)	Finds the first data point that is equal to the specified value, starting from the specified index location.

FindByValue(Double)

Finds the first data point that is equal to the specified value.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindByValue (double valueToFind);
```

Parameters

`valueToFind` Double

The data point value to find.

Returns

DataPoint

The DataPoint that matches the specified value, or `null` if there is no match.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FindByValue(Double, String)

Finds the first data point that is equal to the specified value.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindByValue (double valueToFind, string useValue);
```

Parameters

`valueToFind` Double

The data point value to find.

`useValue` String

The data point value to use (X, Y1, Y2...).

Returns

DataPoint

The DataPoint that matches the specified value, or `null` if there is no match.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FindByValue(Double, String, Int32)

Finds the first data point that is equal to the specified value, starting from the specified index location.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindByValue (double valueToFind, string useValue, int startIndex);
```

Parameters

`valueToFind` Double

The data point value to find.

`useValue` String

The data point value to use (X, Y1, Y2...).

`startIndex` Int32

The index location of the data point to start searching from.

Returns

DataPoint

The DataPoint that matches the specified value, or `null` if there is no match.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.FindMaxByValue Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Finds the data point with the maximum value.

Overloads

FindMaxByValue()	Finds the data point with the maximum value.
FindMaxByValue(String)	Finds the data point value with the maximum value.
FindMaxByValue(String, Int32)	Finds the data point with the maximum value, starting from the specified index location.

FindMaxByValue()

Finds the data point with the maximum value.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindMaxByValue ();
```

Returns

DataPoint

The DataPoint object with the maximum value.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FindMaxByValue(String)

Finds the data point value with the maximum value.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindMaxByValue (string useValue);
```

Parameters

`useValue` String

The data point value to use (X, Y1, Y2...).

Returns

DataPoint

The DataPoint object with the maximum value.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FindMaxByValue(String, Int32)

Finds the data point with the maximum value, starting from the specified index location.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindMaxByValue (string useValue, int startIndex);
```

Parameters

`useValue` String

The data point value to use (X, Y1, Y2...).

`startIndex` Int32

The index location of the data point to start searching from.

Returns

DataPoint

The DataPoint object with the maximum value.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.FindMinByValue Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Finds the data point with the minimum value.

Overloads

FindMinByValue()	Finds the data point with the minimum value.
FindMinByValue(String)	Finds the data point with the minimum value.
FindMinByValue(String, Int32)	Finds the data point with the minimum value, starting from the specified index location.

FindMinByValue()

Finds the data point with the minimum value.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindMinByValue ();
```

Returns

DataPoint

The DataPoint object with the minimum value.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FindMinByValue(String)

Finds the data point with the minimum value.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindMinByValue (string useValue);
```

Parameters

`useValue` String

The data point value to use (X, Y1, Y2...).

Returns

DataPoint

A DataPoint object with the minimum value.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FindMinByValue(String, Int32)

Finds the data point with the minimum value, starting from the specified index location.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DataPoint
FindMinByValue (string useValue, int startIndex);
```

Parameters

`useValue` String

The data point value to use (X, Y1, Y2...).

`startIndex` Int32

The index location of the data point to start searching from.

Returns

DataPoint

The DataPoint object with the minimum value.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.InsertXY(Int32, Object, Object[]) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Inserts a data point with the specified X value and one or more specified Y values.

C#

```
public void InsertXY (int index, object xValue, params object[] yValue);
```

Parameters

`index` Int32

The index at which the data point will be inserted.

`xValue` Object

The X value of the data point.

`yValue` Object[]

A comma-separated list of Y values of the data point.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCollection.InsertY(Int32, Object[]) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Inserts a data point with one or more specified Y values.

C#

```
public void InsertY (int index, params object[] yValue);
```

Parameters

index Int32

The index at which the data point will be inserted.

yValue Object[]

A comma-separated list of Y values of the data point.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointComparer Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Compares data points.

C#

```
public class DataPointComparer :
System.Collections.Generic.IComparer<System.Windows.Forms.DataVisualization.
Charting.DataPoint>
```

Inheritance Object → DataPointComparer

Implements IComparer<DataPoint>

Constructors

DataPointComparer(Series, PointSortOrder, String)	Initializes a new instance of the DataPointComparer class with the specified data series, sort order and sort value.

Methods

Compare(DataPoint, DataPoint)	Compares two data points.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object.

	(Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointComparer(Series, PointSortOrder, String) Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the DataPointComparer class with the specified data series, sort order and sort value.

C#

```
public DataPointComparer
(System.Windows.Forms.DataVisualization.Charting.Series series,
System.Windows.Forms.DataVisualization.Charting.PointSortOrder sortOrder,
string sortBy);
```

Parameters

series Series

Data series.

sortOrder PointSortOrder

Sort order.

sortBy String

Value used for the sort, for example, "X", "Y or Y1", "Y2", and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointComparer.Compare(DataPoint, DataPoint) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Compares two data points.

C#

```
public int Compare
(System.Windows.Forms.DataVisualization.Charting.DataPoint x,
System.Windows.Forms.DataVisualization.Charting.DataPoint y);
```

Parameters

x DataPoint

First data point.

y DataPoint

Second data point.

Returns

Int32

An `integer` value. Returns zero (0) if the two values are equal. Returns a positive integer if the value of `x` is greater than the value of `y`; otherwise, returns a negative integer.

Implements

Compare(T, T)

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Defines custom properties that are common to data points, series and empty points.

C#

```
[System.ComponentModel.TypeConverter("System.Windows.Forms.Design.DataVisual
ization.Charting.DataPointCustomPropertiesConverter,
System.Windows.Forms.DataVisualization.Design, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=31bf3856ad364e35")]
public class DataPointCustomProperties :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → DataPointCustomProperties

Derived System.Windows.Forms.DataVisualization.Charting.DataPoint
System.Windows.Forms.DataVisualization.Charting.Series

Attributes TypeConverterAttribute

Remarks

This class defines custom properties such as colors, the use of background images, hatching, and so forth.

The Series class, DataPoint class and the EmptyPointStyle property are all derived from this class.

Constructors

DataPointCustomProperties()	Initializes a new instance of the DataPointCustomProperties class.
DataPointCustom	Initializes a new instance of the DataPointCustomProperties class

Properties(Series, Boolean)	with the specified series and custom properties flag.

Properties

AxisLabel	Gets or sets the text of the X-axis label for the data point, series or an empty point. This property is only used if a custom label has not been specified for the relevant Axis object.
BackGradientStyle	Gets or sets the background gradient style.
BackHatchStyle	Gets or sets the background hatching style.
BackImage	Gets or sets the background image of the data point.
BackImageAlignment	Gets or sets the alignment of the background image, which is used with the Unscaled drawing mode.
BackImageTransparentColor	Gets or sets a color that will be replaced with a transparent color when the background image is drawn.
BackImageWrapMode	Gets or sets the drawing mode of the background image.
BackSecondaryColor	Gets or sets the secondary background color.
BorderColor	Gets or sets the border color of the data point.
BorderDashStyle	Gets or sets the border style of the data point.
BorderWidth	Gets or sets the border width of the data point.
Color	Gets or sets the color of the data point.
CustomProperties	Gets or sets the custom properties of the data point. Custom properties can be specified in the following format: AttrName1=Value1, AttrName2=Value2 ...
CustomPropertiesExtended	Gets or sets the extended custom properties of the data point. Extended custom properties can be specified in the following format: AttrName1=Value1, AttrName2=Value2...
Font	Gets or sets the font of the data point.
IsValueShownAsLabel	Gets or sets a flag that indicates whether to show the value of the data point on the label.
IsVisibleInLegend	Gets or sets a flag that indicates whether the item is shown in the legend.
Item[Int32]	Gets the DataPointCustomProperties object by its position in the index.

Item[String]	Gets or sets the DataPointCustomProperties object by name.
Label	Gets or sets the text of the data point label.
LabelAngle	Gets or sets the angle of the data point label.
LabelBackColor	Gets or sets the background color of the data point label.
LabelBorderColor	Gets or sets the border color of the data point label.
LabelBorderDashStyle	Gets or sets the border style of the label.
LabelBorderWidth	Gets or sets the border width of the label.
LabelForeColor	Gets or sets the text color of the label.
LabelFormat	Gets or sets the format of the data point label.
LabelToolTip	Gets or sets the tooltip for the data point label.
LegendText	Gets or sets the text of the item in the legend.
LegendToolTip	Gets or sets the tooltip of the item in the legend.
MarkerBorderColor	Gets or sets the border color of the marker.
MarkerBorderWidth	Gets or sets the border width of the marker.
MarkerColor	Gets or sets the marker color.
MarkerImage	Gets or sets the marker image.
MarkerImageTransparentColor	Gets or sets the color that will be replaced with a transparent color when the marker image is drawn.
MarkerSize	Gets or sets the size of the marker.
MarkerStyle	Gets or sets the marker style.
Name	Gets or sets the name of the chart element. (Inherited from ChartNamedElement)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
ToolTip	Gets or sets the tooltip.

Methods

DeleteCustomProperty(String)	Deletes the data point custom property with the specified name.
Dispose()	Releases the resources used by the ChartElement.

	(Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement and optionally releases the managed resources. (Inherited from ChartElement)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetCustomProperty(String)	Returns the data point custom property with the specified name.
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IsCustomPropertySet(String)	Determines whether a custom property with the specified name was set.
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResetIsValueShownAsLabel()	Resets a custom property to its default value.
ResetIsVisibleInLegend()	Resets a custom property to its default value.
SetCustomProperty(String, String)	Sets a custom property of the data point.
SetDefault(Boolean)	Sets the default properties of the data point. Indicates whether the previous property values must be cleared.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the DataPointCustomProperties class.

Overloads

DataPointCustomProperties()	Initializes a new instance of the DataPointCustomProperties class.
DataPointCustomProperties(Series, Boolean)	Initializes a new instance of the DataPointCustomProperties class with the specified series and custom properties flag.

DataPointCustomProperties()

Initializes a new instance of the DataPointCustomProperties class.

C#

```
public DataPointCustomProperties ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties(Series, Boolean)

Initializes a new instance of the DataPointCustomProperties class with the specified series and custom properties flag.

C#

```
public DataPointCustomProperties
(System.Windows.Forms.DataVisualization.Charting.Series series, bool
pointProperties);
```

Parameters

series Series

The series that the data point belongs to.

pointProperties Boolean

A flag that indicates whether this is a data point custom property.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.AxisLabel Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text of the X-axis label for the data point, series or an empty point. This property is only used if a custom label has not been specified for the relevant Axis object.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual string AxisLabel { get; set; }
```

Property Value

String

A `string` value that represents the text of the X-axis label.

Attributes BindableAttribute

Remarks

This property sets the axis label for a data point, series or an empty point.

This property can only be set at run time for a Series object. If you set this property or a Series object, all of its data points will use this label.

If the AxisLabel property is not set and a custom label has not been specified, an axis label will be generated automatically using the axis scale.

Empty point labels are set using the EmptyPointStyle property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background gradient style.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration value that is used to draw the background gradient style.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background hatching style.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration value that is used to draw the background hatching style.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BackImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image of the data point.

C#

```
[System.ComponentModel.Bindable(true)]
public string BackImage { get; set; }
```

Property Value

String

A `string` value that represents the name of the background image file.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BackImageAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of the background image, which is used with the Unscaled drawing mode.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.ChartImageAlignmentStyle
BackImageAlignment { get; set; }
```

Property Value

ChartImageAlignmentStyle

A ChartImageAlignmentStyle enumeration value that specifies the alignment of the background image.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BackImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color that will be replaced with a transparent color when the background image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that will be replaced with a transparent color when the image is drawn.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BackImageWrapMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the drawing mode of the background image.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartImageWrapMode
BackImageWrapMode { get; set; }
```

Property Value

ChartImageWrapMode

A ChartImageWrapMode enumeration value that defines the drawing mode of the image.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary background color.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value that specifies the secondary background color.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of the data point.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderColor { get; set; }
```

Property Value

Color

A Color value that specifies the border color of the data point.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border style of the data point.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that specifies the border style of the data point.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.BorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of the data point.

C#

```
[System.ComponentModel.Bindable(true)]
public int BorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that specifies the border width of the data point.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.Color Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the data point.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color Color { get; set; }
```

Property Value

Color

A Color value that is used to draw the color of the data point.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.CustomProperties Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the custom properties of the data point. Custom properties can be specified in the following format: AttrName1=Value1, AttrName2=Value2 ...

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public string CustomProperties { get; set; }
```

Property Value

String

A `string` value that represents the custom properties of the data point.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.CustomPropertiesExtended Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the extended custom properties of the data point. Extended custom properties can be specified in the following format: AttrName1=Value1, AttrName2=Value2...

C#

```
[System.ComponentModel.Bindable(false)]
public System.Windows.Forms.DataVisualization.Charting.CustomProperties
CustomPropertiesExtended { get; set; }
```

Property Value

CustomProperties

A CustomProperties object that represents the extended custom properties of the data point.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font of the data point.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font value that represents the font of the data point.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.IsValueShownAsLabel Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether to show the value of the data point on the label.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsValueShownAsLabel { get; set; }
```

Property Value

Boolean

`true` if the value of the data point is shown on the label; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.IsVisibleInLegend Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the item is shown in the legend.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsVisibleInLegend { get; set; }
```

Property Value

Boolean

`true` if the item is shown in the legend; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.Item[] Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the DataPointCustomProperties object.

Overloads

Item[String]	Gets or sets the DataPointCustomProperties object by name.
Item[Int32]	Gets the DataPointCustomProperties object by its position in the index.

Item[String]

Gets or sets the DataPointCustomProperties object by name.

```
C#

public string this[string name] { get; set; }
```

Parameters

`name` String

The name of the custom property.

Property Value

String

A `string` value that represents the custom properties of the data point.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Item[Int32]

Gets the DataPointCustomProperties object by its position in the index.

```
C#
public string this[int index] { get; }
```

Parameters

`index` Int32

The index of the custom property.

Property Value

String

A `string` value that represents the custom properties of the data point.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.Label Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text of the data point label.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual string Label { get; set; }
```

Property Value

String

A `string` value that represents the text of the data point label.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LabelAngle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the angle of the data point label.

C#

```
[System.ComponentModel.Bindable(true)]
public int LabelAngle { get; set; }
```

Property Value

Int32

An `integer` value that represents the angle of the label.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LabelBackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the data point label.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color LabelBackColor { get; set; }
```

Property Value

Color

A Color value that represents the background color of the data point label.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LabelBorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of the data point label.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color LabelBorderColor { get; set; }
```

Property Value

Color

A Color value that represents the border color of the data point label.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LabelBorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border style of the label.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
LabelBorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that specifies the border style of the label.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LabelBorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of the label.

C#

```
[System.ComponentModel.Bindable(true)]
public int LabelBorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the width of the border.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LabelForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color of the label.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color LabelForeColor { get; set; }
```

Property Value

Color

A Color value that represents the text color of the label.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LabelFormat Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the format of the data point label.

C#

```
[System.ComponentModel.Bindable(true)]
public string LabelFormat { get; set; }
```

Property Value

String

A `string` value that represents the format of the label.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LabelToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip for the data point label.

C#

```
[System.ComponentModel.Bindable(true)]
public string LabelToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip for the data point label.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LegendText Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text of the item in the legend.

C#

```
[System.ComponentModel.Bindable(true)]
public string LegendText { get; set; }
```

Property Value

String

A `string` value that represents the text of the item in the legend.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.LegendToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip of the item in the legend.

C#

```
[System.ComponentModel.Bindable(true)]
public string LegendToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip of the item in the legend.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.MarkerBorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of the marker.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color MarkerBorderColor { get; set; }
```

Property Value

Color

A Color value that represents the border color of the marker.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.MarkerBorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of the marker.

C#

```
[System.ComponentModel.Bindable(true)]
public int MarkerBorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that specifies the border width of the marker.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.MarkerColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the marker color.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color MarkerColor { get; set; }
```

Property Value

Color

A Color value that is used to draw the marker color.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.MarkerImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the marker image.

C#

```
[System.ComponentModel.Bindable(true)]
public string MarkerImage { get; set; }
```

Property Value

String

A `string` value that represents the name of the marker image file.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.MarkerImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color that will be replaced with a transparent color when the marker image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color MarkerImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that specifies the color that will be replaced with a transparent color when the marker image is drawn.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.MarkerSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the marker.

C#

```
[System.ComponentModel.Bindable(true)]
public int MarkerSize { get; set; }
```

Property Value

Int32

An `integer` value that specifies the size of the marker.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.MarkerStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the marker style.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.MarkerStyle
MarkerStyle { get; set; }
```

Property Value

MarkerStyle

A MarkerStyle enumeration value that specifies the marker style.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip.

C#

```
[System.ComponentModel.Bindable(true)]
public string ToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.DeleteCustomProperty(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Deletes the data point custom property with the specified name.

C#

```
public virtual void DeleteCustomProperty (string name);
```

Parameters

name String

The name of the property to delete.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.GetCustomProperty(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the data point custom property with the specified name.

C#

```
public virtual string GetCustomProperty (string name);
```

Parameters

name String

The name of the property to return.

Returns

String

A `string` value that represents the data point custom property with the specified name. If the requested property is not set, the default custom property of the data series will be returned.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.IsCustomPropertySet(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Determines whether a custom property with the specified name was set.

C#

```
public virtual bool IsCustomPropertySet (string name);
```

Parameters

name String

The name of the custom property to check.

Returns

Boolean

`true` if the custom property was set; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.ResetIsValueShownAsLabel Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Resets a custom property to its default value.

C#

```
public void ResetIsValueShownAsLabel ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.ResetIsVisibleInLegend Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Resets a custom property to its default value.

C#

```
public void ResetIsVisibleInLegend ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.SetCustomProperty(String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets a custom property of the data point.

C#

```
public virtual void SetCustomProperty (string name, string propertyValue);
```

Parameters

name String

The name of the property to set.

propertyValue String

The value of the property to set.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DataPointCustomProperties.SetDefault(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Sets the default properties of the data point. Indicates whether the previous property values must be cleared.

C#

```
public virtual void SetDefault (bool clearAll);
```

Parameters

`clearAll` Boolean

A flag that indicates whether to clear previous property values.

Remarks

If the `clearAll` parameter is set to `true`, all previous property values are cleared, otherwise they are not.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DateRangeType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a date/time interval that determines the ranges that will be filtered when you invoke one of the `Filter` method definitions found in the DataManipulator class.

C#

```
public enum DateRangeType
```

Inheritance Object → ValueType → Enum → DateRangeType

Fields

DayOfMonth	3	Date/time interval is days of the month.
DayOfWeek	2	Date/time interval is days of the week.
Hour	4	Date/time interval is hours.
Minute	5	Date/time interval is minutes.
Month	1	Date/time interval is months.
Year	0	Date/time interval is years.

Remarks

The DateRangeType enumeration is passed in the `DateRange` parameter of some `Filter` methods, which combined with the `rangeElements` parameter determines the date/time ranges that are filtered.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

DateTimeIntervalType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies an interval type.

C#

```
public enum DateTimeIntervalType
```

Inheritance Object → ValueType → Enum → DateTimeIntervalType

Fields

Auto	0	Automatically determined by the Chart control.
Days	5	Interval type is in days.
Hours	6	Interval type is in hours.
Milliseconds	9	Interval type is in milliseconds.
Minutes	7	Interval type is in minutes.
Months	3	Interval type is in months.
NotSet	10	The IntervalType or IntervalOffsetType property is not set. This value is used for grid lines, tick marks, strip lines and axis labels, and indicates that the interval type is being obtained from the Axis object to which the element belongs. Setting this value for an Axis object will have no effect.
Number	1	Interval type is in numerical.
Seconds	8	Interval type is in seconds.
Weeks	4	Interval type is in weeks.
Years	2	Interval type is in years.

Remarks

The DateTimeIntervalType enumeration represents date and time interval types.

Note that if you use the NotSet value, properties other than IntervalType or IntervalOffsetType will treat NotSet as an Auto value.

The DateTimeIntervalType values are applicable to the LabelStyle, Grid, TickMark and StripLine classes.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Docking Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies where a chart element, such as a legend or title, will be docked on the chart.

C#

```
public enum Docking
```

Inheritance Object → ValueType → Enum → Docking

Fields

Bottom	2	Docked to the bottom of either the chart image or a ChartArea object.
Left	3	Docked to the left of either the chart image or a ChartArea object.
Right	1	Docked to the right of either the chart image or a ChartArea object.
Top	0	Docked to the top of either the chart image or a ChartArea object.

Remarks

The Docking enumeration represents the docking location on the Chart control that an chart element should be docked.

Set the `IsDockedInsideChartArea` property of the relevant chart element to specify whether that element will be docked to the entire chart image or to a ChartArea object. The chart area is specified in the DockedToChartArea property of the chart element.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the base class for many visual elements of the chart such as the legend, title, and chart areas. Defines the position of the chart element in relative coordinates, which range from (0,0) to (100,100).

C#

```
public class ElementPosition :
System.Windows.Forms.DataVisualization.Charting.ChartElement
```

Inheritance Object → ChartElement → ElementPosition

Remarks

The ElementPosition class is used to define the position of a chart element, and represents one rectangle using the X, Y, Width and Height properties.

The ElementPosition class is always exposed as a `Position` property, such as Position, Position, and so forth. It is also exposed in the PrePaint and PostPaint events of the Chart control as a property of the ChartPaintEventArgs object.

The position of the applicable chart elements can be set automatically by setting the Auto property to `true`; chart elements can also be positioned at design time in the Design View.

There are two major differences between this class and a RectangleF structure:

- This class can use automatic positioning.
- This class only uses coordinates ranging from 0 to 100, otherwise an exception is thrown.

The position of chart elements is described using relative coordinates, with the top-left corner of an element having coordinates of (0,0) and the bottom-right corner of an

element having coordinates of (100,100).

Elements that are plotted within a chart area use coordinates that are relative to the chart area, whereas elements outside a chart area use coordinates that are relative to the chart image.

Constructors

ElementPosition()	Initializes a new instance of the ElementPosition class.
ElementPosition(Single, Single, Single, Single)	Initializes a new instance of the ElementPosition class with the specified `x`, `y`, `width` and `height` parameters.

Properties

Auto	Gets or sets a value that indicates whether an applicable chart element will be positioned automatically by the Chart control.
Bottom	Gets the bottom position of a chart element, in relative coordinates.
Height	Gets or sets the height of a chart element.
Right	Gets the position of the right side of a chart element, in relative coordinates.
Size	Gets the size of a chart element.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Width	Gets or sets the width of a chart element.
X	Gets or sets the relative X-coordinate of the top-left corner of an applicable chart element.
Y	Gets or sets the relative Y-coordinate of the top-left corner of an applicable chart element.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElement

	and optionally releases the managed resources. (Inherited from ChartElement)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
FromRectangleF(RectangleF)	Initializes an ElementPosition object from a RectangleF structure.
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToRectangleF()	Returns a RectangleF object that uses the coordinates of an ElementPosition object.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ElementPosition class.

Overloads

ElementPosition()	Initializes a new instance of the ElementPosition class.
ElementPosition(Single, Single, Single, Single)	Initializes a new instance of the ElementPosition class with the specified `x`, `y`, `width` and `height` parameters.

ElementPosition()

Initializes a new instance of the ElementPosition class.

C#

```
public ElementPosition ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition(Single, Single, Single, Single)

Initializes a new instance of the ElementPosition class with the specified `x`, `y`, `width` and `height` parameters.

```C#
public ElementPosition (float x, float y, float width, float height);
```

Parameters

`x` Single

The X position of the top-left corner of the chart control.

`y` Single

The Y position of the top-left corner of the chart control.

`width` Single

The width of the chart element.

`height` Single

The height of the chart element.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.Auto Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that indicates whether an applicable chart element will be positioned automatically by the Chart control.

C#

```
[System.ComponentModel.Bindable(true)]
public bool Auto { get; set; }
```

Property Value

Boolean

`true` if an applicable chart element will be positioned automatically by the chart control; otherwise, `false`.

Attributes BindableAttribute

Remarks

Use this property to automatically position a chart element. For example, to position the legend automatically, set the Position property to Auto.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.Bottom Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the bottom position of a chart element, in relative coordinates.

C#

```
[System.ComponentModel.Browsable(false)]
public float Bottom { get; }
```

Property Value

Single

A `float` value that represents the relative position of the bottom of a chart element.

Attributes BrowsableAttribute

Remarks

The bottom of the chart element is always expressed in relative coordinates, and is either relative to a ChartArea object if plotted within a chart area, or relative to the entire chart image if plotted outside of a chart area element.

Chart coordinates use (0,0) for a top-left corner, and (100,100) for a bottom-right corner.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.Height Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the height of a chart element.

C#

```
[System.ComponentModel.Bindable(true)]
public float Height { get; set; }
```

Property Value

Single

A `float` value that represents the height of the chart element.

Attributes BindableAttribute

Remarks

Always uses relative coordinates. If you set a value outside of the 0-100 pixel range, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.Right Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the position of the right side of a chart element, in relative coordinates.

C#

```
[System.ComponentModel.Browsable(false)]
public float Right { get; }
```

Property Value

Single

A `float` value that represents the relative position of the right side of a chart element.

Attributes BrowsableAttribute

Remarks

Returns a value that represents the relative position of the right side of an applicable chart element.

The right side position is always expressed in relative coordinates, and is either relative to a ChartArea object if plotted within a chart area, or relative to the entire chart image if plotted outside of a chart area element.

Chart coordinates use (0,0) for a top-left corner, and (100,100) for a bottom-right corner.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.Size Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the size of a chart element.

C#

```
[System.ComponentModel.Browsable(false)]
public System.Drawing.SizeF Size { get; }
```

Property Value

SizeF

A SizeF value that represents the size of a chart element.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.Width Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of a chart element.

C#

```
[System.ComponentModel.Bindable(true)]
public float Width { get; set; }
```

Property Value

Single

A `float` value that represents the width of a chart element.

Attributes BindableAttribute

Remarks

This property always uses relative coordinates; if you set a value outside the 0-100 pixel range, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.X Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the relative X-coordinate of the top-left corner of an applicable chart element.

C#

```
[System.ComponentModel.Bindable(true)]
public float X { get; set; }
```

Property Value

Single

A `float` value that represents the X-coordinate of the top-left corner of an applicable chart element.

Attributes BindableAttribute

Remarks

The X property is always expressed in relative coordinates, and is either relative to a ChartArea object if plotted within a chart area, or relative to the entire chart image if plotted outside of a chart area element.

This property always uses relative coordinates; if you set a value outside the 0-100 pixel range, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.Y Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the relative Y-coordinate of the top-left corner of an applicable chart element.

C#

```
[System.ComponentModel.Bindable(true)]
public float Y { get; set; }
```

Property Value

Single

A `float` value that represents the Y-coordinate of the top-left corner of an applicable chart element.

Attributes BindableAttribute

Remarks

The Y property is always expressed in relative coordinates, and is either relative to a ChartArea object if plotted within a chart area, or relative to the entire chart image if plotted outside of a chart area element.

This property always uses relative coordinates; if you set a value outside the 0-100 pixel range, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.FromRectangleF(RectangleF) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes an ElementPosition object from a RectangleF structure.

C#

```
public void FromRectangleF (System.Drawing.RectangleF rect);
```

Parameters

`rect` RectangleF

A RectangleF structure used to initialize the element position object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ElementPosition.ToRectangleF Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns a RectangleF object that uses the coordinates of an ElementPosition object.

C#

```
public System.Drawing.RectangleF ToRectangleF ();
```

Returns

RectangleF

A RectangleF structure that represents the coordinates of an element position object.

Remarks

This method converts the position of an element to a RectangleF structure.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

EllipseAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents an ellipse annotation.

C#

```
public class EllipseAnnotation :
System.Windows.Forms.DataVisualization.Charting.RectangleAnnotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → TextAnnotation → RectangleAnnotation → EllipseAnnotation

Remarks

An ellipse annotation can display text inside the ellipse, in addition to being displayed as an ellipse only. You can therefore annotate using an ellipse or an ellipse with text inside it.

Constructors

EllipseAnnotation()	Initializes a new instance of the EllipseAnnotation class.

Properties

Alignment	Gets or sets the content alignment of an annotation. (Inherited from Annotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)

AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point. (Inherited from Annotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the annotation type name.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)

AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	Gets or sets the background color of the annotation. (Inherited from RectangleAnnotation)
BackGradientStyle	Gets or sets the gradient style of the annotation. (Inherited from RectangleAnnotation)
BackHatchStyle	Gets or sets the hatch style of an annotation background. (Inherited from RectangleAnnotation)
BackSecondaryColor	Gets or sets the secondary color for an annotation background. (Inherited from RectangleAnnotation)
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
Font	Gets or sets the font for the annotation text. (Inherited from TextAnnotation)
ForeColor	Gets or sets the text color of an annotation. (Inherited from Annotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsMultiline	Gets or sets a flag that indicates whether the annotation text is multiline. (Inherited from TextAnnotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	Gets or sets the line color of an annotation. (Inherited from RectangleAnnotation)

LineDashStyle	Gets or sets the line style of an annotation. (Inherited from RectangleAnnotation)
LineWidth	Gets or sets the line width, in pixels, of an annotation. (Inherited from RectangleAnnotation)
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the text of the annotation. (Inherited from TextAnnotation)
TextStyle	Gets or sets the text style of an annotation. (Inherited from Annotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)
BeginTextEditing()	Begins editing of the annotation text by an end-user. (Inherited from TextAnnotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends user placement of an annotation. (Inherited from TextAnnotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
StopTextEditing()	Stops end-user editing of the annotation text. (Inherited from TextAnnotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

EllipseAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the EllipseAnnotation class.

C#

```
public EllipseAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

EllipseAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type name.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

This abstract property is used to get the name of each annotation type, such as a LineAnnotation, RectangleAnnotation, or EllipseAnnotation.

This property is for internal use and should not be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FinancialFormula Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a financial formula.

C#

```
public enum FinancialFormula
```

Inheritance Object → ValueType → Enum → FinancialFormula

Fields

AccumulationDistribution	0	Accumulation Distribution formula uses a relationship between volume and prices to estimate the strength of price movements; if volume is increased, there is a high probability that prices will go up.
AverageTrueRange	1	Average True Range indicator measures commitment and compares the range between the High, Low and Close prices.
BollingerBands	2	Bollinger Bands indicators are plotted at standard deviation levels above and below a simple moving average.
ChaikinOscillator	3	Chaikin Oscillator indicator is the difference between a 3-day exponential moving average and a 10-day exponential moving average applied to the accumulation distribution.
CommodityChannelIndex	4	Commodity Channel Index compares prices with their moving averages.
DetrendedPriceOscillator	5	Detrended Price Oscillator attempts to remove trends from prices.
EaseOfMovement	6	Ease of Movement deals with the relationship

		between volume and price change, and uses volume to indicate how strong a trend is for prices.
Envelopes	7	Envelopes are plotted above and below a moving average using a specified percentage as the shift.
ExponentialMovingAverage	8	An Exponential Moving Average is an average of data calculated over a period of time where the most recent days have more weight.
Forecasting	9	Forecasting predicts future values using historical observations.
MassIndex	11	The Mass Index is used to predict trend reversal by comparing the difference and range between High and Low prices.
MedianPrice	12	Median prices are mid-point values of daily prices and can be used as a filter for trend indicators.
MoneyFlow	13	The Money Flow indicator compares upward changes and downward changes of volume-weighted typical prices.
MovingAverage	21	A Simple Moving Average is an average of data calculated over a period of time. The moving average is the most popular price indicator used in technical analysis, and can be used with any price, for example, Hi, Low, Open and Close, or can be applied to other indicators.
MovingAverageConvergenceDivergence	10	The Moving Average Convergence/Divergence indicator compares two moving averages of prices and is used with a 9-day Exponential Moving average as a signal that indicates buying and selling moments.
NegativeVolumeIndex	14	The Negative Volume Index should be used with the Positive Volume index; the Negative Volume Index only changes if the volume decreases from the previous day.
OnBalanceVolume	15	The On Balance Volume indicator measures positive and negative volume flow.
Performance	16	The Performance indicator compares a current closing price or any other price with the first closing value from the first time period.
PositiveVolumeIndex	17	The Positive Volume Index should be used with the Negative Volume index. The Positive volume

		index only changes if the volume decreases from the previous day.
PriceVolumeTrend	18	The Price Volume Trend is a cumulative volume total that is calculated using relative changes of the closing price, and should be used with other indicators.
RateOfChange	19	The Rate of Change indicator compares a specified closing price with the current price.
RelativeStrengthIndex	20	The Relative Strength Index is a momentum oscillator that compares upward movements of the closing price with downward movements, and results in values that range from 0 to 100.
StandardDeviation	22	Standard Deviation is used to indicate volatility, and measures the difference between values, for example, closing price, and their moving average.
StochasticIndicator	23	The Stochastic Indicator helps to find trend reversal by searching in a period for when the closing prices are close to low prices in an upward trending market and when the closing prices are close to high prices in a downward trending market.
TriangularMovingAverage	24	A Triangular Moving Average is an average of data calculated over a period of time where the middle portion of data has more weight.
TripleExponentialMovingAverage	25	The Triple Exponential Moving Average is based on a triple moving average of the closing price. Its purpose is to eliminate short cycles. This indicator keeps the closing price in trends that are shorter than the specified period.
TypicalPrice	26	Typical Price is the average value of daily prices, and can be used as a filter for trend indicators.
VolatilityChaikins	27	The Volatility Chaikins indicator measures the difference between High and Low prices, and is used to indicate tops or bottoms of the market.
VolumeOscillator	28	The Volume Oscillator attempts to identify trends in volume by comparing two moving averages: one with a short period and another with a longer period.
WeightedClose	29	The Weighted Close formula calculates the average value of daily prices. The only difference between Typical Price and the Weighted Close is

		that the closing price has extra weight, and is considered the most important price.
WeightedMovingAverage	30	Weighted Moving Average is an average of data that is calculated over a period of time, where greater weight is attached to the most recent data.
WilliamsR	31	Williams %R is a momentum indicator, and is used to measure overbought and oversold levels.

Remarks

The FinancialFormula enumeration is used in calls to the `FinancialFormula` methods that are contained in the DataFormula class, and specifies the type of financial formula to be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FormatNumberEventArgs Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the event arguments of the localized number-formatting event.

C#

```
public class FormatNumberEventArgs : EventArgs
```

Inheritance Object → EventArgs → FormatNumberEventArgs

Properties

ElementType	Gets the chart element type.
Format	Gets or sets the format string.
LocalizedValue	Gets or sets the localized text.
SenderTag	Gets the sender object of the event.
Value	Gets the value to be formatted.
ValueType	Gets the value type.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object.

	(Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FormatNumberEventArgs.ElementType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the chart element type.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartElementType
ElementType { get; }
```

Property Value

ChartElementType

A ChartElementType enumeration value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FormatNumberEventArgs.Format Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the format string.

C#

```
public string Format { get; }
```

Property Value

String

A `string` value that represents the format.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FormatNumberEventArgs.LocalizedValue Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the localized text.

C#

```
public string LocalizedValue { get; set; }
```

Property Value

String

A `string` value that represents the localized text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FormatNumberEventArgs.SenderTag Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the sender object of the event.

C#

```
public object SenderTag { get; }
```

Property Value

Object

The sender object of the event.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FormatNumberEventArgs.Value Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the value to be formatted.

C#

```
public double Value { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FormatNumberEventArgs.ValueType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the value type.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartValueType
ValueType { get; }
```

Property Value

ChartValueType

A ChartValueType enumeration value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Stores results of the FTest statistical methods.

C#

```
public class FTestResult
```

Inheritance Object → FTestResult

Constructors

FTestResult()	Initializes a new instance of the FTestResult class.

Properties

FCriticalValueOneTail	Gets the Critical F one tail.
FirstSeriesMean	Gets the first series mean.
FirstSeriesVariance	Gets the first series variance.
FValue	Gets the F value.
ProbabilityFOneTail	Gets the Probability F one tail.
SecondSeriesMean	Gets the second series mean.
SecondSeriesVariance	Gets the second series variance.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object.

	(Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the FTestResult class.

C#

```
public FTestResult ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult.FCriticalValueOneTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Critical F one tail.

C#

```
public double FCriticalValueOneTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult.FirstSeriesMean Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the first series mean.

C#

```
public double FirstSeriesMean { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult.FirstSeriesVariance Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the first series variance.

C#

```
public double FirstSeriesVariance { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult.FValue Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the F value.

C#

```
public double FValue { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult.ProbabilityFOneTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Probability F one tail.

C#

```
public double ProbabilityFOneTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult.SecondSeriesMean Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the second series mean.

C#

```
public double SecondSeriesMean { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

FTestResult.SecondSeriesVariance Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the second series variance.

C#

```
public double SecondSeriesVariance { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

GradientStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a gradient style to be used for a chart element.

C#

```
public enum GradientStyle
```

Inheritance Object → ValueType → Enum → GradientStyle

Fields

Center	3	Gradient is applied from the center outwards.
DiagonalLeft	4	Gradient is applied diagonally from left to right.
DiagonalRight	5	Gradient is applied diagonally from right to left.
HorizontalCenter	6	Gradient is applied horizontally from the center outwards.
LeftRight	1	Gradient is applied from left to right.
None	0	No gradient is applied.
TopBottom	2	Gradient is applied from top to bottom.
VerticalCenter	7	Gradient is applied vertically from the center outwards.

Remarks

The GradientStyle enumeration represents a gradient style, and is used with the BackGradientStyle property of the Chart.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides methods and properties that manage the major and minor grids of Axis objects.

```
C#
public class Grid
```

Inheritance Object → Grid

Derived System.Windows.Forms.DataVisualization.Charting.TickMark

Remarks

This class provides the functionality for the grids on the chart, and is exposed using the MajorGrid and MinorGrid properties of each Axis object.

Major grid lines by default do not have their Interval, IntervalOffset, IntervalType, and IntervalOffsetType properties set. When these properties for major grid lines are not set, the Chart control determines the appropriate values for grid lines automatically.

The axis and its minor grid lines by default have their Interval, IntervalOffset, IntervalType, and IntervalOffsetType properties set to **Auto** (0). If axis labels or minor grid lines are enabled and these properties are not modified, the Chart control automatically determines how often to draw the minor grid lines.

Axis grids:

- Can be enabled or disabled.
- Have a starting point that determines where they are first drawn.
- Display lines that can be set for style, width and color.

- Can have either automatic intervals, or intervals that are set manually.

Constructors

Grid()	Initializes a new instance of the Grid class.

Properties

Enabled	Gets or sets a flag that determines whether major or minor grid lines are enabled.
Interval	Gets or sets the interval between major or minor grid lines.
IntervalOffset	Gets or sets the offset of grid lines.
IntervalOffsetType	Gets or sets the interval offset type of major and minor grid lines.
IntervalType	Gets or sets the interval type for major or minor grid lines.
LineColor	Gets or sets the line color of a grid.
LineDashStyle	Gets or sets the line style of a grid.
LineWidth	Gets or sets the line width of major and minor grid lines.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Grid class.

C#

```
public Grid ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether major or minor grid lines are enabled.

C#

```
[System.ComponentModel.Bindable(true)]
public bool Enabled { get; set; }
```

Property Value

Boolean

`true` if enabled, `false` if disabled. The default is `true` for major grid objects and `false` for minor grid objects.

Attributes BindableAttribute

Remarks

Major and minor grids are visible when enabled, and not visible when disabled.

Tick marks will not be displayed if disabled.

You can disable tick marks by setting the TickMarkStyle property to None.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid.Interval Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval between major or minor grid lines.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisElementIntervalValueConverter))]
public double Interval { get; set; }
```

Property Value

Double

A `double` value that represents the interval between grid lines. By default, the value is not set (NaN) for major grid lines. For minor grid lines, the default value is zero (0).

Attributes BindableAttribute, TypeConverterAttribute

Remarks

By default this property is not set for axis labels, major tick marks and major grid lines. If it is not set - either by default, by explicitly using NotSet at design time, or by assigning a value of NaN at run time - the actual value of this property is determined by the Interval property of the Axis object to which the chart element, such as a grid line or tick mark, belongs.

The interval of minor tick marks and grid lines is never affected by the Interval property value.

The interval is measured in axis units, and determines how often an applicable chart element is drawn. For example, suppose a column chart has an X axis step of 1 (that is, 1, 2, 3, 4, and so forth); if you set the Interval property for the X axis to 2, an axis label will be displayed for every other column.

The same holds true for all applicable chart elements. In other words, if in the previous example strip lines were being used, with an IntervalOffset of 0.5, and their Interval property was 2, a strip line would be displayed at 0.5, 2.5, 4.5, and so on.

If you set a chart element `Interval` property to zero, an interval will be automatically determined by the Chart control.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid.IntervalOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the offset of grid lines.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisElementIntervalValueConverter))]
public double IntervalOffset { get; set; }
```

Property Value

Double

A `double` value that represents the interval offset.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

By default this property is not set for axis labels, major tick marks and major grid lines. If it is not set - either by default, by explicitly specifying "Not Set"at design time, or by assigning it a value of NotSet at run time - the actual value of this property is determined by the IntervalOffset property of the Axis object that the chart element, such as a grid line or a tick mark, belongs to.

The interval offset of minor tick marks and grid lines is never affected by the value of the IntervalOffset property of the Axis object.

The IntervalOffsetType property setting defines the units of measurement of the offset.

The IntervalOffset can be a negative value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid.IntervalOffsetType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval offset type of major and minor grid lines.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalOffsetType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that indicates the interval type. By default it is not set for major grid lines. The default value for minor grid lines is Auto.

Attributes BindableAttribute

Remarks

By default this property is not set for axis labels, major tick marks and major grid lines. If it is not set - either by default, by explicitly specifying "Not Set" at design time, or by assigning it a value of NotSet at run time - the actual value of this property is determined by the IntervalOffsetType property of the Axis object that the chart element - a grid line or tick mark, for example - belongs to.

The interval offset type of minor tick marks and grid lines is never affected by the IntervalOffsetType property value.

If this interval offset type is set to Auto, the IntervalType property will be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid.IntervalType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval type for major or minor grid lines.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that indicates the interval type. By default it is not set for axis labels, major tick marks and major grid lines, by using the NotSet enumeration value. The default value for minor tick marks and grid lines is Auto.

Attributes BindableAttribute

Remarks

By default this property is not set for axis labels, major tick marks and major grid lines. If it is not set - either by default, by explicitly using "Not Set" at design time, or by assigning it a value of NotSet at run time - the actual value of this property is determined by the IntervalOffsetType property of the Axis object that the chart element - a grid line or tick mark, for example - belongs to.

The interval type of minor tick marks and grid lines is never affected by the value of the IntervalType property.

Interval types can be set to Auto; the type of interval then depends on either the X-values or Y-values of the plotted DataPoint objects, depending on which axis the applicable elements belong to.

Setting the IntervalType property for a value axis will only have an effect if the values of the data points are DateTime or `double` values. Also, the XValueType or YValueType property of the plotted Series must be set to Auto or DateTime.

Setting the IntervalType property for a categorical axis will only have an effect if the XValueType property (or YValueType property, depending on the axis in question) of the plotted Series object is either Auto or DateTime.

> **Note**
>
> Sunday is considered the first day of the week.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line color of a grid.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color object. The default value is Black.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The LineColor property is used to specify the color of major and minor tick marks or grids.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line style of a grid.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
LineDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value. The default is Solid.

Attributes BindableAttribute

Remarks

The LineDashStyle property can be used to specify the line style of major and minor tick marks or grids. The line can be solid, dotted, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Grid.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line width of major and minor grid lines.

C#

```
[System.ComponentModel.Bindable(true)]
public int LineWidth { get; set; }
```

Property Value

Int32

An `integer` that represents the line width in pixels. The default value is one (1).

Attributes BindableAttribute

Remarks

The LineWidth property can be used to specify the line width, in pixels, of major and minor tick marks or grids.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

GridTickTypes Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the options for displaying a grid line or a tick mark for a custom label.

This enumeration supports a bitwise combination of its member values.

C#

```
[System.Flags]
public enum GridTickTypes
```

Inheritance Object → ValueType → Enum → GridTickTypes

Attributes FlagsAttribute

Fields

All	3	A tick mark and a grid line are shown.
Gridline	2	A grid line is shown.
None	0	No tick mark or grid line is shown.
TickMark	1	A tick mark is shown.

Remarks

The GridTickTypes enumeration is used in conjunction with the GridTicks property to display a grid line and/or a tick mark for a custom label.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the results of a hit test.

```
C#
public class HitTestResult
```

Inheritance Object → HitTestResult

Remarks

The HitTestResult class represents the results of a hit test, and is exposed as the return object from a HitTest method call.

A hit test checks a specified point in the chart to determine if a chart element is located at that point. If a chart element is detected, the HitTestResult class is used to obtain information about that chart element.

If a chart element is found, the Object property returns a reference to the chart element; the only exception to this being DataPoint objects, in which case the return value of this property is `null`.

If a data point is located, you can use the PointIndex property to get the index of that data point; that is, the zero-based index of the point in the DataPointCollection. Use the Series property to get the associated Series object for that data point.

The ChartArea and Axis properties are used to get the associated chart area and axis of the chart element - if any - returned by the hit test. Note that these properties may be `null` if the returned chart element does not have an associated chart area or axis.

Use the ChartElementType property to determine the type of the chart element returned by a hit test.

Constructors

HitTestResult()	Initializes a new instance of the HitTestResult class.

Properties

Axis	Gets or sets the Axis object that may be associated with a chart element returned by a hit test. The presence of an associated Axis object depends on whether a chart element was detected at the given location, and the type of chart element returned.
ChartArea	Gets or sets the ChartArea object that may be associated with a chart element returned by a hit test. The presence of a ChartArea object depends on whether a chart element was detected at the given location, and the type of chart element returned.
ChartElementType	Gets or sets the type of chart element, if any, returned by a hit test.
Object	Gets or sets the object, if any, returned by a hit test.
PointIndex	Gets or sets the index of a data point, if any, returned by a hit test.
Series	Gets or sets the associated series of a data point, if any, returned by a hit test.
SubObject	Gets the sub-object, if any, returned by a hit test.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the HitTestResult class.

C#

```
public HitTestResult ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult.Axis Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Axis object that may be associated with a chart element returned by a hit test. The presence of an associated Axis object depends on whether a chart element was detected at the given location, and the type of chart element returned.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Axis Axis { get; set;
}
```

Property Value

Axis

An Axis object, otherwise `null`. The default value is `null`.

Remarks

If the chart object returned by the hit test is an object that is associated with an axis - for example, a scrollbar element, tick mark, grid line or strip line - the Axis property will reference the Axis object with which the returned chart object is associated. Otherwise this property will be `null`.

Note that the HitTest method can be called explicitly, in which case a chart element may or may not be at the specified location.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult.ChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the ChartArea object that may be associated with a chart element returned by a hit test. The presence of a ChartArea object depends on whether a chart element was detected at the given location, and the type of chart element returned.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartArea ChartArea {
get; set; }
```

Property Value

ChartArea

A ChartArea object, otherwise `null`. The default value is `null`.

Remarks

If a Chart object is returned by the hit test, the ChartArea property will reference the ChartArea object that is associated with the returned chart element.

If no chart element is returned by the hit test, the value of this property will be `null`.

Note The HitTest method can be called explicitly, in which case a chart element may or may not be at the specified location.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult.ChartElementType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the type of chart element, if any, returned by a hit test.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartElementType
ChartElementType { get; set; }
```

Property Value

ChartElementType

A ChartElementType enumeration value that indicates the type of chart element returned by a hit test. Defaults to Nothing.

Remarks

Use this property to determine the type of chart element, if any, returned by a hit test.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult.Object Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the object, if any, returned by a hit test.

C#

```
public object Object { get; set; }
```

Property Value

Object

An object of type Object, otherwise `null`.

Remarks

If a chart element is located at the point that is checked by the hit test, this property will reference that chart element, otherwise the value of this property will be `null`.

> **Note**
>
> If a data point is returned by the hit test, this property will still return a `null`. To obtain information about the data point, use the **PointIndex** and **Series** properties instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult.PointIndex Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the index of a data point, if any, returned by a hit test.

C#

```
public int PointIndex { get; set; }
```

Property Value

Int32

An `integer` value that represents the index of a data point returned by a hit test. The default value is negative one (-1).

Remarks

If a data point is at the location that is checked by a hit test, the PointIndex property will return the zero-based index of that data point, otherwise it will return a negative one (-1).

All data points are stored in a DataPointCollection, which is exposed as the Points collection property of a Series object; the index of the DataPointCollection is zero-based.

Use the Series property to determine the series to which the data point belongs.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult.Series Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the associated series of a data point, if any, returned by a hit test.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Series Series { get;
set; }
```

Property Value

Series

The Series that a data point returned by a hit test belongs to. The default value is `null`.

Remarks

If a data point is at the location checked by a hit test, this property will reference the Series object that the data point belongs to. If no data point is returned by the hit test, the value of this property will be `null`.

To determine the index of a returned data point, use the PointIndex property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HitTestResult.SubObject Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the sub-object, if any, returned by a hit test.

C#

```
public object SubObject { get; set; }
```

Property Value

Object

A sub-object of type Object, otherwise `null`.

Remarks

An example of a sub-object is an Axis object that is displayed inside a ChartArea object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HorizontalLineAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a horizontal line annotation.

C#

```
public class HorizontalLineAnnotation :
System.Windows.Forms.DataVisualization.Charting.LineAnnotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → LineAnnotation → HorizontalLineAnnotation

Constructors

HorizontalLineAnnotation()	Initializes a new instance of the HorizontalLineAnnotation class.

Properties

Alignment	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)

AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the position alignment of an annotation to the anchor point. (Inherited from LineAnnotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the annotation type name.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)

AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
BackGradientStyle	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
BackHatchStyle	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
BackSecondaryColor	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
EndCap	Gets or sets a style for the anchor cap at the end of the annotation line. (Inherited from LineAnnotation)
Font	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
ForeColor	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsInfinitive	Gets or sets a flag that indicates whether an infinite line should be drawn. (Inherited from LineAnnotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from LineAnnotation)
LineColor	Gets or sets the line color of an annotation. (Inherited from Annotation)
LineDashStyle	Gets or sets the line style of an annotation. (Inherited from Annotation)

LineWidth	Gets or sets the line width of an annotation. (Inherited from Annotation)
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
StartCap	Gets or sets a style for the anchor cap at the start of an annotation line. (Inherited from LineAnnotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the

	mouse. (Inherited from Annotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends the end user placement of an annotation. (Inherited from Annotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, Data Point)	Anchors an annotation to two data points. (Inherited from Annotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HorizontalLineAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the HorizontalLineAnnotation class.

C#

```
public HorizontalLineAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

HorizontalLineAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type name.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

This abstract property is used to get the name of each annotation type, for example, LineAnnotation, RectangleAnnotation, EllipseAnnotation, and so forth.

This property is for internal use.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

IDataPointFilter Interface

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Exposes the FilterDataPoint(DataPoint, Series, Int32) method, which allows Series data to be custom filtered.

C#

```
public interface IDataPointFilter
```

Remarks

The IDataPointFilter interface exposes the FilterDataPoint method, which takes a specific data point within a series and returns a `Boolean` value that determines whether that data point will be filtered.

To perform custom filtering:

- Derive a new or existing class from this interface.
- Override the FilterDataPoint method to provide the custom filtering logic.
- To initiate the filtering operation, call one of the `Filter` methods in the DataManipulator class that takes an IDataPointFilter object as its first parameter.

For sample source code, see the FilterDataPoint method.

Methods

FilterDataPoint(DataPoint, Series, Int32)	Determines whether a particular data point in the series will be filtered.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

IDataPointFilter.FilterDataPoint(DataPoint, Series, Int32) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Determines whether a particular data point in the series will be filtered.

C#

```
public bool FilterDataPoint
(System.Windows.Forms.DataVisualization.Charting.DataPoint point,
System.Windows.Forms.DataVisualization.Charting.Series series, int
pointIndex);
```

Parameters

`point` DataPoint

The DataPoint object to be filtered.

`series` Series

The Series object that the data point belongs to.

`pointIndex` Int32

The index of the data point, which determines its position in the series.

Returns

Boolean

`true` if the data point is removed; otherwise, `false`.

Remarks

This method should be called for each data point in a series, and returns a `Boolean` value indicating whether a data point should be filtered.

To provide custom filtering, derive a new or existing class from the IDataPointFilter interface and then implement this method. Provide custom criteria matching logic, then return `true` or `false` to mark the data points to be filtered. After this method has been implemented, use one of the `Filter` method definitions in the DataManipulator class that takes an IDataPointFilter object as a parameter to perform the filtering.

Filtered points, by default, are removed from a series. They can also be displayed as empty points by setting the value of the FilterSetEmptyPoints property to `true`.

Filtered points can also be marked as filtered if they do not match filtering criteria, as determined by the FilterMatchedPoints value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents an image annotation.

C#

```
public class ImageAnnotation :
System.Windows.Forms.DataVisualization.Charting.Annotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → ImageAnnotation

Constructors

ImageAnnotation()	Initializes a new instance of the ImageAnnotation class.

Properties

Alignment	Gets or sets the alignment of the annotation image.
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)

AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point. (Inherited from Annotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the annotation type name.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to.

	(Inherited from Annotation)
BackColor	This property is not applicable to the image annotation type.
BackGradientStyle	This property is not applicable to the image annotation type.
BackHatchStyle	This property is not applicable to the image annotation type.
BackSecondaryColor	This property is not applicable to the image annotation type.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
Font	This property is not applicable to the image annotation type.
ForeColor	This property is not applicable to the image annotation type.
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
Image	Gets or sets the name of the annotation image.
ImageTransparentColor	Gets or sets a color that will be replaced with a transparent color when the image is drawn.
ImageWrapMode	Gets or sets the drawing mode of the image.
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	This property is not applicable to the image annotation type.
LineDashStyle	This property is not applicable to the image annotation type.
LineWidth	This property is not applicable to the image annotation type.
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)

ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	Gets or sets the text style of the annotation.
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends the end user placement of an annotation. (Inherited from Annotation)
Equals(Object)	Determines whether the specified Object is equal to the current

	ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ImageAnnotation class.

C#

```
public ImageAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of the annotation image.

C#

```
public override System.Drawing.ContentAlignment Alignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment value that represents the alignment of the annotation image.

Remarks

This property is used to align a non-scaled image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type name.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

This abstract property is used to get the name of each annotation type such as LineAnnotation, RectangleAnnotation, EllipseAnnotation, and so forth.

This property is for internal use.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.Image Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the annotation image.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual string Image { get; set; }
```

Property Value

String

A `string` value that represents the name of the annotation image.

Attributes BindableAttribute

Remarks

The image name can be a file name, a URL or a named image - that is, its name component - from the NamedImagesCollection class.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.ImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color that will be replaced with a transparent color when the image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that will be replaced with a transparent color when the image is drawn.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.ImageWrapMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the drawing mode of the image.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartImageWrapMode
ImageWrapMode { get; set; }
```

Property Value

ChartImageWrapMode

A ChartImageWrapMode enumeration value that defines the drawing mode of the image.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartDashStyle LineDashStyle
{ get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the image annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override int LineWidth { get; set; }
```

Property Value

Int32

An `integer` value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the image annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ImageAnnotation.TextStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text style of the annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Windows.Forms.DataVisualization.Charting.TextStyle
TextStyle { get; set; }
```

Property Value

TextStyle

A TextStyle enumeration value that represents the style used to draw the annotation text.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

IntervalAutoMode Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the automatic axis interval mode.

C#

```
public enum IntervalAutoMode
```

Inheritance Object → ValueType → Enum → IntervalAutoMode

Fields

FixedCount	0	A fixed number of intervals are always created on the axis.
VariableCount	1	The number of axis intervals depends on the axis length.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

IntervalType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies an interval type.

C#

```
public enum IntervalType
```

Inheritance Object → ValueType → Enum → IntervalType

Fields

Days	4	An interval of days.
Hours	5	An interval of hours.
Milliseconds	8	An interval of milliseconds.
Minutes	6	An interval of minutes.
Months	2	An interval of months.
Number	0	A numerical interval.
Seconds	7	An interval of seconds.
Weeks	3	An interval of weeks.
Years	1	An interval of years.

Remarks

The IntervalType enumeration represents the type of interval and interval offset used for some definitions of the `Group` and `InsertEmptyPoints` methods of the DataManipulator class.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelAlignmentStyles Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a label alignment when SmartLabelStyle is used.

This enumeration supports a bitwise combination of its member values.

C#

```
[System.Flags]
public enum LabelAlignmentStyles
```

Inheritance Object → ValueType → Enum → LabelAlignmentStyles

Attributes FlagsAttribute

Fields

Bottom	2	Label is aligned to the bottom of the data point.
BottomLeft	64	Label is aligned to the bottom-left of the data point.
BottomRight	128	Label is aligned to the bottom-right of the data point.
Center	256	Label is aligned to the center of the data point.
Left	8	Label is aligned to the left of the data point.
Right	4	Label is aligned to the right of the data point.
Top	1	Label is aligned to the top of the data point.
TopLeft	16	Label is aligned to the top-left corner of the data point.
TopRight	32	Label is aligned to the top-right corner of the data point.

Remarks

The LabelAlignmentStyles enumeration represents the alignment of a label; it is set by the MovingDirection property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelAutoFitStyles Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies style changes that can automatically be made to a label when the LabelAutoFitStyle property is used.

This enumeration supports a bitwise combination of its member values.

C#

```
[System.Flags]
public enum LabelAutoFitStyles
```

Inheritance Object → ValueType → Enum → LabelAutoFitStyles

Attributes FlagsAttribute

Fields

DecreaseFont	2	Label font can be decreased.
IncreaseFont	1	Label font can be increased.
LabelsAngleStep30	8	Labels can be angled in 30 degree steps: 0, 30, 60 and 90.
LabelsAngleStep45	16	Labels can be angled in 45 degree steps: 0, 45, and 90.
LabelsAngleStep90	32	Labels can be angled in 90 degree steps: 0 and 90.
None	0	No label changes are allowed.
StaggeredLabels	4	Labels can be staggered.
WordWrap	64	Labels can be word wrapped.

Remarks

The LabelAutoFitStyles enumeration represents the changes to the font or orientation of labels that can be automatically made to enable labels to fit along the axis.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelCalloutStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the type of leader line that should be drawn from the point to the label.

C#

```
public enum LabelCalloutStyle
```

Inheritance Object → ValueType → Enum → LabelCalloutStyle

Fields

Box	2	Line will terminate at a box drawn around the label text.
None	0	Line will terminate at the label without any added style.
Underlined	1	Line will terminate with a horizontal line beneath the label text.

Remarks

The LabelCalloutStyle enumeration represents the type of leader line that should be drawn from the point to the label. It is used in conjunction with the CalloutStyle property of the SmartLabelStyle object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelMarkStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a mark for custom labels.

C#

```
public enum LabelMarkStyle
```

Inheritance Object → ValueType → Enum → LabelMarkStyle

Fields

Box	3	Draws a box around the label. The box always starts at the axis position.
LineSideMark	2	Custom labels use line and side marks.
None	0	No label marks are used.
SideMark	1	Custom labels use side marks.

Remarks

The LabelMarkStyle enumeration represents mark(s) for custom labels that are displayed along the second axis label row. This row can only be generated from custom labels. This enumeration is used in conjunction with the LabelMark property of the CustomLabel class.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelOutsidePlotAreaStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the degree to which a label can be drawn outside a plotting area.

C#

```
public enum LabelOutsidePlotAreaStyle
```

Inheritance Object → ValueType → Enum → LabelOutsidePlotAreaStyle

Fields

No	1	Labels cannot de drawn outside the plotting area.
Partial	2	Labels can be drawn partially outside the plotting area.
Yes	0	Labels can be drawn outside the plotting area.

Remarks

The LabelOutsidePlotAreaStyle enumeration represents the degree to which a label is allowed to leave a plotting area. It is used in conjunction with the AllowOutsidePlotArea property of the SmartLabelStyle object.

If you specify the Partial value, no more than 50% of the label can be drawn outside the plotting area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides properties that define the visual appearance of the axis labels.

C#

```
public class LabelStyle :
System.Windows.Forms.DataVisualization.Charting.ChartElement
```

Inheritance Object → ChartElement → LabelStyle

Remarks

Members of this class are also used to calculate the intervals and positions of the axis labels, and to draw them.

Constructors

LabelStyle()	Initializes a new instance of the LabelStyle class.

Properties

Angle	Gets or sets a value that represents the angle at which the font is drawn.
Enabled	Gets or sets a flag that indicates whether the label is enabled.
Font	Gets or sets the font of the label.
ForeColor	Gets or sets the color of the label.
Format	Gets or sets the formatting string for the label text.
Interval	Gets or sets the size of the label interval.

IntervalOffset	Gets or sets the offset of the label interval.
IntervalOffsetType	Gets or sets the unit of measurement of the label interval offset.
IntervalType	Gets or sets the unit of measurement for the size of a label interval.
IsEndLabelVisible	Gets or sets a flag that determines whether the labels are shown at axis ends.
IsStaggered	Gets or sets a flag that determines whether the labels are shown with an offset.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TruncatedLabels	Gets or sets a flag that determines whether the labels can be truncated.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the resources used by the LabelStyle.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the LabelStyle class.

C#

```
public LabelStyle ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.Angle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that represents the angle at which the font is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
public int Angle { get; set; }
```

Property Value

Int32

An `integer` value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the label is enabled.

C#

```
[System.ComponentModel.Bindable(true)]
public bool Enabled { get; set; }
```

Property Value

Boolean

`true` if the label is enabled; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font of the label.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the label.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.Format Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the formatting string for the label text.

C#

```
[System.ComponentModel.Bindable(true)]
public string Format { get; set; }
```

Property Value

String

A `string` value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.Interval Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the label interval.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisElementIntervalValueConverter))]
public double Interval { get; set; }
```

Property Value

Double

A `double` value.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property is not set by default.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.IntervalOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the offset of the label interval.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisElementIntervalValueConverter))]
public double IntervalOffset { get; set; }
```

Property Value

Double

A `double` value.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property is not set by default.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.IntervalOffsetType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unit of measurement of the label interval offset.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalOffsetType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value.

Attributes BindableAttribute

Remarks

This property is not set by default.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.IntervalType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unit of measurement for the size of a label interval.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value.

Attributes BindableAttribute

Remarks

This property is not set by default.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.IsEndLabelVisible Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether the labels are shown at axis ends.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsEndLabelVisible { get; set; }
```

Property Value

Boolean

`true` if the labels are shown at axis ends; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.IsStaggered Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether the labels are shown with an offset.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsStaggered { get; set; }
```

Property Value

Boolean

`true` if the labels are shown with an offset; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.TruncatedLabels Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether the labels can be truncated.

C#

```
[System.ComponentModel.Bindable(true)]
public bool TruncatedLabels { get; set; }
```

Property Value

Boolean

`true` if the labels can be truncated; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LabelStyle.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the resources used by the LabelStyle.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the legend for the chart image.

C#

```
public class Legend :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → Legend

Remarks

The Legend class encapsulates all the functionality of the Chart control's legend, and is exposed using the Legends collection property of the root Chart object.

There can be any number of legends for a chart image.

Legends can:

- Be docked and aligned.
- Be displayed inside or outside the plotted chart areas.
- Automatically fit their contents.
- Use a predefined style, such as row, column or table.
- Use hatching, gradient colors and background images.
- Be sized and positioned anywhere within the chart image.

Legend items are represented by the LegendItem object, and are either default or custom items. Custom items are not associated with data, and can be added, deleted and modified using the CustomItems property.

Default items are by default automatically added to the legend, and are the names of the plotted data points for doughnut and pie charts.

For all other chart types, the default items are the names of the plotted data series. Each series has a Legend property that defines the name of the legend that will be used. A legend item will not be created for a series if the IsVisibleInLegend property is set to `false`. The text for these default items can be changed using the LegendText property. This property will allow you to specify predefined keywords that will be replaced by their appropriate values at run-time.

If you have a pie chart and you set the Label property, it will also set the legend text to the value you set for the labels. If you want to set the text to a different value, you can set the LegendText property. In most cases, you would want to set the LegendText property to "#AXISLABEL" or "#VALX".

Legend series symbols are rectangles, dots or lines. The symbol type is set automatically by the chart and depends on the plotted chart type.

Legend text by default is automatically sized. You can disable automatic sizing of legend text by disabling the IsTextAutoFit property. When this property is set to `false`, the size of the legend text is determined by the Size property of the legend Font. If the IsTextAutoFit property is enabled, the AutoFitMinFontSize property determines the minimum font size that a legend item can have when resizing.

The TextWrapThreshold property defines a limit on the number of characters that can be shown on one line before text is wrapped to another line. If you do not wish to wrap text in the legend, set the value of the TextWrapThreshold property to zero. By default, when the entire legend text does not fit in the legend, the text will be truncated. You can set the TextWrapThreshold on the legend to get the text to wrap instead of being truncated.

If the contents of the legend cannot fit into the given space, legend entries are truncated. An ellipsis is shown to indicate that there are more entries than the legend has space for. To prevent the legend items from being truncated, set the position and size of the legend using the Position property. When setting the legend position or size, the chart can no longer automatically position the elements in the chart. In some cases, you many need to manually position other chart elements to achieve the desired appearance.

Another option would be to define your own cell columns instead of using the default legend settings.

The Legend object has a CellColumn Collection; each CellColumn object represents a column in the Legend. A cell column has MaximumWidth and MinimumWidth properties that can be used to adjust the size of the columns.

When the legend contains many entries, InterlacedRows can be used to improve readability. When this property is set to True, it will display a back color that you can set using the InterlacedRowsColor property.

By default, the legend is positioned in the top-right corner of the chart. This position can be modified by using the Docking and Alignment properties of the legend. The legend can also be associated with a chart area by use of the DockedToChartArea property. This property specifies the name of the chart area for which the legend should be docked. In addition, the IsDockedInsideChartArea property can be used to display the legend inside a chart area.

When IsDockedInsideChartArea is set to False, the MaximumAutoSize property defines the maximum size, in percent, of the legend within the available chart when the legend is automatically positioned. This property should be set if you wish to adjust the width or height of the legend after the docking properties have been set.

The legend can be modified using the CustomizeLegend event.

Constructors

Legend()	Initializes a new instance of the Legend class.
Legend(String)	Initializes a new instance of the Legend class with the specified legend name.

Properties

Alignment	Gets or sets the alignment of the legend.
AutoFitMinFontSize	Gets or sets the minimum font size that can be used by the autofitting algorithm for the legend text.
BackColor	Gets or sets the background color of a legend.
BackGradientStyle	Gets or sets the orientation for the background gradient of a legend. Also determines whether a gradient is used.
BackHatchStyle	Gets or sets the hatching style of a legend.
BackImage	Gets or sets the background image of a legend.

BackImageAlignment	Gets or sets the background image alignment used for the Unscaled drawing mode.
BackImageTransparentColor	Gets or sets a color that will be replaced with a transparent color when the background image is drawn.
BackImageWrapMode	Gets or sets the legend background image drawing mode.
BackSecondaryColor	Gets or sets the secondary color of a legend background.
BorderColor	Gets or sets the border color of a legend.
BorderDashStyle	Gets or sets the border style of a legend.
BorderWidth	Gets or sets the border width of a legend.
CellColumns	Gets the LegendCellColumnCollection object that contains the collection of cell columns for a legend.
CustomItems	Gets a LegendItemsCollection object used for custom legend items.
DockedToChartArea	Gets or sets the name of the ChartArea where the legend will be docked.
Docking	Gets or sets a value that determines whether the legend is docked.
Enabled	Gets or sets a value that indicates whether the legend is enabled.
Font	Gets or sets a Font object, which is used to set font properties of the legend.
ForeColor	Gets or sets the color of the legend text.
HeaderSeparator	Gets or sets the visual separator type for the legend header.
HeaderSeparatorColor	Gets or sets the color of the separator for the legend header.
InsideChartArea	Gets or sets the name of the chart area in which the legend is drawn.
InterlacedRows	Gets or sets a flag that indicates whether legend rows will be drawn with interlaced background color.
InterlacedRowsColor	Gets or sets the background color of the legend's interlaced rows.
IsDockedInsideChartArea	Gets or sets a flag that indicates whether the legend is docked inside or outside the chart area.
IsEquallySpacedItems	Gets or sets a flag that indicates whether all legend items are equally spaced.

IsTextAutoFit	Gets or sets a flag that indicates whether the legend text is automatically sized.
ItemColumnSeparator	Gets or sets the visual separator type for the legend table columns.
ItemColumnSeparatorColor	Gets or sets the color of the legend table column separator.
ItemColumnSpacing	Gets or sets the legend table column spacing.
LegendItemOrder	Gets or sets a value that specifies the order in which legend items are displayed. This property only affects legend items automatically added for the chart series; it has no effect on custom legend items.
LegendStyle	Gets or sets the style of the legend.
MaximumAutoSize	Gets or sets the maximum size of the legend, measured as a percentage of the chart area. This value is used by the automatic layout algorithm.
Name	Gets or sets the name of the legend.
Position	Gets or sets an ElementPosition object, which can be used to get or set the position of the legend.
ShadowColor	Gets or sets the shadow color of the legend.
ShadowOffset	Gets or sets the shadow offset, in pixels, of the legend.
TableStyle	Gets or sets the legend table style.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextWrapThreshold	Gets or sets the number of characters that can be sequentially displayed in the legend before the text is wrapped.
Title	Gets or sets the text of the legend title.
TitleAlignment	Gets or sets the alignment of the legend title.
TitleBackColor	Gets or sets the background color of the legend title.
TitleFont	Gets or sets the font for the legend title.
TitleForeColor	Gets or sets the text color of the legend title.
TitleSeparator	Gets or sets the visual separator type for the legend title.
TitleSeparatorColor	Gets or sets the color of the legend title separator.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the Legend and optionally releases the managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Legend class.

Overloads

Legend()	Initializes a new instance of the Legend class.
Legend(String)	Initializes a new instance of the Legend class with the specified legend name.

Legend()

Initializes a new instance of the Legend class.

C#

```
public Legend ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend(String)

Initializes a new instance of the Legend class with the specified legend name.

C#

```
public Legend (string name);
```

Parameters

`name` String

The name of the Legend object.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of the legend.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.StringAlignment Alignment { get; set; }
```

Property Value

StringAlignment

A .NET Framework StringAlignment enumeration value that represents the alignment of the legend. The default value is Near.

Attributes BindableAttribute

Remarks

The legend can be aligned so that it is Near, Center or Far; its alignment is also affected by the Docking property setting.

If the legend is docked to the top or bottom, Near is to the left; if it is docked to the left or right, Near is at the top. The Far property setting is just the opposite, and aligns the legend to the right or the bottom, depending on where it is docked.

Center always centers the legend, either vertically or horizontally.

Setting the Alignment property will have no effect if the Position property is not set to Auto.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.AutoFitMinFontSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the minimum font size that can be used by the autofitting algorithm for the legend text.

C#

```
public int AutoFitMinFontSize { get; set; }
```

Property Value

Int32

An `integer` value that represents the minimum font size that can be used by the autofitting algorithm for the legend text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of a legend.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value used to draw the background color of a legend. The default color is White.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set this property to any valid ARGB (alpha, red, green, blue) value.

The BackColor value will be the first color used if you have specified gradient colors for the background.

The alpha value controls the opacity of a color. If you set this color to "transparent" - that is, if you use an alpha value of 0 - no color will be assigned to the background of the legend. As a result, the background will be transparent.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the orientation for the background gradient of a legend. Also determines whether a gradient is used.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration that specifies the orientation for the background gradient of a legend. The default value is None.

Attributes BindableAttribute

Remarks

Use the BackGradientStyle property to determine whether a legend will display a gradient, and to determine the orientation of the gradient if one is used.

For a listing of available patterns, refer to the GradientStyle enumeration.

To use gradient colors:

Set the primary color of the legend background with the BackColor property.

Set the secondary color of the legend background with the BackSecondaryColor property.

Set this BackGradientStyle property to the desired gradient style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatching style of a legend.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration that specifies the hatching style of a legend. The default value is None.

Attributes BindableAttribute

Remarks

The ChartHatchStyle enumeration specifies the hatch pattern used by a brush of type HatchBrush. The hatch pattern consists of a solid background color and lines drawn over the background.

For a listing of available hatching styles, refer to the ChartHatchStyle enumeration.

Note The color of hatching lines is determined by the BackSecondaryColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BackImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image of a legend.

C#

```
[System.ComponentModel.Bindable(true)]
public string BackImage { get; set; }
```

Property Value

String

A `string` value that represents the URL of an image file. The default value is an empty string.

Attributes BindableAttribute

Remarks

This property defines a path to a background image for a Chart legend, and can be an absolute or relative URL.

To position the image, use the BackImageAlignment and BackImageWrapMode properties.

You can also specify one of the colors displayed by the image as "transparent", by setting the BackImageTransparentColor property.

An image can originate from any valid graphics file supported by .NET Framework, including icons.

Background images take precedence over other background properties such as gradient color.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BackImageAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image alignment used for the Unscaled drawing mode.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.ChartImageAlignmentStyle
BackImageAlignment { get; set; }
```

Property Value

ChartImageAlignmentStyle

A ChartImageAlignmentStyle enumeration value that specifies the background image alignment. The default value is TopLeft.

Attributes BindableAttribute

Remarks

This property determines the alignment of the background image for the legend; it is only applied when the BackImageWrapMode property is set to Unscaled.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BackImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color that will be replaced with a transparent color when the background image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that will be displayed as transparent color when the background image is drawn. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BackImageWrapMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the legend background image drawing mode.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartImageWrapMode
BackImageWrapMode { get; set; }
```

Property Value

ChartImageWrapMode

A ChartImageWrapMode enumeration value that specifies the legend background image drawing mode. The default value is Tile.

Attributes BindableAttribute

Remarks

The BackImageWrapMode property determines how the image is displayed; the image can be tiled, scaled, and so forth.

If an image is scaled, it will be stretched to fit within the bounds of the legend. If the image is tiled, it is repeatedly drawn, starting from the top-left corner of the legend.

For more information about the available wrapping modes, see the ChartImageWrapMode enumeration topic.

If the BackImageWrapMode property is set to Unscaled, the location where the image is drawn is determined by the BackImageAlignment property setting.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color of a legend background.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value used for the secondary color of background with hatching or gradient fill. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The BackSecondaryColor property determines the secondary background color of many Chart objects, and is used when gradient colors are being applied.

You can set BackSecondaryColor to any valid ARGB (alpha, red, green, blue) value.

If hatching is used, the BackSecondaryColor property determines the color of the hatching lines for the legend.

To use gradient colors for the background of a legend:

- Set the primary color with the BackColor property.
- Set the secondary color with the BackSecondaryColor property.
- Set the BackGradientStyle property to the desired gradient style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of a legend.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderColor { get; set; }
```

Property Value

Color

A Color value used to draw the border color of a legend. The default color is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set this property to any valid ARGB (alpha, red, green, blue) value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border style of a legend.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that determines the border style of the chart element.

Attributes BindableAttribute

Remarks

The BorderDashStyle property determines the line style to use for a border. The line can be solid, dotted, and so forth.

For other border styles, see the ChartDashStyle enumeration topic.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.BorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of a legend.

C#

```
[System.ComponentModel.Bindable(true)]
public int BorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that determines the border width, in pixels, of the legend.

Attributes BindableAttribute

Remarks

Note that BorderWidth property can also set the width of lines in such elements as line charts, custom legend items displayed with a "line" style, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.CellColumns Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the LegendCellColumnCollection object that contains the collection of cell columns for a legend.

C#

```
public
System.Windows.Forms.DataVisualization.Charting.LegendCellColumnCollection
CellColumns { get; }
```

Property Value

LegendCellColumnCollection

A LegendCellColumnCollection object that contains the collection of cell columns for a legend.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.CustomItems Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a LegendItemsCollection object used for custom legend items.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.LegendItemsCollection
CustomItems { get; }
```

Property Value

LegendItemsCollection

A LegendItemsCollection object.

Attributes BindableAttribute

Remarks

This collection property can be used to add, delete or modify custom legend items.

Custom legend items, unlike default legend items, do not have associated series or data points.

To modify legend items, use the CustomizeLegend event.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.DockedToChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the ChartArea where the legend will be docked.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.LegendAreaNameConverter))]
public string DockedToChartArea { get; set; }
```

Property Value

String

A `string` value that represents the name of the ChartArea where this legend will be docked. The default value is the empty string.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.Docking Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that determines whether the legend is docked.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.Docking Docking {
get; set; }
```

Property Value

Docking

A Docking enumeration value that determines whether the legend is docked. The default value is Right.

Attributes BindableAttribute

Remarks

The Legend can be docked to the top, left, bottom or right of either the entire chart image, or the inside or outside of a chart area. If the IsDockedInsideChartArea property is set to `false`, docking applies to the entire chart image. If the IsDockedInsideChartArea property is set to `true`, docking applies to the ChartArea object specified by the DockedToChartArea property

This property has no effect if the Position property is not set to Auto.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that indicates whether the legend is enabled.

C#

```
[System.ComponentModel.Bindable(true)]
public bool Enabled { get; set; }
```

Property Value

Boolean

`true` if the legend is enabled; otherwise, `false`. The default value is `true`.

Attributes BindableAttribute

Remarks

To hide a legend, set this property to `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a Font object, which is used to set font properties of the legend.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object which is used to set font properties of the legend.

Attributes BindableAttribute

Remarks

If you attempt to change the properties of this Font object at run time, a compilation error will occur. To set font properties at run time, you must create a new Font object.

> ⓘ **Note**
>
> If you set the Legend.Font.Size property, the **IsTextAutoFit** property will be set to false. Also, if you set the **IsTextAutoFit** property to true, this property will be reset to its default, which is 8 points.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the legend text.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value that specifies the color of the legend text.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The ForeColor property is used to set the legend text color.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.HeaderSeparator Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the visual separator type for the legend header.

C#

```
public System.Windows.Forms.DataVisualization.Charting.LegendSeparatorStyle
HeaderSeparator { get; set; }
```

Property Value

LegendSeparatorStyle

A LegendSeparatorStyle enumeration value that specifies the visual separator type for the legend header.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.HeaderSeparatorColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the separator for the legend header.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color HeaderSeparatorColor { get; set; }
```

Property Value

Color

A Color value that specifies the color of the separator for the legend header.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.InsideChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the chart area in which the legend is drawn.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.LegendAreaNameConverter))]
public string InsideChartArea { get; set; }
```

Property Value

String

A `string` value that represents the name of the chart area.

Attributes BindableAttribute, BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.InterlacedRows Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether legend rows will be drawn with interlaced background color.

```
C#
public bool InterlacedRows { get; set; }
```

Property Value

Boolean

`true` if legend rows will be drawn with interlaced background color; otherwise, `false`.

Remarks

When the legend contains many entries, the InterlacedRows property can be used to improve readability. When this property is set to `true`, the legend will display a background color for the rows that are interlaced, which can be set using the InterlacedRowsColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.InterlacedRowsColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the legend's interlaced rows.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color InterlacedRowsColor { get; set; }
```

Property Value

Color

A Color value that specifies the background color of the legend's interlaced rows.

Attributes TypeConverterAttribute

Remarks

This property is only applicable if interlaced rows are used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.IsDockedInsideChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the legend is docked inside or outside the chart area.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsDockedInsideChartArea { get; set; }
```

Property Value

Boolean

`true` is the legend is docked inside the chart area; `false` if it is docked outside the chart area. The default value is `false`.

Attributes BindableAttribute

Remarks

This property will only have an effect if the DockedToChartArea property is set to the name of a ChartArea object.

> **Note**
>
> If the **Position** property is set to any value other than **Auto**, the **IsDockedInsideChartArea** property also has no effect. This is because the position of the legend is determined by the **Position** property setting.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.IsEquallySpacedItems Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether all legend items are equally spaced.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsEquallySpacedItems { get; set; }
```

Property Value

Boolean

`true` if legend items are equally spaced; otherwise, `false`. The default value is `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.IsTextAutoFit Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the legend text is automatically sized.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsTextAutoFit { get; set; }
```

Property Value

Boolean

`true` if the legend text is automatically sized; otherwise, `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.ItemColumnSeparator Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the visual separator type for the legend table columns.

C#

```
public System.Windows.Forms.DataVisualization.Charting.LegendSeparatorStyle
ItemColumnSeparator { get; set; }
```

Property Value

LegendSeparatorStyle

A LegendSeparatorStyle enumeration value that specifies the visual separator type for the legend table columns.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.ItemColumnSeparatorColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the legend table column separator.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ItemColumnSeparatorColor { get; set; }
```

Property Value

Color

A Color value used to draw the color of the legend table column separator.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.ItemColumnSpacing Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the legend table column spacing.

C#

```
public int ItemColumnSpacing { get; set; }
```

Property Value

Int32

An `integer` value that represents the table column spacing.

Remarks

The ItemColumnSpacing property value is calculated as a percentage of the legend text font.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.LegendItemOrder Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that specifies the order in which legend items are displayed. This property only affects legend items automatically added for the chart series; it has no effect on custom legend items.

C#

```
public System.Windows.Forms.DataVisualization.Charting.LegendItemOrder
LegendItemOrder { get; set; }
```

Property Value

LegendItemOrder

A LegendItemOrder enumeration value that specifies the order in which legend items are displayed.

Remarks

If the LegendItemOrder property is set to Auto, the legend will automatically be reversed if StackedColumn, StackedColumn100, StackedArea or StackedArea100 chart types are used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.LegendStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the legend.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.LegendStyle
LegendStyle { get; set; }
```

Property Value

LegendStyle

A LegendStyle enumeration value that determines the legend style. The default value is Table.

Attributes BindableAttribute

Remarks

By default, the legend is displayed as a series of items in a table. You can specify whether to expand items in the table width-wise or height-wise by setting the TableStyle property. When the TableStyle property is set to Auto, the Chart control will determine whether to draw a wide table or a tall table, depending on the alignment and positioning of the chart. This may cause the legend to be drawn using multiple legend item columns.

There are three possible styles for a legend: Column, Row and Table. Refer to the table below for a summary of these styles:

Style	Description
Column	Legend items are displayed in one column, with multiple rows. Most commonly used when the legend is docked to the left or right of the chart.

Style	Description
Row	Legend items are displayed in one row, with multiple columns. Most commonly used when the legend is docked to the top or bottom of the chart.
Table	Legend items are displayed using multiple columns and rows.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.MaximumAutoSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the maximum size of the legend, measured as a percentage of the chart area. This value is used by the automatic layout algorithm.

```
C#
public float MaximumAutoSize { get; set; }
```

Property Value

Single

A `float` value that represents the maximum size of the legend.

Remarks

If the legend is docked to the left or right of the chart area, this property determines the maximum width of the legend, which is measured as a percentage of the chart area. If the legend is docked to the top or bottom, this property determines the maximum height of the legend, which is also measured as a percentage.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the legend.

C#

```
[System.ComponentModel.Bindable(true)]
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the name of the legend.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.Position Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an ElementPosition object, which can be used to get or set the position of the legend.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.ElementPositionConverter))]
public System.Windows.Forms.DataVisualization.Charting.ElementPosition
Position { get; set; }
```

Property Value

ElementPosition

If the Position property is automatic (that is, Legend.Position.Auto = `true`) the legend position is calculated automatically by the Chart control, taking into account the Docking, Alignment and IsDockedInsideChartArea property settings.

If it is not automatic, the Docking, DockedToChartArea and IsDockedInsideChartArea properties are ignored, and the legend position is solely determined by the value of the returned ElementPosition object.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.ShadowColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow color of the legend.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ShadowColor { get; set; }
```

Property Value

Color

A Color value that specifies the shadow color of the legend. The default color is Black.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property can be set to any valid ARGB (alpha, red, green, blue) value.

The alpha value can be used to achieve a "realistic" shadowing effect; try a color value of "128,0,0,0" for shadows.

To implement a shadow, set the ShadowOffset property to a value other than zero (0).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.ShadowOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow offset, in pixels, of the legend.

C#

```
[System.ComponentModel.Bindable(true)]
public int ShadowOffset { get; set; }
```

Property Value

Int32

An `integer` value that represents the shadow offset, in pixels, of the legend.

Attributes BindableAttribute

Remarks

To implement shadowing, set this property to a value other than zero (0). A positive value will result in a shadow that shifts down and to the right; a negative value will result in a shadow that shifts up and to the left.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.TableStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the legend table style.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.LegendTableStyle
TableStyle { get; set; }
```

Property Value

LegendTableStyle

A LegendTableStyle enumeration value that specifies the legend table style.

Attributes BindableAttribute

Remarks

This property is only applicable if the LegendStyle property is set to Table.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.TextWrapThreshold Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the number of characters that can be sequentially displayed in the legend before the text is wrapped.

C#

```
public int TextWrapThreshold { get; set; }
```

Property Value

Int32

An `integer` value that represents the number of characters that can be placed in the legend sequentially before the text is wrapped. The default value is 25.

Remarks

When the legend text exceeds the value specified by this property, text will be automatically wrapped when the next white-space character is encountered. Text will not be wrapped if there are no white-space characters in the text.

Set the TextWrapThreshold property to zero to disable the feature.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.Title Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text of the legend title.

C#

```
public string Title { get; set; }
```

Property Value

String

A `string` value that represents the legend title text. The default value is a zero-length string.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.TitleAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of the legend title.

C#

```
public System.Drawing.StringAlignment TitleAlignment { get; set; }
```

Property Value

StringAlignment

A StringAlignment enumeration value that represents the alignment of the legend title. The default value is Center.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.TitleBackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the legend title.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color TitleBackColor { get; set; }
```

Property Value

Color

A Color value used to draw the background of the legend title. The default value is Empty.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.TitleFont Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font for the legend title.

C#

```
public System.Drawing.Font TitleFont { get; set; }
```

Property Value

Font

A Font object that represents the font for the legend title. The default value is "Microsoft Sans Serif, 8pt, Bold".

Remarks

This property can be used to set font properties of the legend title.

If you attempt to change the properties of this Font object at run time, a compilation error will occur. To set font properties at run time, you must create a new Font object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.TitleForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color of the legend title.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color TitleForeColor { get; set; }
```

Property Value

Color

A Color value used to draw the title text color. The default value is Black.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.TitleSeparator Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the visual separator type for the legend title.

C#

```
public System.Windows.Forms.DataVisualization.Charting.LegendSeparatorStyle
TitleSeparator { get; set; }
```

Property Value

LegendSeparatorStyle

A LegendSeparatorStyle value that represents the type of separator that will be drawn below the title. The default value is None.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.TitleSeparatorColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the legend title separator.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color TitleSeparatorColor { get; set; }
```

Property Value

Color

A Color value used to draw the color of the legend title separator. The default value is Black.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Legend.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the unmanaged resources used by the Legend and optionally releases the managed resources.

```
C#
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a legend cell in a legend item.

C#

```
public class LegendCell :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → LegendCell

Remarks

A legend is populated with legend items, which are represented by the LegendItemsCollection. These legend items consist of individual legend cells, which are in turn represented by the LegendCellCollection.

Constructors

LegendCell()	Initializes a new instance of the LegendCell class.
LegendCell(LegendCellType, String)	Initializes a new instance of the LegendCell class with the specified cell type and a `text` parameter that specifies the cell text or image name.
LegendCell(LegendCellType, String, ContentAlignment)	Initializes a new instance of the LegendCell class with the specified cell type, content alignment, and a `text` parameter that specifies the cell text or image name.
LegendCell(String)	Initializes a new instance of the LegendCell class with a `text` parameter that specifies the cell text or image name.

Properties

Alignment	Gets or sets the alignment of the legend cell contents.
BackColor	Gets or sets the background color of the legend cell.
CellSpan	Gets or sets the number of horizontal cells used to draw the contents of the legend cell.
CellType	Gets or sets the legend cell type that is used to define the contents of the legend cell.
Font	Gets or sets the font for the legend cell.
ForeColor	Gets or sets the text color of the legend cell.
Image	Gets or sets the name of the image in a legend cell.
ImageSize	Gets or sets the size of the image in a legend cell.
ImageTransparentColor	Gets or sets a color which will be replaced with a transparent color when the image is drawn.
Legend	Gets the legend in which this cell belongs.
LegendItem	Gets the legend item in which this cell belongs.
Margins	Gets or sets the margins of the legend cell.
Name	Gets or sets the name of the legend cell.
SeriesSymbolSize	Gets or sets the size of the series symbol that is displayed in the legend cell.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the legend cell text.
ToolTip	Gets or sets the tooltip text for the legend cell.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the LegendCell and optionally releases the managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)

GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the LegendCell class.

Overloads

LegendCell()	Initializes a new instance of the LegendCell class.
LegendCell(String)	Initializes a new instance of the LegendCell class with a `text` parameter that specifies the cell text or image name.
LegendCell(LegendCellType, String)	Initializes a new instance of the LegendCell class with the specified cell type and a `text` parameter that specifies the cell text or image name.
LegendCell(LegendCellType, String, ContentAlignment)	Initializes a new instance of the LegendCell class with the specified cell type, content alignment, and a `text` parameter that specifies the cell text or image name.

LegendCell()

Initializes a new instance of the LegendCell class.

C#

```
public LegendCell ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell(String)

Initializes a new instance of the LegendCell class with a `text` parameter that specifies the cell text or image name.

```
C#

public LegendCell (string text);
```

Parameters

`text` String

A `string` value that represent the cell text or image name, depending on the type of cell.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell(LegendCellType, String)

Initializes a new instance of the LegendCell class with the specified cell type and a `text` parameter that specifies the cell text or image name.

```
C#

public LegendCell
(System.Windows.Forms.DataVisualization.Charting.LegendCellType cellType,
string text);
```

Parameters

`cellType` LegendCellType

A LegendCellType value that represents the legend cell type to be used.

`text` String

A string value that represents the cell text or image name, depending on the type of cell.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell(LegendCellType, String, ContentAlignment)

Initializes a new instance of the LegendCell class with the specified cell type, content alignment, and a `text` parameter that specifies the cell text or image name.

C#

```
public LegendCell
(System.Windows.Forms.DataVisualization.Charting.LegendCellType cellType,
string text, System.Drawing.ContentAlignment alignment);
```

Parameters

`cellType` LegendCellType

A LegendCellType value that represents the legend cell type to be used.

`text` String

A `string` value representing the cell text or image name, depending on the type of cell.

`alignment` ContentAlignment

A ContentAlignment value that represents the alignment of the legend cell contents.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of the legend cell contents.

C#

```
public virtual System.Drawing.ContentAlignment Alignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment value that represents the alignment of the legend cell contents. The default value is MiddleCenter.

Remarks

The Alignment property is used to align text, series symbols or images within the legend cell.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the legend cell.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value used to draw the legend cell's background.

Attributes TypeConverterAttribute

Remarks

The BackColor property has the highest priority when setting the background color of any entity in the legend.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.CellSpan Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the number of horizontal cells used to draw the contents of the legend cell.

C#

```
public virtual int CellSpan { get; set; }
```

Property Value

Int32

An `integer` value that represents the number of horizontal cells used to draw the legend cell contents. The default value is 1.

Remarks

If the cell contents must extend beyond the cell width, you can set the CellSpan property to the number of cells required to fit the contents. This property is most often used if you wish to add an extended header or title to the legend.

When this property is set to a value larger than 1, empty legend cells must be added in order to accommodate the contents of the extended legend cell.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.CellType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the legend cell type that is used to define the contents of the legend cell.

C#

```
public virtual
System.Windows.Forms.DataVisualization.Charting.LegendCellType CellType {
get; set; }
```

Property Value

LegendCellType

A LegendCellType enumeration value that represents the type that is used to define the contents of the legend cell. The default value is Text.

Remarks

The CellType property determines the type of content that will be displayed in each individual legend cell. One of the following cell types must be specified:

- Text: A string is specified using the Text property, and will be displayed in the legend cell.
- SeriesSymbol: A series symbol is defined by the ImageStyle property in the LegendItem class. It will be displayed for the legend cell. Recall that a series symbol is a visual indicator - a line, rectangle or marker. When ImageStyle is set to Marker, all marker attributes of the custom item apply to the cell.
- Image: An image is defined using the Image property and will be displayed in the legend cell.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font for the legend cell.

C#

```
public virtual System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object that represents the font for the legend cell. The default value is "Microsoft Sans Serif, 8pt".

Remarks

The Font property can be used to set font properties of the legend cell.

If you attempt to change the properties of this Font object at run time, a compilation error will occur. To set font properties at run time, you must create a new Font object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color of the legend cell.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value used to draw the legend cell text. The default value is Empty.

Attributes TypeConverterAttribute

Remarks

This property is only applicable if the CellType property is set to Text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.Image Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the image in a legend cell.

C#

```
public virtual string Image { get; set; }
```

Property Value

String

A `string` value that represents the name of the image to be used in the legend cell.

Remarks

The image name should be a URL.

This property is only applicable if the CellType property is set to Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.ImageSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the image in a legend cell.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.SizeEmptyValueConverter))]
public virtual System.Drawing.Size ImageSize { get; set; }
```

Property Value

Size

A Size value that represents the size of the image. The default value is (0,0), which means the original size of the image will be used.

Attributes TypeConverterAttribute

Remarks

The ImageSize property is used to explicitly set the width and height of the legend cell image.

This property is only applicable if the CellType property is set to Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.ImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color which will be replaced with a transparent color when the image is drawn.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color ImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that is used as a transparent color in a legend cell image.

Attributes TypeConverterAttribute

Remarks

The ImageTransparentColor property is only applicable if the CellType property is set to Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.Legend Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the legend in which this cell belongs.

C#

```
[System.ComponentModel.Browsable(false)]
public virtual System.Windows.Forms.DataVisualization.Charting.Legend Legend
{ get; }
```

Property Value

Legend

A Legend object that represents the legend in which this cell belongs.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.LegendItem Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the legend item in which this cell belongs.

C#

```
[System.ComponentModel.Browsable(false)]
public virtual System.Windows.Forms.DataVisualization.Charting.LegendItem
LegendItem { get; }
```

Property Value

LegendItem

A LegendItem object that represents the legend item in which this cell belongs.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.Margins Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the margins of the legend cell.

C#

```
public virtual System.Windows.Forms.DataVisualization.Charting.Margins
Margins { get; set; }
```

Property Value

Margins

A Margins object that represents the top, bottom, left, and right values of the legend cell margins. The default values for top, bottom, left and right are "0,0,15,15".

Remarks

Margins are measured as a percentage of the font size for the legend cell.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the legend cell.

C#

```
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the text name of the legend cell.

Remarks

The name of the legend cell must be unique; otherwise an exception will be thrown.

The Name property can be used as the indexer when retrieving a LegendCell object from the LegendCellCollection.

If you set this property to a zero-length string, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.SeriesSymbolSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the series symbol that is displayed in the legend cell.

C#

```
public virtual System.Drawing.Size SeriesSymbolSize { get; set; }
```

Property Value

Size

A Size object that represents the size of the series symbol. The default values for the width and height of the symbol are 200 and 70, respectively.

Remarks

The size of the series symbol is measured as a percentage of the legend font size.

The SeriesSymbolSize property is only applicable if the CellType property is set to SeriesSymbol.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.Text Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the legend cell text.

C#

```
public virtual string Text { get; set; }
```

Property Value

String

A `string` value that represents the text for the legend cell. The default value is a zero-length string.

Remarks

The Text property is only applicable if the CellType property is set to Text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip text for the legend cell.

C#

```
public virtual string ToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip for the legend cell. The default value is a zero-length string.

Remarks

If the `ToolTip` property is set for both a legend item and a legend cell, the legend cell property has precedence.

This ToolTip property allows you to set special keywords to display different types of chart-related data.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCell.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the unmanaged resources used by the LegendCell and optionally releases the managed resources.

```
C#
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Stores legend cells in a LegendCellCollection object.

C#

```
public class LegendCellCollection :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<
System.Windows.Forms.DataVisualization.Charting.LegendCell>
```

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<LegendCell> → LegendCellCollection

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Item[String]	Gets or sets the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)
NamePrefix	Gets the name prefix that is used to create unique chart element names. (Inherited from ChartNamedElementCollection<T>)

Methods

Add(LegendCellType, String, ContentAlignment)	Adds a legend cell with the specified cell type, text and alignment to the end of the collection.

Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds the chart element by the specified name. (Inherited from ChartNamedElementCollection<T>)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(String)	Finds the index of the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, LegendCellType, String, ContentAlignment)	Inserts a legend cell with the specified cell type, text and alignment into the collection at a given location.
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index.

	(Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index. (Inherited from ChartNamedElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection. (Inherited from ChartNamedElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection. (Inherited from ChartNamedElementCollection<T>)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection. (Inherited from ChartNamedElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element. (Inherited from ChartNamedElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)

ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.

CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.

Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource>	Returns the elements of the specified

(IEnumerable<TSource>)	sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.

GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.

Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.

Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Nullable<Double>>)	of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>,	Projects each element of a sequence into a new form by incorporating the element's

Func<TSource,Int32,TResult>)	index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>,	Returns the only element of a sequence that satisfies a specified condition or a default

Func<TSource,Boolean>)	value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.

Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.

ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the

	source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellCollection.Add(LegendCellType, String, ContentAlignment) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a legend cell with the specified cell type, text and alignment to the end of the collection.

C#

```
public int Add
(System.Windows.Forms.DataVisualization.Charting.LegendCellType cellType,
string text, System.Drawing.ContentAlignment alignment);
```

Parameters

`cellType` LegendCellType

A LegendCellType value representing the cell type.

`text` String

A `string` value that represents either the cell text or its image name, depending on the value of the `cellType` parameter.

`alignment` ContentAlignment

A ContentAlignment value that represents the cell content alignment.

Returns

Int32

An `integer` value that represents the index of the newly added legend cell in the collection. The index is zero-based.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellCollection.Insert Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Inserts a legend cell with the specified cell type, text and alignment into the collection at a given location.

C#

```
public void Insert (int index,
System.Windows.Forms.DataVisualization.Charting.LegendCellType cellType,
string text, System.Drawing.ContentAlignment alignment);
```

Parameters

`index` Int32

Index of the location to insert the object.

`cellType` LegendCellType

A LegendCellType value that represents the cell type.

`text` String

A `string` value that represents either the cell text or its image name, depending on the value of the `cellType` parameter.

`alignment` ContentAlignment

A ContentAlignment enumeration value that represents the alignment of the legend cell content.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a cell column in a legend, which is used to extend the functionality of the default legend.

C#

```
public class LegendCellColumn :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → LegendCellColumn

Remarks

Any number of cell columns can be placed in any order, which gives you the flexibility to extend the abilities of the default legend entries.

The legend cell columns are only applicable to items that are automatically generated from series or data points.

Constructors

LegendCellColumn()	Initializes a new instance of the LegendCellColumn class.
LegendCellColumn(String, LegendCellColumnType, String)	Initializes a new instance of the LegendCellColumn class with the specified header text, column type and column cell text.
LegendCellColumn(String, LegendCellColumnType, String, ContentAlignment)	Initializes a new instance of the LegendCellColumn class with the specified header text, column type, column cell text and content alignment.

Properties

Alignment	Gets or sets the legend column content alignment.
BackColor	Gets or sets the background color of the legend cell column.
ColumnType	Gets or sets the legend cell column type.
Font	Gets or sets the font for the legend column text.
ForeColor	Gets or sets the text color of the legend cell column.
HeaderAlignment	Gets or sets the horizontal text alignment of the legend cell column header.
HeaderBackColor	Gets or sets the background color for the legend cell column header.
HeaderFont	Gets or sets the font used in the legend column header.
HeaderForeColor	Gets or sets the color of the legend column header text (HeaderText property).
HeaderText	Gets or sets the text for the legend cell column header.
Legend	Gets the legend in which this legend cell column belongs.
Margins	Gets or sets the margins of the legend cell column.
MaximumWidth	Gets or sets the maximum width of the legend cell column.
MinimumWidth	Gets or sets the minimum width of the legend cell column.
Name	Gets or sets the name of the legend cell column.
SeriesSymbolSize	Gets or sets the size of the series symbol displayed in a legend cell column.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the text of the legend cell column.
ToolTip	Gets or sets the tooltip text for the legend cell column.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the LegendCellColumn and optionally releases the managed resources.

Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ShouldSerializeMargins()	Determines whether the property should be serialized.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the LegendCellColumn class.

Overloads

LegendCellColumn()	Initializes a new instance of the LegendCellColumn class.
LegendCellColumn(String, LegendCellColumnType, String)	Initializes a new instance of the LegendCellColumn class with the specified header text, column type and column cell text.
LegendCellColumn(String, LegendCellColumnType, String, ContentAlignment)	Initializes a new instance of the LegendCellColumn class with the specified header text, column type, column cell text and content alignment.

LegendCellColumn()

Initializes a new instance of the LegendCellColumn class.

C#

```
public LegendCellColumn ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn(String,

LegendCellColumnType, String)

Initializes a new instance of the LegendCellColumn class with the specified header text, column type and column cell text.

C#

```
public LegendCellColumn (string headerText,
System.Windows.Forms.DataVisualization.Charting.LegendCellColumnType
columnType, string text);
```

Parameters

`headerText` String

The text for the legend cell column header.

`columnType` LegendCellColumnType

A LegendCellColumnType enumeration value that represents the legend cell column type.

`text` String

The text for the legend cell column.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn(String, LegendCellColumnType, String, ContentAlignment)

Initializes a new instance of the LegendCellColumn class with the specified header text, column type, column cell text and content alignment.

C#

```
public LegendCellColumn (string headerText,
System.Windows.Forms.DataVisualization.Charting.LegendCellColumnType
columnType, string text, System.Drawing.ContentAlignment alignment);
```

Parameters

`headerText` String

The text for the legend cell column header.

`columnType` LegendCellColumnType

A LegendCellColumnType enumeration value that represents the legend cell column type.

`text` String

The text for the legend cell column.

`alignment` ContentAlignment

The legend column content alignment.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the legend column content alignment.

C#

```
public virtual System.Drawing.ContentAlignment Alignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment value that represents the alignment of the legend cell contents. The default value is MiddleCenter.

Remarks

This property is used to align text or series symbols within the legend cell column.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the legend cell column.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value used to draw the legend cell column background.

The default value is Empty.

Attributes TypeConverterAttribute

Remarks

This property overrides the BackColor property of the legend that the legend cell column belongs to. It also overrides the InterlacedRows property of the legend.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.ColumnType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the legend cell column type.

C#

```
public virtual
System.Windows.Forms.DataVisualization.Charting.LegendCellColumnType
ColumnType { get; set; }
```

Property Value

LegendCellColumnType

A LegendCellColumnType enumeration value that represents the legend cell column type. This value represents the type that will be used to define the legend cell column contents. The default value is Text.

Remarks

This property determines the type of content that will be displayed in each individual legend cell column. There are two possible values:

- Text: By default, the #LEGENDTEXT keyword is used to place the LegendText value in each cell column. However, you can use the Text property to set any keyword(s) that will be replaced by a value of the corresponding series.
- SeriesSymbol: The legend cell will contain a series symbol, such as a rectangle, line, line with marker, or marker, of the same fill color as its corresponding series.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font for the legend column text.

C#

```
public virtual System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object that specifies the font for the legend column text. The default value is `null`.

Remarks

This property is only applicable when the LegendCellColumnType is set to Text, and a `string` value is defined for the Text property. You can use the Font property to set font properties of the legend cell column.

If you attempt to change the properties of this Font object at run time, a compilation error will occur. To set font properties at run time, you must create a new Font object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text color of the legend cell column.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value used to draw the legend cell column text. The default value is Empty.

Attributes TypeConverterAttribute

Remarks

This property is only applicable if the LegendCellColumnType is set to Text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.HeaderAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the horizontal text alignment of the legend cell column header.

C#

```
public System.Drawing.StringAlignment HeaderAlignment { get; set; }
```

Property Value

StringAlignment

A StringAlignment enumeration value that specifies the horizontal text alignment of the legend cell column header.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.HeaderBackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color for the legend cell column header.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color HeaderBackColor { get; set; }
```

Property Value

Color

A Color value used to draw the background of the legend cell column header. The default value is Empty.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.HeaderFont Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font used in the legend column header.

C#

```
public virtual System.Drawing.Font HeaderFont { get; set; }
```

Property Value

Font

A Font object that represents the font used in the legend column header. The default value is "Microsoft Sans Serif, 8pt, Bold".

Remarks

This property can be used to set font properties of the legend cell column header.

If you attempt to change the properties of this Font object at run time, a compilation error will occur. To set font properties at run time, you must create a new Font object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.HeaderForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the legend column header text (HeaderText property).

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color HeaderForeColor { get; set; }
```

Property Value

Color

A Color value that specifies the color of the legend column header text.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.HeaderText Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text for the legend cell column header.

C#

```
public virtual string HeaderText { get; set; }
```

Property Value

String

A `string` value that represents the legend cell column header text. The default value is a zero-length string.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.Legend Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the legend in which this legend cell column belongs.

C#

```
[System.ComponentModel.Browsable(false)]
public virtual System.Windows.Forms.DataVisualization.Charting.Legend Legend
{ get; }
```

Property Value

Legend

A Legend object that represents the legend which this legend cell column belongs to.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.Margins Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the margins of the legend cell column.

C#

```
public virtual System.Windows.Forms.DataVisualization.Charting.Margins
Margins { get; set; }
```

Property Value

Margins

A Margins object that represents the top, bottom, left, and right values of the legend cell margins. The default values for top, bottom, left and right are "0,0,15,15".

Remarks

This property is measured as a percentage of the legend font size.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.MaximumWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the maximum width of the legend cell column.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.IntNanValueConverter))]
public virtual int MaximumWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the maximum width of the legend cell column. The default value is minus one (-1), which means that the maximum width is automatically calculated.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.MinimumWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the minimum width of the legend cell column.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.IntNanValueConverter))]
public virtual int MinimumWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the minimum width of the legend cell column. The default value is minus one (-1), which means that the minimum width is automatically calculated.

Attributes TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the legend cell column.

C#

```
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the text name of the legend cell column.

Remarks

The name of the legend cell must be unique; otherwise an exception will be thrown.

The Name property can be used as the indexer when retrieving a LegendCellColumn object from the LegendCellColumnCollection.

If you set this property to a zero-length string, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.SeriesSymbolSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the series symbol displayed in a legend cell column.

C#

```
public virtual System.Drawing.Size SeriesSymbolSize { get; set; }
```

Property Value

Size

A Size object that represents the size of the series symbol. The default values for the width and height of the symbol are 200 and 70, respectively.

Remarks

The size of the series symbol is measured as a percentage of the legend font size.

This property is only applicable if the LegendCellColumnType property is set to SeriesSymbol.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.Text Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text of the legend cell column.

C#

```
public virtual string Text { get; set; }
```

Property Value

String

A `string` value that represents the text for the legend cell column. The default value is a zero-length string.

Remarks

This property is only applicable if the LegendCellColumnType is set to Text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip text for the legend cell column.

C#

```
public virtual string ToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip for the legend cell column. The default value is a zero-length string.

Remarks

If the `ToolTip` property is set for both a legend item and a legend cell, the legend cell property has precedence.

This property allows you to set special keywords to display different types of chart-related data.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the unmanaged resources used by the LegendCellColumn and optionally releases the managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumn.ShouldSerializeMargins Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Determines whether the property should be serialized.

C#

```
public bool ShouldSerializeMargins ();
```

Returns

Boolean

`true` if property should be serialized; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumnCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Stores LegendCellColumn objects, which are applied to default legend items.

C#

```
public class LegendCellColumnCollection :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<
System.Windows.Forms.DataVisualization.Charting.LegendCellColumn>
```

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<LegendCellColumn> → LegendCellColumnCollection

Remarks

The LegendCellColumnCollection class stores LegendCellColumn objects.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Item[String]	Gets or sets the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)
NamePrefix	Gets the name prefix that is used to create unique chart element names. (Inherited from ChartNamedElementCollection<T>)

Methods

Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the LegendCellColumnCollection and optionally releases the managed resources.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds the chart element by the specified name. (Inherited from ChartNamedElementCollection<T>)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(String)	Finds the index of the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)

InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index. (Inherited from ChartNamedElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection. (Inherited from ChartNamedElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection. (Inherited from ChartNamedElementCollection<T>)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection. (Inherited from ChartNamedElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element. (Inherited from ChartNamedElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe).

	(Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T>

	object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by

	invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource>(IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default

	value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement>	Groups the elements of a sequence according

(IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>,	Produces the set intersection of two

IEnumerable<TSource>, IEquality Comparer<TSource>)	sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.

Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Nullable<Double>>)	of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>,	Projects each element of a sequence into a new form by incorporating the element's

Func<TSource,Int32,TResult>)	index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>,	Returns the only element of a sequence that satisfies a specified condition or a default

Func<TSource,Boolean>)	value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.

Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.

ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the

	source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumnCollection.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the unmanaged resources used by the LegendCellColumnCollection and optionally releases the managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellColumnType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a legend cell column type.

C#

```
public enum LegendCellColumnType
```

Inheritance Object → ValueType → Enum → LegendCellColumnType

Fields

SeriesSymbol	1	The legend cell column contains a series symbol such as a line, rectangle, or line with marker.
Text	0	The legend cell column contains text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCellType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a legend cell type.

C#

```
public enum LegendCellType
```

Inheritance Object → ValueType → Enum → LegendCellType

Fields

Image	2	The legend cell contains an image.
SeriesSymbol	1	The legend cell contains a series symbol, such as a line or rectangle.
Text	0	The legend cell contains text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of Legend objects.

C#

```
public class LegendCollection :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<
System.Windows.Forms.DataVisualization.Charting.Legend>
```

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<Legend> → LegendCollection

Remarks

The LegendCollection class is implemented as the Legends collection property of the root Chart object.

Legend objects can be appended, inserted or removed from the collection at either design time or run time.

The Count property indicates the total number of items in the collection and is commonly used to find the upper bound of the collection.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Item[String]	Gets or sets the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)

Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)
NamePrefix	Gets the name prefix that is used to create unique chart element names. (Inherited from ChartNamedElementCollection<T>)

Methods

Add(String)	Adds the specified Legend object to the end of the collection.
Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds the chart element by the specified name. (Inherited from ChartNamedElementCollection<T>)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance.

	(Inherited from Object)
IndexOf(String)	Finds the index of the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index. (Inherited from ChartNamedElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection. (Inherited from ChartNamedElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection. (Inherited from ChartNamedElementCollection<T>)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection. (Inherited from ChartNamedElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element. (Inherited from ChartNamedElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object.

	(Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.

Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified

	IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>,	Returns the first element of the sequence that satisfies a condition or a default value if no

Func<TSource,Boolean>)	such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.

GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Double>)	of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32

	value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>,	Adds a value to the beginning of the

TSource)	sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an

	exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.

Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>,	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key

Func<TSource,TElement>, IEqualityComparer<TKey>)	selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey>(IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey>(IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement>(IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement>(IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEqualityComparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult>(IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.

AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendCollection.Add(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds the specified Legend object to the end of the collection.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Legend Add (string
name);
```

Parameters

name String

The name of legend to be added.

Returns

Legend

A `string` value that represents the index of the added item.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendImageStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the style to apply to the symbol for a legend item.

C#

```
public enum LegendImageStyle
```

Inheritance Object → ValueType → Enum → LegendImageStyle

Fields

Line	1	The symbol will be a line.
Marker	2	The symbol will be a marker.
Rectangle	0	The symbol will be a rectangle.

Remarks

The LegendImageStyle enumeration represents the style that is used for a legend item, and is applied to the symbol for that item.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a legend item, which is stored in the LegendItemsCollection class.

C#

```
public class LegendItem :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → LegendItem

Remarks

The LegendItem class represents a legend item, which can be either a default or custom item:

- Default legend items are automatically inserted into the legend for either data series or data points. Default entries for pie and doughnut charts represent the plotted data points, whereas entries for all other chart types represent the plotted series. By default, these legend entries use the data point or series color for the symbol color, and the name of the series or data point for the legend item text. Default legend items can be customized or deleted at run time by using the CustomizeLegend event of the Chart control.
- Custom legend items have user-defined text, symbol color, and so forth. Unlike the default entries, they are not connected with the plotted data. Custom legend items can be added at either design time or run time, using the CustomItems property of the Legend object.

Legend symbols can be rectangles, lines or markers, depending on the ImageStyle property setting of a LegendItem object.

The text of a custom legend item is determined by the Name property.

Constructors

LegendItem()	Initializes a new instance of the LegendItem class.
LegendItem(String, Color, String)	Initializes a new instance of the LegendItem class with the specified name, color and image path.

Properties

BackGradientStyle	Gets or sets the orientation for the background gradient of a legend item. Also determines whether a gradient is used.
BackHatchStyle	Gets or sets the hatching style of a legend item.
BackImageTransparentColor	Gets or sets a color which will be replaced with a transparent color when the background image is drawn.
BackSecondaryColor	Gets or sets the secondary color of a legend item.
BorderColor	Gets or sets the border color of a legend item.
BorderDashStyle	Gets or sets the border style of a legend item.
BorderWidth	Gets or sets the border width of a legend item.
Cells	Gets the collection of cells in the legend item.
Color	Gets or sets the color of a legend item.
Enabled	Gets or sets a flag that indicates whether the legend item is enabled.
Image	Gets or sets the image that will be displayed for a legend item symbol.
ImageStyle	Gets or sets the display style of the legend item image.
Legend	Gets or sets the Legend object associated with this legend item.
MarkerBorderColor	Gets or sets the border color of the markers, if used.
MarkerBorderWidth	Gets or sets the border width of the markers, if used. Measured in pixels.
MarkerColor	Gets or sets the color of the markers, if used.
MarkerImage	Gets or sets a marker image of the legend item.
MarkerImageTransparentColor	Gets or sets a color that will be replaced with a transparent color when the marker image is drawn.

MarkerSize	Gets or sets the size of the legend item markers, if used.
MarkerStyle	Gets or sets a legend item marker style. Also used to enable or disable the display of markers.
Name	Gets or sets the name of the legend item.
SeparatorColor	Gets or sets the color of the legend item separator.
SeparatorType	Gets or sets the separator style of the legend item.
SeriesName	Gets or sets the name of the Series associated with a default legend item.
SeriesPointIndex	Gets or sets the index of a DataPoint object associated with a default legend item.
ShadowColor	Gets or sets the shadow color of a legend item.
ShadowOffset	Gets or sets the shadow offset, in pixels, of a legend item.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
ToolTip	Gets or sets the tooltip of the legend item.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the LegendItem and optionally releases the managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the LegendItem class.

Overloads

LegendItem()	Initializes a new instance of the LegendItem class.
LegendItem(String, Color, String)	Initializes a new instance of the LegendItem class with the specified name, color and image path.

LegendItem()

Initializes a new instance of the LegendItem class.

C#

```
public LegendItem ();
```

Remarks

Use this constructor to create and initialize a new instance of the LegendItem class.

Note that you can also add custom legend items using the `Add` and `Insert` methods associated with the CustomItems collection property of the Legend object.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem(String, Color, String)

Initializes a new instance of the LegendItem class with the specified name, color and image path.

```
C#

public LegendItem (string name, System.Drawing.Color color, string
image);
```

Parameters

`name` String

The name of the legend item.

`color` Color

The color of the legend item.

`image` String

The path of the image to be displayed for the legend item.

Remarks

Only the `image` or `color` parameter needs to be specified. If both are specified, the `image` parameter has precedence.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the orientation for the background gradient of a legend item. Also determines whether a gradient is used.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration that specifies the orientation for the background gradient of a legend item. The default value is None.

Attributes BindableAttribute

Remarks

Use the BackGradientStyle property to determine whether a title will display a gradient, and to determine the orientation of the gradient if one is used.

This property is only implemented if the ImageStyle property is set to Rectangle.

For a listing of available patterns, refer to the GradientStyle enumeration.

To use gradient colors:

- Use the Color property to set the primary color.
- Use the BackGradientStyle property to set the secondary color.
- Use this BackGradientStyle property to set the desired gradient style.

> **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatching style of a legend item.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration that specifies the hatching style of a legend item. The default value is None.

Attributes BindableAttribute

Remarks

The ChartHatchStyle enumeration specifies the hatch pattern used by a brush of type HatchBrush. The hatch pattern consists of a solid background color and lines drawn over the background.

For a listing of available hatching styles, refer to the ChartHatchStyle enumeration.

> ⓘ **Note**
>
> The color of hatching lines is determined by the **BackSecondaryColor** property.

> ⓘ **Note**

This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.BackImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color which will be replaced with a transparent color when the background image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that will be displayed as transparent when the background image is drawn.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property is applicable only to LegendCell objects with a LegendCellType of either SeriesSymbol or Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color of a legend item.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value used for the secondary color of background with hatching or gradient fill. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The BackSecondaryColor property is one of several related properties that determine the secondary background color of many Chart objects; it is used when gradient colors are being applied.

This property is only implemented if the ImageStyle property is set to Rectangle.

You can set BackSecondaryColor to any valid ARGB (alpha, red, green, blue) value.

If hatching is used, the BackSecondaryColor property determines the color of the hatching lines for the legend item.

To use gradient colors for a legend cell's background:

To use gradient colors:

- Use the Color property to set the primary color.
- Use the BackGradientStyle property to set the secondary color.
- Use the BackGradientStyle property to set the desired gradient style.

> ⓘ **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.BorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of a legend item.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderColor { get; set; }
```

Property Value

Color

A Color value used to draw the border color of a legend item. The default color is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set this property to any valid ARGB (alpha, red, green, blue) value.

> ⓘ **Note**
>
> If you use a marker for a custom legend item. this property will set the border color of the marker. However, if the **MarkerBorderColor** property is also set, its value has precedence.

> ⓘ **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.BorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border style of a legend item.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that determines the border style of the legend item.

Attributes BindableAttribute

Remarks

This property determines the line style used for a border. The line can be solid, dotted, and so forth.

For other border styles, see the ChartDashStyle enumeration topic.

> **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.BorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of a legend item.

C#

```
[System.ComponentModel.Bindable(true)]
public int BorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that determines the width, in pixels, of the legend item border.

Attributes BindableAttribute

Remarks

The BorderWidth property can also be used to set the width of lines when, for example, custom legend items are displayed with a Line style.

Note This property is applicable only to LegendCell objects with a LegendCellType of either SeriesSymbol or Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.Cells Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the collection of cells in the legend item.

C#

```
public System.Windows.Forms.DataVisualization.Charting.LegendCellCollection
Cells { get; }
```

Property Value

LegendCellCollection

A LegendCellCollection object that represents the collection of cells in the legend item.

Remarks

A legend item consists of one or more legend cells, each represented by a LegendCell object and stored in a LegendCellCollection.

> **Note**
>
> This property is applicable to the entire **LegendItem**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.Color Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of a legend item.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color Color { get; set; }
```

Property Value

Color

A Color value that is used to draw the legend item. The default color is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This Color property is one several related properties that get or set the color of many chart elements; it can be set to any ARGB (alpha, red, green, blue) value.

If this property is set for both a data point and its series, the data point color will have precedence. The series color will only be used for data points that do not define their own color.

If a custom legend item uses a marker, this property will determine the marker color. However, if the MarkerColor property is set, its value then has precedence.

This property will be ignored if the Image property specifies an image to be used for the legend item symbol.

Note

This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the legend item is enabled.

C#

```
public bool Enabled { get; set; }
```

Property Value

Boolean

`true` if the legend item is enabled; otherwise, `false`. The default value is `true`.

Remarks

This property is applicable to the entire LegendItem.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.Image Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the image that will be displayed for a legend item symbol.

C#

```
[System.ComponentModel.Bindable(true)]
public string Image { get; set; }
```

Property Value

String

A `string` value that represents the URL of an image file. The default value is an empty string.

Attributes BindableAttribute

Remarks

This property determines the image file to be used for the LegendItem symbol.

The path can be an absolute or a relative URL.

If you set a background image, that image will be displayed regardless of the ImageStyle property setting.

Set the BackImageTransparentColor property to enable one color used by the image to appear transparent.

> **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.ImageStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the display style of the legend item image.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.LegendImageStyle
ImageStyle { get; set; }
```

Property Value

LegendImageStyle

A LegendImageStyle enumeration value that specifies the display style of the legend item image.

Attributes BindableAttribute

Remarks

These styles determine how the item's corresponding symbol will be displayed.

A Line style can be combined with a Marker style. To accomplish this, first set the ImageStyle property to Line. Then use the MarkerImage property to specify a marker image, or the MarkerStyle property to specify a marker style. The resulting symbol will be a combination of a line and an image or marker.

> ⓘ **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.Legend Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Legend object associated with this legend item.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public System.Windows.Forms.DataVisualization.Charting.Legend Legend { get;
}
```

Property Value

Legend

A Legend object associated with this legend item.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.MarkerBorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of the markers, if used.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color MarkerBorderColor { get; set; }
```

Property Value

Color

A Color value that represents the ARGB (alpha, red, green, blue) color of a marker border.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

To use markers, set the MarkerStyle property to a value other than None.

Note that markers will not be applied to pie, doughnut, and any stacked chart types.

For custom legends, the BorderColor property will also set the border color of markers. The MarkerBorderColor property setting has precedence over the BorderColor property setting.

> **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.MarkerBorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of the markers, if used. Measured in pixels.

C#

```
public int MarkerBorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the border width of the markers. The default value is one (1) pixel.

Remarks

This property is applicable only to LegendCell objects with a LegendCellType of either SeriesSymbol or Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.MarkerColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the markers, if used.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color MarkerColor { get; set; }
```

Property Value

Color

A Color value that represents the color of the markers. The default color is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property is applicable only to LegendCell objects with a LegendCellType of either SeriesSymbol or Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.MarkerImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a marker image of the legend item.

C#

```
[System.ComponentModel.Bindable(true)]
public string MarkerImage { get; set; }
```

Property Value

String

A `string` value that represents the URL of an image file.

Attributes BindableAttribute

Remarks

Represents a path to an image file, which can be an absolute or relative URL.

To use markers, set the MarkerStyle property to a value other than None.

The MarkerImage property takes precedence over the MarkerColor property.

A marker image can be any valid graphics file supported by .NET Framework, including icons.

To set one color of the marker image to be transparent, use the MarkerImageTransparentColor property.

> **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.MarkerImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color that will be replaced with a transparent color when the marker image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color MarkerImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that represents an ARGB (alpha, red, green, blue) color that will be replaced with a transparent color when the marker image is drawn.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The MarkerImageTransparentColor property is used to select one color in a marker image that will be displayed as transparent.

You can set this property to any valid ARGB (alpha, red, green, blue) value.

> **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.MarkerSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of the legend item markers, if used.

C#

```
[System.ComponentModel.Bindable(true)]
public int MarkerSize { get; set; }
```

Property Value

Int32

An `integer` expression that represents the height and width of the plotting area of markers, in pixels. The default value is 5 pixels.

Attributes BindableAttribute

Remarks

To use markers, set the MarkerStyle property to a value other than None.

Note that markers will not be applied to data points of pie, doughnut and any stacked chart types.

> **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.MarkerStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a legend item marker style. Also used to enable or disable the display of markers.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.MarkerStyle
MarkerStyle { get; set; }
```

Property Value

MarkerStyle

A MarkerStyle integer enumeration that specifies a legend item marker style. The default value is None.

Attributes BindableAttribute

Remarks

To disable markers, set this property to None.

> ⓘ **Note**
>
> This property is applicable only to **LegendCell** objects with a **LegendCellType** of either **SeriesSymbol** or **Image**.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the legend item.

C#

```
[System.ComponentModel.Bindable(true)]
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the name of the legend item.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.SeparatorColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the legend item separator.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color SeparatorColor { get; set; }
```

Property Value

Color

A Color value used to draw the color of the legend item separator. The default value is Black.

Attributes TypeConverterAttribute

Remarks

This property is applicable to the entire LegendItem.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.SeparatorType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the separator style of the legend item.

C#

```
public System.Windows.Forms.DataVisualization.Charting.LegendSeparatorStyle
SeparatorType { get; set; }
```

Property Value

LegendSeparatorStyle

A LegendSeparatorStyle enumeration value that specifies the separator style of the legend item.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.SeriesName Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the Series associated with a default legend item.

C#

```
[System.ComponentModel.Browsable(false)]
public string SeriesName { get; set; }
```

Property Value

String

A `string` value that determines the name of the series associated with a default legend item. The default value is a zero-length string.

Attributes BrowsableAttribute

Remarks

This property should only be used in the CustomizeLegend event, to determine which data series a default legend entry represents.

The SeriesName property applies to all charts except pie and sphere charts; their default legend items correspond to plotted data points, and not to plotted series. When you deal with pie and sphere charts, use the SeriesPointIndex property instead.

This property should not be set by the developer.

> **Note**
>
> This property is applicable to the entire **LegendItem**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.SeriesPointIndex Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the index of a DataPoint object associated with a default legend item.

C#

```
[System.ComponentModel.Browsable(false)]
public int SeriesPointIndex { get; set; }
```

Property Value

Int32

An `integer` value that represents the location of the default legend's associated data point in a zero-based index. All chart types except for pie and doughnut charts will return a negative one (-1).

Attributes BrowsableAttribute

Remarks

This property should only be used in the CustomizeLegend event to determine the DataPoint object that is represented by a default pie or sphere chart legend entry.

Chart types other than pie or sphere charts will return a negative one (-1), because the default legend items correspond to plotted series, and not plotted data points. Use the SeriesName property for these chart types.

This property should not be set by the developer.

> **Note**
>
> This property is applicable to the entire **LegendItem**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.ShadowColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow color of a legend item.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ShadowColor { get; set; }
```

Property Value

Color

A Color value that represents the shadow color of a legend item. The default color is Black.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

Gets or sets the shadow color of several chart elements, and can be set to any valid ARGB (alpha, red, green, blue) value.

The alpha value can be used to achieve a "realistic" shadowing effect; try using a color of "128,0,0,0" for shadows.

To implement a shadow, set the ShadowOffset property to a value other than zero (0).

Note This property is applicable only to LegendCell objects with a LegendCellType of either SeriesSymbol or Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.ShadowOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow offset, in pixels, of a legend item.

C#

```
[System.ComponentModel.Bindable(true)]
public int ShadowOffset { get; set; }
```

Property Value

Int32

An `integer` value that represents the shadow offset, in pixels, of the relevant chart element.

Attributes BindableAttribute

Remarks

To implement shadowing, use a value other than zero (0). A positive value will result in a shadow that shifts down and to the right. A negative value will result in a shadow that shifts up and to the left.

Note This property is applicable only to LegendCell objects with a LegendCellType of either SeriesSymbol or Image.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip of the legend item.

C#

```
[System.ComponentModel.Bindable(true)]
public string ToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip for the legend item.

Attributes BindableAttribute

Remarks

This property allows you to set special keywords to display different types of chart-related data. For more information, see Keywords [rs_vsDataVis] .

> **Note**
>
> This property is applicable to the entire **LegendItem**.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItem.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the unmanaged resources used by the LegendItem and optionally releases the managed resources.

```
C#
protected override void Dispose (bool disposing);
```

Parameters

disposing Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItemOrder Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the order in which items are added to the legend.

C#

```
public enum LegendItemOrder
```

Inheritance Object → ValueType → Enum → LegendItemOrder

Fields

Auto	0	The chart automatically determines the order in which items are added to the legend.
ReversedSeriesOrder	2	Items are added to the legend in the reverse order of items added to the chart series.
SameAsSeriesOrder	1	Items are added to the legend in the same order as items added to the chart series.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItemsCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of LegendItem objects.

C#

```
public class LegendItemsCollection :
System.Windows.Forms.DataVisualization.Charting.ChartElementCollection<Syste
m.Windows.Forms.DataVisualization.Charting.LegendItem>
```

Inheritance Object → Collection<T> → ChartElementCollection<LegendItem> → LegendItemsCollection

Remarks

The LegendItemsCollection class represents a collection of LegendItem objects, which in turn represent items in a legend.

This collection class is exposed in two ways:

- As the CustomItems object property of the Legend object, which can be used to add, delete or modify custom legend items. The term "custom" refers to legend items that have user-defined text, symbol color, and so forth. In comparison, "default" legend items are items that are automatically inserted into the legend and represent a data series or data point. Default legend item entries for pie and doughnut charts represent the plotted data points, whereas item entries for all other chart types represent the plotted series. The CustomItems property is available at both design time and run time.

- In the CustomizeLegend event, as the LegendItems property of the CustomizeLegendEventArgs parameter. The LegendItems property stores both default legend items and custom legend items that have been added to the Chart control at design time. Legend items can be modified or deleted; in addition, their order can be changed.

To add legend items to the collection, use one of the overloaded `Add` and `Insert` methods in this class.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)

Methods

Add(Color, String)	Adds a custom LegendItem object to the end of the collection.
Add(String, String)	Adds a custom LegendItem object with the specified image to the end of the collection.
Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current

	object. (Inherited from Object)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, Color, String)	Inserts a LegendItem object with the specified text and symbol color into the collection at the specified location.
Insert(Int32, String, String)	Inserts a LegendItem object with the specified text and symbol image into the collection at the specified location.
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts an element into the Collection<T> at the specified index location. (Inherited from ChartElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the element at the specified index location of the Collection<T>. (Inherited from ChartElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
Reverse()	Reverses the order of items in the collection.

SetItem(Int32, T)	Replaces the element at the specified index location. (Inherited from ChartElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)

IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking

	a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>,	Concatenates two sequences.

IEnumerable<TSource>)	
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>,	Returns the first element in a sequence that

Func<TSource,Boolean>)	satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each

	group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.

Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum

	Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.

OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>,	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.

IEnumerable<TSource>, IEquality Comparer<TSource>)	
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>,	Computes the sum of the sequence of Int64

Func<TSource,Int64>)	values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey>	Creates a Dictionary<TKey,TValue> from an

(IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.

Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.

Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItemsCollection.Add Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a custom LegendItem object to the end of the collection.

Overloads

Add(Color, String)	Adds a custom LegendItem object to the end of the collection.
Add(String, String)	Adds a custom LegendItem object with the specified image to the end of the collection.

Add(Color, String)

Adds a custom LegendItem object to the end of the collection.

C#

```
public int Add (System.Drawing.Color color, string text);
```

Parameters

`color` Color

The symbol color of the legend item.

`text` String

The legend item text.

Returns

Int32

An `integer` that represents the location in the zero-based index where the item was inserted into the collection.

Remarks

Appends a custom legend item with the specified text and symbol color to the end of the collection.

To add a custom legend item that uses an image for its symbol, call the Add(String, String) method.

To insert a custom LegendItem into the collection, use one of the Insert methods of this class.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Add(String, String)

Adds a custom LegendItem object with the specified image to the end of the collection.

C#

```
public int Add (string image, string text);
```

Parameters

`image` String

The absolute or relative URL of the image file to be used for the legend item symbol.

`text` String

The legend item text.

Returns

Int32

An `integer` that represents the location in the zero-based index where the item was inserted into the collection.

Remarks

Appends a custom legend item to the collection with the specified text and image to be used as the symbol for the legend item.

To add a custom legend item that uses a color for its symbol, call the Add(Color, String) method.

To insert a LegendItem object into the collection use one of the Insert methods of this class.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItemsCollection.Insert Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Inserts a custom LegendItem object into the collection at the given location.

Overloads

Insert(Int32, Color, String)	Inserts a LegendItem object with the specified text and symbol color into the collection at the specified location.
Insert(Int32, String, String)	Inserts a LegendItem object with the specified text and symbol image into the collection at the specified location.

Insert(Int32, Color, String)

Inserts a LegendItem object with the specified text and symbol color into the collection at the specified location.

C#

```
public void Insert (int index, System.Drawing.Color color, string text);
```

Parameters

index Int32

The location in the zero-based index where the LegendItem object is to be inserted.

color Color

The symbol color of the legend item.

text String

The legend item text.

Remarks

Inserts a custom legend item into the collection at the specified location using the specified text and symbol color.

To insert a legend item that uses an image for its symbol, use the Insert method.

Use the Add method to append a LegendItem object that uses a color for its symbol to the end if the collection.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Insert(Int32, String, String)

Inserts a LegendItem object with the specified text and symbol image into the collection at the specified location.

C#

```
public void Insert (int index, string image, string text);
```

Parameters

`index` Int32

The location in the zero-based index where the LegendItem object is to be inserted.

`image` String

The absolute or relative URL of the image file to be used for the legend item symbol.

`text` String

The legend item text.

Remarks

Inserts a custom legend item into the collection at the specified location using the specified text and image for the symbol.

To insert a legend item that uses a color for its symbol, use the Insert method.

Use the Add method to append a LegendItem object with a symbol image to the end if the collection.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendItemsCollection.Reverse Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Reverses the order of items in the collection.

C#

```
public void Reverse ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendSeparatorStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the styles for legend separator lines.

C#

```
public enum LegendSeparatorStyle
```

Inheritance Object → ValueType → Enum → LegendSeparatorStyle

Fields

DashLine	4	Legend separator is a dashed line.
DotLine	5	Legend separator is a dotted line.
DoubleLine	3	Legend separator is a double line.
GradientLine	6	Legend separator is a solid line with a gradient on the ends.
Line	1	Legend separator is a solid line.
None	0	No separator will be shown.
ThickGradientLine	7	Legend separator is a thick solid line with a gradient on the ends.
ThickLine	2	Legend separator is a thick solid line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the legend style.

C#

```
public enum LegendStyle
```

Inheritance Object → ValueType → Enum → LegendStyle

Fields

Column	0	Legend items are displayed using one column and multiple rows.
Row	1	Legend items are displayed using multiple columns and one row.
Table	2	Legend items are displayed using multiple columns and multiple rows.

Remarks

The LegendStyle enumeration represents a legend style, and is used in conjunction with the LegendStyle property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LegendTableStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the legend table style.

C#

```
public enum LegendTableStyle
```

Inheritance Object → ValueType → Enum → LegendTableStyle

Fields

Auto	0	The legend table style is automatically determined by the Chart control.
Tall	2	Legend items will be fit vertically within the legend. This style is preferred when Docking is set to Left or Right.
Wide	1	Legend items will be fit horizontally within the legend. This style is preferred when Docking is set to Top or Bottom.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LightStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a lighting style for a three-dimensional (3D) chart area.

C#

```
public enum LightStyle
```

Inheritance Object → ValueType → Enum → LightStyle

Fields

None	0	No lighting is applied.
Realistic	2	A realistic lighting style is applied, where the hue of all chart area elements changes depending on the amount of rotation.
Simplistic	1	A simplistic lighting style is applied, where the hue of all chart area elements is fixed.

Remarks

The LightStyle enumeration is used with the LightStyle property to set the lighting style of a 3D chart area.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnchorCapStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the shape - that is, the anchor cap - to draw on the point end of the label callout line.

C#

```
public enum LineAnchorCapStyle
```

Inheritance Object → ValueType → Enum → LineAnchorCapStyle

Fields

Arrow	1	Anchor cap will be drawn with an arrowhead.
Diamond	2	Anchor cap will be drawn with a diamond-shaped end.
None	0	Anchor cap will not be drawn.
Round	4	Anchor cap will be drawn with a round end.
Square	3	Anchor cap will be drawn with a square head.

Remarks

The LineAnchorCapStyle enumeration represents the shape to be drawn on the point end of the label callout line. It is used in conjunction with the CalloutStyle and the CalloutLineAnchorCapStyle properties. If a callout line is not used, this enumeration will have no effect.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a line annotation.

C#

```
public class LineAnnotation :
System.Windows.Forms.DataVisualization.Charting.Annotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → LineAnnotation

Derived System.Windows.Forms.DataVisualization.Charting.HorizontalLineAnnotation
System.Windows.Forms.DataVisualization.Charting.VerticalLineAnnotation

Constructors

LineAnnotation()	Initializes a new instance of the LineAnnotation class.

Properties

Alignment	This property is not applicable to the line annotation type.
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)

AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the position alignment of an annotation to the anchor point.
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the type of annotation.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached.

	(Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	This property is not applicable to the line annotation type.
BackGradientStyle	This property is not applicable to the line annotation type.
BackHatchStyle	This property is not applicable to the line annotation type.
BackSecondaryColor	This property is not applicable to the line annotation type.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
EndCap	Gets or sets a style for the anchor cap at the end of the annotation line.
Font	This property is not applicable to the line annotation type.
ForeColor	This property is not applicable to the line annotation type.
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsInfinitive	Gets or sets a flag that indicates whether an infinite line should be drawn.
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates.
LineColor	Gets or sets the line color of an annotation. (Inherited from Annotation)
LineDashStyle	Gets or sets the line style of an annotation. (Inherited from Annotation)
LineWidth	Gets or sets the line width of an annotation. (Inherited from Annotation)
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation.

	(Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
StartCap	Gets or sets a style for the anchor cap at the start of an annotation line.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	This property is not applicable to the line annotation type.
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)

Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends the end user placement of an annotation. (Inherited from Annotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, Data Point)	Anchors an annotation to two data points. (Inherited from Annotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the LineAnnotation class.

C#

```
public LineAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the line annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.ContentAlignment Alignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment enumeration value.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.AnchorAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the position alignment of an annotation to the anchor point.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.ContentAlignment AnchorAlignment { get; set;
}
```

Property Value

ContentAlignment

A ContentAlignment value that represents the annotation alignment to the anchor point.

Attributes BrowsableAttribute

Remarks

The annotation must be anchored using either AnchorDataPoint, or the AnchorX and AnchorY properties. Its X and Y properties must be set to NaN.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the type of annotation.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

For internal use, should not be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the line annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the line annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration value.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the line annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle value.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the line annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.EndCap Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a style for the anchor cap at the end of the annotation line.

C#

```
public virtual
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle EndCap {
get; set; }
```

Property Value

LineAnchorCapStyle

A LineAnchorCapStyle enumeration value that represents the style to apply to the anchor cap at the end of the line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the line annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the line annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.IsInfinitive Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether an infinite line should be drawn.

C#

```
public virtual bool IsInfinitive { get; set; }
```

Property Value

Boolean

`true` if a line should be drawn infinitely through two points provided, otherwise `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.IsSizeAlwaysRelative Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates.

C#

```
public override bool IsSizeAlwaysRelative { get; set; }
```

Property Value

Boolean

`true` if the Width and Height of an annotation are always in chart relative coordinates, otherwise `false`.

Remarks

The width and height of an annotation may be set in relative chart or in axes coordinates. By default, relative chart coordinates are used. To use axes coordinates for the annotation size, set the IsSizeAlwaysRelative property to `false`. Then anchor the annotation to a data point or set the AxisX or AxisY properties.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.StartCap Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a style for the anchor cap at the start of an annotation line.

C#

```
public virtual
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle StartCap
{ get; set; }
```

Property Value

LineAnchorCapStyle

A LineAnchorCapStyle enumeration value that represents the style to apply to the anchor cap at the start of the line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

LineAnnotation.TextStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the line annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Windows.Forms.DataVisualization.Charting.TextStyle
TextStyle { get; set; }
```

Property Value

TextStyle

A TextStyle enumeration value.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the margins for various chart elements.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.MarginExpandableObjectConverter))]
public class Margins
```

Inheritance Object → Margins

Attributes TypeConverterAttribute

Constructors

Margins()	Initializes a new instance of the Margins class.
Margins(Int32, Int32, Int32, Int32)	Initializes a new instance of the Margins class with the specified top, bottom, left and right coordinates of the margin.

Properties

Bottom	Gets or sets the bottom margin.
Left	Gets or sets the left margin.
Right	Gets or sets the right margin.
Top	Gets or sets the top margin.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object.
GetHashCode()	Gets the hash code for the Margins object.
GetType()	Gets the Type of the current instance. (Inherited from Object)
IsEmpty()	Determines whether the margins are empty.
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToRectangleF()	Converts a Margins object to a RectangleF object.
ToString()	Converts a Margins object to its `string` representation.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Margins class.

Overloads

Margins()	Initializes a new instance of the Margins class.
Margins(Int32, Int32, Int32, Int32)	Initializes a new instance of the Margins class with the specified top, bottom, left and right coordinates of the margin.

Margins()

Initializes a new instance of the Margins class.

C#

```
public Margins ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins(Int32, Int32, Int32, Int32)

Initializes a new instance of the Margins class with the specified top, bottom, left and right coordinates of the margin.

```C#
public Margins (int top, int bottom, int left, int right);
```

Parameters

`top` Int32

Top margin.

`bottom` Int32

Bottom margin.

`left` Int32

Left margin.

`right` Int32

Right margin.

Remarks

When used for legend cells, the margins are measured as a percentage of the legend's font size.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.Bottom Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the bottom margin.

```
C#
public int Bottom { get; set; }
```

Property Value

Int32

Returns an `integer` value. The default value is 0.

Remarks

When used for legend cells, the margins are measured as a percentage of the legend's font size.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.Left Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the left margin.

```
C#
public int Left { get; set; }
```

Property Value

Int32

An `integer` value. The default value is 0.

Remarks

When used for legend cells, the margins are measured as a percentage of the legend's font size.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.Right Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the right margin.

```
C#
public int Right { get; set; }
```

Property Value

Int32

An `integer` value. The default value is 0.

Remarks

When used for legend cells, the margins are measured as a percentage of the legend's font size.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.Top Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the top margin.

```
C#
public int Top { get; set; }
```

Property Value

Int32

An `integer` value. The default value is 0.

Remarks

When used for legend cells, the margins are measured as a percentage of the legend's font size.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.Equals(Object) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Determines whether the specified object is equal to the current object.

C#

```
public override bool Equals (object obj);
```

Parameters

obj Object

The object to compare with the current object.

Returns

Boolean

`true` if the specified object is equal to the current object; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.GetHashCode Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the hash code for the Margins object.

C#

```
public override int GetHashCode ();
```

Returns

Int32

An `integer` value that represents the hash code for a Margins object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.IsEmpty Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Determines whether the margins are empty.

C#

```
public bool IsEmpty ();
```

Returns

Boolean

`true` if all margins values are zero; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.ToRectangleF Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Converts a Margins object to a RectangleF object.

C#

```
public System.Drawing.RectangleF ToRectangleF ();
```

Returns

RectangleF

A RectangleF object that has margins.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Margins.ToString Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Converts a Margins object to its `string` representation.

```
C#

public override string ToString ();
```

Returns

String

A `string` value that represents this object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

MarkerStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a style for markers.

C#

```
public enum MarkerStyle
```

Inheritance Object → ValueType → Enum → MarkerStyle

Fields

Circle	2	A circular marker is displayed.
Cross	5	A cross-shaped marker is displayed.
Diamond	3	A diamond-shaped marker is displayed.
None	0	No marker is displayed for the series or data point.
Square	1	A square marker is displayed.
Star10	9	A 10-point star-shaped marker is displayed.
Star4	6	A 4-point star-shaped marker is displayed.
Star5	7	A 5-point star-shaped marker is displayed.
Star6	8	A 6-point star-shaped marker is displayed.
Triangle	4	A triangular marker is displayed.

Remarks

The MarkerStyle enumeration represents a style that is used for a series or data point marker. It is used in conjunction with the MarkerStyle property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

NamedImage Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents named images, which are stored in the NamedImagesCollection.

C#

```
public class NamedImage :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement
```

Inheritance Object → ChartElement → ChartNamedElement → NamedImage

Remarks

The NamedImage class represents an image that has a unique name, and is useful when an image is dynamically created using a particular drawing routine, and then specified as the image for one of the image properties of the Chart control.

The Name property determines the unique name of a NamedImage object. The Image property determines its associated image, which can be any object derived from the .NET Framework Image class.

When one of the image properties of the Chart control is set by specifying a `string` value, the Images collection property is first checked to determine whether an image with the specified name exists. If a matching NamedImage object is found, this image is used. If a matching NamedImage object is not located, the `string` value is treated as a URL, image resource name, or an image path.

Constructors

NamedImage()	Initializes a new instance of the NamedImage class.
NamedImage(String, Image)	Initializes a new instance of the NamedImage class with the specified name of the named image and its associated image.

Properties

Image	Gets or sets the image of a NamedImage object.
Name	Gets or sets the unique name of a NamedImage object.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

NamedImage Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the NamedImage class.

Overloads

NamedImage()	Initializes a new instance of the NamedImage class.
NamedImage(String, Image)	Initializes a new instance of the NamedImage class with the specified name of the named image and its associated image.

NamedImage()

Initializes a new instance of the NamedImage class.

C#

```
public NamedImage ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

NamedImage(String, Image)

Initializes a new instance of the NamedImage class with the specified name of the named image and its associated image.

```
C#

public NamedImage (string name, System.Drawing.Image image);
```

Parameters

`name` String

The name of the named image. Use this name when setting a named image to one of the image properties.

`image` Image

The associated image of the named image.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

NamedImage.Image Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the image of a NamedImage object.

C#

```
[System.ComponentModel.Bindable(false)]
public System.Drawing.Image Image { get; set; }
```

Property Value

Image

The image component of a NamedImage object. This can be any object derived from the .NET Framework Image class, for example, a Bitmap object.

Attributes BindableAttribute

Remarks

The Image property determines which image is stored in a NamedImage object.

Use NamedImage objects for dynamically created images, which can then be assigned to a chart image property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

NamedImage.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unique name of a NamedImage object.

C#

```
[System.ComponentModel.Bindable(false)]
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the unique name of a NamedImage object stored in the NamedImagesCollection.

Attributes BindableAttribute

Remarks

The property identifies a NamedImage object that is stored in the NamedImagesCollection. This name must be unique, or an exception will be thrown when the object is added to the collection.

Use NamedImage objects for dynamically created images, which can then be assigned to a chart image property.

This Name property value is used when you set a chart image property to a NamedImage object.

When one of the image properties of the Chart control is set by specifying a `string` value, the Images collection property - of type NamedImagesCollection - is first checked to determine whether an image with the specified name exists. If a matching NamedImage object is found, this image is used. If a matching NamedImage object is

not located, the `string` value is treated as a URL, an image resource name, or an image path.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

NamedImage.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

NamedImagesCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of NamedImage objects.

C#

```
public class NamedImagesCollection :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<
System.Windows.Forms.DataVisualization.Charting.NamedImage>
```

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<NamedImage> → NamedImagesCollection

Remarks

The NamedImagesCollection class represents a collection of NamedImage objects, which are objects that have an associated image and a name that is unique to the collection.

This class is exposed as the Images collection property of the root Chart object, and is only available at run time.

An image can be any object derived from the .NET Framework Image class, such as Bitmap, which is created dynamically. The image can be stored in this collection, and then assigned at a later time to any image property in the Chart control, such as BackImage.

When one of the image properties in the Chart control is set by specifying a `string` value, the Images collection property, of type NamedImagesCollection, is first checked to see if an image with that name exists. If a matching NamedImage object is found, this image is used. If a matching NamedImage object is not located, the string expression is treated as a URL, image resource name or pathname.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Item[String]	Gets or sets the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)
NamePrefix	Gets the name prefix that is used to create unique chart element names. (Inherited from ChartNamedElementCollection<T>)

Methods

Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds the chart element by the specified name.

	(Inherited from ChartNamedElementCollection<T>)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(String)	Finds the index of the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index. (Inherited from ChartNamedElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection. (Inherited from ChartNamedElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection. (Inherited from ChartNamedElementCollection<T>)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection. (Inherited from ChartNamedElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)

SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element. (Inherited from ChartNamedElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)

IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking

	a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>,	Concatenates two sequences.

IEnumerable<TSource>)	
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>,	Returns the first element in a sequence that

Func<TSource,Boolean>)	satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each

	group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.

Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum

	Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.

OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>,	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.

IEnumerable<TSource>, IEquality Comparer<TSource>)	
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>,	Computes the sum of the sequence of Int64

Func<TSource,Int64>)	values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey>	Creates a Dictionary<TKey,TValue> from an

(IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.

Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.

Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Point3D Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the coordinates of a three-dimensional (3D) data point. This class is used when performing custom drawing with 3D charts.

C#

```
public class Point3D
```

Inheritance Object → Point3D

Remarks

Any point used in custom 3D drawing - which is accomplished using GDI+ - must be transformed from relative X, Y and Z (3D) coordinates into relative X and Y (2D) coordinates using the TransformPoints method. By calling TransformPoints, the X and Y-values of the Point3D objects are changed so that they reflect the 3-dimensional space. The new X and Y points can then be used for GDI+ method calls - the Z value is then disregarded - after being changed to absolute pixel coordinates using the GetAbsolutePoint method.

Note that the Z coordinate may be less than zero (located behind a chart area) or greater than 100 (located in front of a chart area).

To both enable 3D charts and work with their styles, use the Area3DStyle property. The Point3D class is not used for 3D styles.

Constructors

Point3D()	Initializes a new instance of the Point3D class.
Point3D(Single, Single, Single)	Initializes a new instance of the AnnotationPathPoint class with the specified X, Y and Z coordinates.

Properties

PointF	Gets or sets a PointF structure, which stores the X and Y coordinates of a 3D point.
X	Gets or sets the X coordinate of a 3D point.
Y	Gets or sets the Y coordinate of a 3D point.
Z	Gets or sets the Z coordinate of a 3D point.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Point3D Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Point3D class.

Overloads

Point3D()	Initializes a new instance of the Point3D class.
Point3D(Single, Single, Single)	Initializes a new instance of the AnnotationPathPoint class with the specified X, Y and Z coordinates.

Point3D()

Initializes a new instance of the Point3D class.

C#

```
public Point3D ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Point3D(Single, Single, Single)

Initializes a new instance of the AnnotationPathPoint class with the specified X, Y and Z coordinates.

```
C#

public Point3D (float x, float y, float z);
```

Parameters

x Single

The X coordinate of 3D point.

y Single

The Y coordinate of 3D point.

z Single

The Z coordinate, which represents the depth of 3D point.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Point3D.PointF Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a PointF structure, which stores the X and Y coordinates of a 3D point.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.PointF PointF { get; set; }
```

Property Value

PointF

A PointF structure which stores the X and Y coordinates of a 3D point.

Attributes BindableAttribute

Remarks

Any point used in custom 3D drawing - which is accomplished using GDI+ - must be transformed from X, Y and Z (3D) coordinates into X and Y (2D) coordinates using the TransformPoints method. This method takes an array of Point3D objects as its only parameter. By calling TransformPoints, the X and Y property values are changed to reflect the 3-dimensional space.

These new X and Y coordinates are then converted to absolute coordinates, by using a method such as the GetAbsolutePoint method. After converting the coordinates, the PointF property of each Point3D object can be used for any GDI+ method that take a PointF object as a parameter.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Point3D.X Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the X coordinate of a 3D point.

C#

```
[System.ComponentModel.Bindable(true)]
public float X { get; set; }
```

Property Value

Single

A `float` that represents the X coordinate of a 3D point.

Attributes BindableAttribute

Remarks

Any point used in custom 3D drawing - which is accomplished using GDI+ - must be transformed from X, Y and Z (3D) coordinates into X and Y (2D) coordinates using the TransformPoints method. This method takes an array of Point3D objects as its only parameter. By calling the TransformPoints method, the X and Y property values are changed to reflect the 3-dimensional space.

These new X and Y coordinates are then converted to absolute coordinates using the GetAbsolutePoint method. They are then used for GDI+ method calls.

A relative X coordinate can be obtained from an axis value by calling the ValueToPosition method.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Point3D.Y Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Y coordinate of a 3D point.

C#

```
[System.ComponentModel.Bindable(true)]
public float Y { get; set; }
```

Property Value

Single

A `float` value that represents the Y coordinate of a 3D point.

Attributes BindableAttribute

Remarks

Any point used in custom 3D drawing - which is accomplished using GDI+ - must be transformed from X, Y and Z (3D) coordinates into X and Y (2D) coordinates using the TransformPoints method. This method takes an array of Point3D objects as its only parameter. By calling the TransformPoints method, the X and Y property values are changed to reflect the 3-dimensional space.

These new X and Y coordinates are then converted to absolute coordinates by using the GetAbsolutePoint method. They are then used for GDI+ method calls.

A relative Y coordinate can be obtained from an axis value using the ValueToPosition method.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Point3D.Z Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Z coordinate of a 3D point.

C#

```
[System.ComponentModel.Bindable(true)]
public float Z { get; set; }
```

Property Value

Single

The Z coordinate of a 3D point, which is measured as a percentage of the relevant chart area's depth.

Attributes BindableAttribute

Remarks

This Z property is always expressed as a percentage of the relevant chart area's depth. With 3D charts, Z coordinates are relative, and have a value of 0 for the back wall of the chart area and a value of 100 for the front wall of the chart area.

Note This Z property may be less than zero (located behind a chart area) or greater than 100 (located in front of a chart area).

Any point used in custom 3D drawing - which is accomplished using GDI+ - must be transformed from X, Y and Z (3D) coordinates into X and Y (2D) coordinates using the TransformPoints method. This method takes an array of Point3D objects as its only parameter. By calling the TransformPoints method, the X and Y property values are changed to reflect the 3-dimensional space.

These new X and Y coordinates are then converted to absolute coordinates by using the GetAbsolutePoint method. They are then used for GDI+ method calls.

Relative Z coordinates can be obtained for series by using the GetSeriesDepth and GetSeriesZPosition methods.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PointSortOrder Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Indicates whether the sort order for data points is ascending or descending.

C#

```
public enum PointSortOrder
```

Inheritance Object → ValueType → Enum → PointSortOrder

Fields

Ascending	0	Ascending sort order, that is, smallest to largest values.
Descending	1	Descending sort order, that is, largest to smallest values.

Remarks

The PointSortOrder enumeration represents the order of the sort to be performed on data point values. It is used with the `Sort` method definitions in the DataManipulator class that include a `pointSortOrder` parameter.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a polygon annotation.

C#

```
public class PolygonAnnotation :
System.Windows.Forms.DataVisualization.Charting.PolylineAnnotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → PolylineAnnotation → PolygonAnnotation

Constructors

PolygonAnnotation()	Initializes a new instance of the PolygonAnnotation class.

Properties

Alignment	This property is not applicable to the polyline annotation type. (Inherited from PolylineAnnotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)

AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point. (Inherited from Annotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the annotation type.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is

	attached to. (Inherited from Annotation)
BackColor	Gets or sets the background color of an annotation.
BackGradientStyle	Gets or sets the background gradient style of an annotation.
BackHatchStyle	Gets or sets the background hatch style of an annotation.
BackSecondaryColor	Gets or sets the secondary color of the annotation background.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
EndCap	This property is not applicable to the polygon annotation type.
Font	This property is not applicable to the polyline annotation type. (Inherited from PolylineAnnotation)
ForeColor	This property is not applicable to the polyline annotation type. (Inherited from PolylineAnnotation)
GraphicsPath	Gets or sets the path points of a polyline annotation at run time. (Inherited from PolylineAnnotation)
GraphicsPathPoints	Gets or sets the path points of the polyline annotation at design time. (Inherited from PolylineAnnotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsFreeDrawPlacement	Gets or sets a flag that determines whether an annotation should be placed using the free-draw mode. (Inherited from PolylineAnnotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	Gets or sets the line color of an annotation. (Inherited from Annotation)
LineDashStyle	Gets or sets the line style of an annotation. (Inherited from Annotation)

LineWidth	Gets or sets the line width of an annotation. (Inherited from Annotation)
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
StartCap	This property is not applicable to the polygon annotation type.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	Gets or sets the text style of an annotation. (Inherited from PolylineAnnotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)

BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources. (Inherited from PolylineAnnotation)
EndPlacement()	Ends the user placement of a polyline annotation. (Inherited from PolylineAnnotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the PolygonAnnotation class.

C#

```
public PolygonAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

For internal use, should not be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of an annotation.

C#

```
[System.ComponentModel.Browsable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value used to draw the annotation background.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background gradient style of an annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public override
System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle value used to draw the annotation background.

Attributes BrowsableAttribute

Remarks

Two colors are used to draw the gradient fill; the primary color is specified with the BackColor property and the secondary color is specified with the BackSecondaryColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background hatch style of an annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle value used to draw the annotation background.

Attributes BrowsableAttribute

Remarks

Two colors are used to draw the hatching fill; the primary color is specified with the BackColor property and the secondary color is specified with the BackSecondaryColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color of the annotation background.

C#

```
[System.ComponentModel.Browsable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value used to draw the annotation background with hatching or gradient fill.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This secondary color is used with a primary color, defined with the BackColor property, when the BackHatchStyle and BackGradientStyle properties are used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation.EndCap Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polygon annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle EndCap {
get; set; }
```

Property Value

LineAnchorCapStyle

A LineAnchorCapStyle value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the polygon annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolygonAnnotation.StartCap Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polygon annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle StartCap
{ get; set; }
```

Property Value

LineAnchorCapStyle

A LineAnchorCapStyle value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the polygon annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a polyline annotation.

C#

```
public class PolylineAnnotation :
System.Windows.Forms.DataVisualization.Charting.Annotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → PolylineAnnotation

Derived System.Windows.Forms.DataVisualization.Charting.PolygonAnnotation

Constructors

PolylineAnnotation()	Initializes a new instance of the PolylineAnnotation class.

Properties

Alignment	This property is not applicable to the polyline annotation type.
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed

	to resize an annotation using a mouse. (Inherited from Annotation)
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point. (Inherited from Annotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the annotation type.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)

AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	This property is not applicable to the polyline annotation type.
BackGradientStyle	This property is not applicable to the polyline annotation type.
BackHatchStyle	This property is not applicable to the polyline annotation type.
BackSecondaryColor	This property is not applicable to the polyline annotation type.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
EndCap	Gets or sets the style of an anchor cap at the end of the annotation line.
Font	This property is not applicable to the polyline annotation type.
ForeColor	This property is not applicable to the polyline annotation type.
GraphicsPath	Gets or sets the path points of a polyline annotation at run time.
GraphicsPathPoints	Gets or sets the path points of the polyline annotation at design time.
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsFreeDrawPlacement	Gets or sets a flag that determines whether an annotation should be placed using the free-draw mode.
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	Gets or sets the line color of an annotation. (Inherited from Annotation)
LineDashStyle	Gets or sets the line style of an annotation. (Inherited from Annotation)
LineWidth	Gets or sets the line width of an annotation. (Inherited from Annotation)

Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
StartCap	Gets or sets a style for the anchor cap at the start of the annotation line.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	Gets or sets the text style of an annotation.
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
EndPlacement()	Ends the user placement of a polyline annotation.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the PolylineAnnotation class.

C#

```
public PolylineAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polyline annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.ContentAlignment Alignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment value.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

For internal use, should not be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polyline annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polyline annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the polyline annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polyline annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the polyline annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polyline annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the polyline annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.EndCap Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of an anchor cap at the end of the annotation line.

C#

```
public virtual
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle EndCap {
get; set; }
```

Property Value

LineAnchorCapStyle

A LineAnchorCapStyle value that represents the style used for an anchor cap located at the end of an annotation line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polyline annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the polyline annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the polyline annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the polyline annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.GraphicsPath Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the path points of a polyline annotation at run time.

C#

```
[System.ComponentModel.Browsable(false)]
public virtual System.Drawing.Drawing2D.GraphicsPath GraphicsPath { get;
set; }
```

Property Value

GraphicsPath

A GraphicsPath object with the polyline shape.

Attributes BrowsableAttribute

Remarks

A polyline must use coordinates relative to an annotation object, where (0,0) denotes the top-left coordinates and (100,100) denotes the bottom-right coordinates of the annotation.

This property is not accessible at design time, when you should use the GraphicsPathPoints property instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.GraphicsPathPoints Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the path points of the polyline annotation at design time.

C#

```
public
System.Windows.Forms.DataVisualization.Charting.AnnotationPathPointCollectio
n GraphicsPathPoints { get; }
```

Property Value

AnnotationPathPointCollection

An AnnotationPathPointCollection object with the polyline shape.

Remarks

A polyline must use coordinates relative to an annotation object, where (0,0) denotes the top-left coordinates and (100,100) denotes the bottom-right coordinates of the annotation.

This property is not accessible at run time, when you should use the GraphicsPath property instead.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.IsFreeDrawPlacement Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that determines whether an annotation should be placed using the free-draw mode.

C#

```
public virtual bool IsFreeDrawPlacement { get; set; }
```

Property Value

Boolean

`true` if an annotation should be placed using free-draw mode; otherwise, `false`. The default value is `false`.

Remarks

Two different placement modes are supported when the BeginPlacement method is called. Set the IsFreeDrawPlacement property to `true` to switch from the default mode to free-draw mode, which allows the user to free draw while moving the mouse cursor.

To allow the user to free draw the annotation, you must also set the AllowPathEditing property to `true`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.StartCap Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a style for the anchor cap at the start of the annotation line.

C#

```
public virtual
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle StartCap
{ get; set; }
```

Property Value

LineAnchorCapStyle

A LineAnchorCapStyle value that represents the style for an anchor cap located at the start of an annotation line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.TextStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text style of an annotation.

C#

```
[System.ComponentModel.Browsable(false)]
public override System.Windows.Forms.DataVisualization.Charting.TextStyle
TextStyle { get; set; }
```

Property Value

TextStyle

A TextStyle value used to draw the text of an annotation.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases unmanaged and, optionally, managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PolylineAnnotation.EndPlacement Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Ends the user placement of a polyline annotation.

C#

```
public override void EndPlacement ();
```

Remarks

This method ends an annotation placement operation previously started by a BeginPlacement method call.

You are not required to call this method, because placement will automatically end when an end user enters all required points. However, it is useful when an annotation placement operation needs to be aborted for some reason.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PrintingManager Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the print manager for the chart.

C#

```
public class PrintingManager : IDisposable
```

Inheritance Object → PrintingManager

Implements IDisposable

Constructors

PrintingManager(IServiceContainer)	Initializes a new instance of the PrintingManager class.

Properties

PrintDocument	Gets or sets a print document for the chart.

Methods

Dispose()	Frees, releases, or resets managed resources.
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)

GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
PageSetup()	Displays a Page Setup dialog.
Print(Boolean)	Prints the chart.
PrintPaint(Graphics, Rectangle)	Draws the chart on the printer graphics.
PrintPreview()	Provides a preview of the chart.
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PrintingManager(IServiceContainer) Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the PrintingManager class.

C#

```
public PrintingManager (System.ComponentModel.Design.IServiceContainer
container);
```

Parameters

`container` IServiceContainer

An IServiceContainer reference.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PrintingManager.PrintDocument Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a print document for the chart.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public System.Drawing.Printing.PrintDocument PrintDocument { get; set; }
```

Property Value

PrintDocument

A PrintDocument object for the chart.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PrintingManager.Dispose Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Frees, releases, or resets managed resources.

Overloads

Dispose()	Frees, releases, or resets managed resources.
Dispose(Boolean)	Releases unmanaged and, optionally, managed resources.

Dispose()

Frees, releases, or resets managed resources.

C#

```
public void Dispose ();
```

Implements

Dispose()

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Dispose(Boolean)

Releases unmanaged and, optionally, managed resources.

```
C#

protected virtual void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both unmanaged and managed resources; `false` to release only unmanaged resources.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PrintingManager.PageSetup Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Displays a Page Setup dialog.

C#

```
public void PageSetup ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PrintingManager.Print(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Prints the chart.

C#

```
public void Print (bool showPrintDialog);
```

Parameters

showPrintDialog Boolean

`true` to indicate that a print dialog should be shown; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PrintingManager.PrintPaint(Graphics, Rectangle) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Draws the chart on the printer graphics.

C#

```
public void PrintPaint (System.Drawing.Graphics graphics,
System.Drawing.Rectangle position);
```

Parameters

graphics Graphics

The printer Graphics object.

position Rectangle

The position to draw on the graphics.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

PrintingManager.PrintPreview Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides a preview of the chart.

C#

```
public void PrintPreview ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a rectangle annotation.

C#

```
public class RectangleAnnotation :
System.Windows.Forms.DataVisualization.Charting.TextAnnotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → TextAnnotation → RectangleAnnotation

Derived System.Windows.Forms.DataVisualization.Charting.Border3DAnnotation
System.Windows.Forms.DataVisualization.Charting.EllipseAnnotation

Remarks

A rectangle annotation can display text inside the rectangle, in addition to being displayed as a rectangle only. So you can annotate using a rectangle or a rectangle with text inside it.

Constructors

RectangleAnnotation()	Initializes a new instance of the AnnotationPathPoint class.

Properties

Alignment	Gets or sets the content alignment of an annotation. (Inherited from Annotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)

AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point. (Inherited from Annotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the annotation type.

AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	Gets or sets the background color of the annotation.
BackGradientStyle	Gets or sets the gradient style of the annotation.
BackHatchStyle	Gets or sets the hatch style of an annotation background.
BackSecondaryColor	Gets or sets the secondary color for an annotation background.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
Font	Gets or sets the font for the annotation text. (Inherited from TextAnnotation)
ForeColor	Gets or sets the text color of an annotation. (Inherited from Annotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsMultiline	Gets or sets a flag that indicates whether the annotation text is multiline. (Inherited from TextAnnotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	Gets or sets the line color of an annotation.
LineDashStyle	Gets or sets the line style of an annotation.

LineWidth	Gets or sets the line width, in pixels, of an annotation.
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the text of the annotation. (Inherited from TextAnnotation)
TextStyle	Gets or sets the text style of an annotation. (Inherited from Annotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)

BeginTextEditing()	Begins editing of the annotation text by an end-user. (Inherited from TextAnnotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends user placement of an annotation. (Inherited from TextAnnotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
StopTextEditing()	Stops end-user editing of the annotation text. (Inherited from TextAnnotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the AnnotationPathPoint class.

C#

```
public RectangleAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

For internal use, should not be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the annotation.

C#

```
[System.ComponentModel.Browsable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value used to draw the annotation background.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the gradient style of the annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public override
System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle value used to draw an annotation background.

Attributes BrowsableAttribute

Remarks

Two colors are used to draw the gradient, a primary color that is specified the BackColor and a secondary color that is specified with the BackSecondaryColor.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatch style of an annotation background.

C#

```
[System.ComponentModel.Browsable(true)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle value used to draw the background.

Attributes BrowsableAttribute

Remarks

Two colors are used to draw the hatching, a primary color that is specified the BackColor property, and a secondary color that is specified with the BackColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color for an annotation background.

C#

```
[System.ComponentModel.Browsable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value used to draw the annotation background with hatching or gradient fill.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This color is used with BackColor when the BackHatchStyle or BackGradientStyle property is used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line color of an annotation.

C#

```
[System.ComponentModel.Browsable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color value used to draw the annotation line.

Attributes BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line style of an annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartDashStyle LineDashStyle
{ get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle value used to draw the line style of an annotation.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

RectangleAnnotation.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the line width, in pixels, of an annotation.

C#

```
[System.ComponentModel.Browsable(true)]
public override int LineWidth { get; set; }
```

Property Value

Int32

An `integer` value that represents the line width, in pixels.

Attributes BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarButtonStyles Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the type of button that will be displayed on the scrollbar.

This enumeration supports a bitwise combination of its member values.

C#

```
[System.Flags]
public enum ScrollBarButtonStyles
```

Inheritance Object → ValueType → Enum → ScrollBarButtonStyles

Attributes FlagsAttribute

Fields

All	3	All buttons are displayed.
None	0	No buttons are displayed.
ResetZoom	2	The reset and zoom (thumb) buttons are displayed.
SmallScroll	1	The up/down and left/right buttons only are displayed.

Remarks

The ScrollBarButtonStyles enumeration is used with the ButtonStyle property to specify the buttons to be displayed for a scrollbar.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarButtonType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the button clicked by a user in the AxisScrollBarClicked event.

C#

```
public enum ScrollBarButtonType
```

Inheritance Object → ValueType → Enum → ScrollBarButtonType

Fields

LargeDecrement	3	The scroll view was decreased by a large decrement, by clicking the area beneath a scrolling thumb for a vertical scrollbar, or to the left of a scrolling thumb for a horizontal scrollbar.
LargeIncrement	4	The scroll view was increased by a large increment, by clicking the area above a scrolling thumb for a vertical scrollbar, or to the right of a scrolling thumb for a horizontal scrollbar.
SmallDecrement	1	A down or left arrow button was clicked to decrease the scroll view by a small decrement.
SmallIncrement	2	An up or right arrow button was clicked to increase the scroll view by a small increment.
ThumbTracker	0	The tracker button (also known as the scrollbar thumb) was clicked.
ZoomReset	5	The zoom or reset button was clicked.

Remarks

The ScrollBarButtonType enumeration is used with the ButtonType property, and allows the developer to determine what type of scrollbar button the end-user clicked on.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarEventArgs Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Used as a parameter object in the AxisScrollBarClicked of the root Chart object.

```
C#
public class ScrollBarEventArgs : EventArgs
```

Inheritance Object → EventArgs → ScrollBarEventArgs

Remarks

The ScrollBarEventArgs class is exposed as the `e` parameter in the AxisScrollBarClicked event.

The ScrollBarEventArgs class contains the following properties:

- The ChartArea property, which is used to get the ChartArea object that the event was raised for. The Name property gets the name of the chart area; other chart area properties can also be set.
- The Axis property, which is used to get the Axis object that the scrollbar belongs to, using the AxisName property. Other Axis properties can also be set.
- The ButtonType property, which is used to get the scrollbar button that the end-user clicked.
- The IsHandled property, which determines if the Chart control cancels default behavior of the scrollbar button after the code in the AxisScrollBarClicked event completes execution.
- The MousePositionX property, which represents the X position of the mouse cursor at the time the end-user clicked the scrollbar element.

- The MousePositionY property, which represents the Y-position of the mouse cursor at the time the end-user clicked the scrollbar element.

Constructors

ScrollBarEventArgs(Axis, Int32, Int32, ScrollBarButtonType)	Initializes a new instance of the ScrollBarEventArgs class with the specified axis, X and Y positions of the mouse cursor, and type of button clicked.

Properties

Axis	Gets the Axis object that a scrollbar belongs to.
ButtonType	Gets the type of scrollbar button that the end-user clicked.
ChartArea	Gets the ChartArea object that the scrollbar belongs to.
IsHandled	Gets or sets a flag that indicates whether to cancel the default behavior of specific scrollbar events.
MousePositionX	Gets the X position of the mouse cursor at the time the AxisScrollBarClicked event was raised.
MousePositionY	Gets the Y position of the mouse cursor at the time the AxisScrollBarClicked event was raised.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarEventArgs(Axis, Int32, Int32, ScrollBarButtonType) Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ScrollBarEventArgs class with the specified axis, X and Y positions of the mouse cursor, and type of button clicked.

C#

```
public ScrollBarEventArgs
(System.Windows.Forms.DataVisualization.Charting.Axis axis, int x, int y,
System.Windows.Forms.DataVisualization.Charting.ScrollBarButtonType
buttonType);
```

Parameters

`axis` Axis

The Axis that contains the scrollbar.

`x` Int32

The X position of mouse cursor.

`y` Int32

The Y position of mouse cursor.

`buttonType` ScrollBarButtonType

The type of button clicked.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarEventArgs.Axis Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Axis object that a scrollbar belongs to.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Axis Axis { get; }
```

Property Value

Axis

An Axis object that represents the axis that the scrollbar belongs to.

Remarks

This property is exposed in the AxisScrollBarClicked event of the root Chart object.

This property can be used to:

- determine the type of axis - X, X2, Y or Y2 - that the event is being raised for, using the AxisName property.
- get and set axis properties, such as LineColor.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarEventArgs.ButtonType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the type of scrollbar button that the end-user clicked.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ScrollBarButtonType
ButtonType { get; }
```

Property Value

ScrollBarButtonType

A ScrollBarButtonType enumeration value that represents the type of scrollbar button the end-user clicked.

Remarks

This property is exposed in the AxisScrollBarClicked event of the root Chart object, and can be used to determine the type of scrollbar button the end-user clicked.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarEventArgs.ChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the ChartArea object that the scrollbar belongs to.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartArea ChartArea {
get; }
```

Property Value

ChartArea

A ChartArea object.

Remarks

This property is exposed in the AxisScrollBarClicked event of the root Chart object.

The ChartArea property can be used to determine the chart area that the scrollbar belongs to by accessing the Name property. This property can also be used to get and set chart area properties.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarEventArgs.IsHandled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether to cancel the default behavior of specific scrollbar events.

```
C#
public bool IsHandled { get; set; }
```

Property Value

Boolean

`true` if the default behavior of the event should be cancelled, otherwise `false`.

Remarks

This property is exposed in the AxisScrollBarClicked event of the root Chart object.

This property can be used to cancel the default behavior on specific scrollbar events.

If you click on a scrollbar button, the default manipulation of the view will occur.

The following list summarizes this default behavior:

- If the user clicks the Reset/Zoom button, the previous view will be displayed.
- If the user clicks in the scrollbar background - that is, not on the scrolling thumb, otherwise known as the tracker - large increments and/or decrements of the scrollbar will occur, with the increment and/or decrement size being determined by the Size property.

To turn off these default behaviors, set the IsHandled property to `true` after providing custom logic in the AxisScrollBarClicked event.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarEventArgs.MousePositionX Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the X position of the mouse cursor at the time the AxisScrollBarClicked event was raised.

```
C#

public int MousePositionX { get; }
```

Property Value

Int32

An `integer` value that represents the X mouse cursor position. The position is expressed in pixels, with 0,0 being the top-left corner of the chart image.

Remarks

This property is used to get the X mouse position, in pixels, when the end-user clicks on a scrollbar element. The 0,0 coordinate represents the top-left corner of the chart.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollBarEventArgs.MousePositionY Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Y position of the mouse cursor at the time the AxisScrollBarClicked event was raised.

```
C#

public int MousePositionY { get; }
```

Property Value

Int32

An `integer` value that represents the Y mouse cursor position. The position is expressed in pixels, with 0,0 being the top-left corner of the chart image.

Remarks

This property is used to get the Y mouse position, in pixels, when the end-user clicks on a scrollbar element. The 0,0 coordinate represents the top-left corner of the chart.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ScrollType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the scrolling that will occur when the one of the `Scroll` methods in the AxisScaleView class is called.

C#

```
public enum ScrollType
```

Inheritance Object → ValueType → Enum → ScrollType

Fields

First	4	Data is scrolled to the first displayed view.
LargeDecrement	2	The data view is decreased by one large scrolling interval.
LargeIncrement	3	The data view is increased by one large scrolling interval.
Last	5	Data is scrolled to the last displayed view.
SmallDecrement	0	The data view is decreased by one small scrolling interval.
SmallIncrement	1	The data view is increased by one small scrolling interval.

Remarks

The ScrollType enumeration is used with one of the `Scroll` methods to specify the direction and amount of scrolling.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SerializationContents Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the type of chart content to be serialized.

This enumeration supports a bitwise combination of its member values.

C#

```
[System.Flags]
public enum SerializationContents
```

Inheritance Object → ValueType → Enum → SerializationContents

Attributes FlagsAttribute

Fields

All	7	Serialize all chart information. This includes all appearance properties and data point values.
Appearance	4	Serialize all appearance properties, such as color or line style, if they have non-default values.
Data	2	Serialize data values from all chart series. Does not include appearance properties.
Default	1	Serialize all non-default chart content. This includes appearance properties, series and their data points, axis minimums/maximums, and so forth.

Remarks

The SerializationContents enumeration represents the chart content type to be serialized. This enumeration is used with the Content property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SerializationFormat Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the formatting type to use when the chart is serialized.

C#

```
public enum SerializationFormat
```

Inheritance Object → ValueType → Enum → SerializationFormat

Fields

Binary	1	Chart data will be serialized in a binary format.
Xml	0	Chart data will be serialized using well-formed XML.

Remarks

The SerializationFormat enumeration is used in conjunction with the Format property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the data points and series attributes to store.

C#

```
public class Series :
System.Windows.Forms.DataVisualization.Charting.DataPointCustomProperties
```

Inheritance Object → ChartElement → ChartNamedElement → DataPointCustomProperties → Series

Remarks

A Series object represents a data series and is stored in the SeriesCollection class.

A Series stores DataPoint objects using the DataPointCollection class. It also stores attributes of the data series. For example, a data series has a color attribute, background image attribute, background gradient attribute, and so forth.

The DataPoint objects that belong to a series also have their own attributes. Normally, if the same attribute is set for a data point and the series to which it belongs, the data point attribute has precedence.

A series is not necessarily plotted on a chart; it can store data point values that are used to perform a calculation from which another series is created and then plotted. For example, Series A, which is plotted, may be the result of adding Series B and Series C data points together, though neither of these series is plotted. In order to draw a Series object, the ChartArea property must be set to the name of the ChartArea object used to plot the series.

When a series is created, it is automatically assigned to the ChartArea object named "Default", or to the first available chart area in the chart area collection if the "Default" object does not exist.

To disable the display of a series, set its ChartArea property to a zero-length string.

The ChartType property determines the type of chart that will be used to plot this series, if it is drawn.

Constructors

Series()	Initializes a new instance of the Series class.
Series(String)	Initializes a new instance of the Series class with the specified series name.
Series(String, Int32)	Initializes a new instance of the Series class with the specified name and maximum number of Y-values.

Properties

AxisLabel	Gets or sets the axis label text for the series.
BackGradientStyle	Gets or sets the background gradient style. (Inherited from DataPointCustomProperties)
BackHatchStyle	Gets or sets the background hatching style. (Inherited from DataPointCustomProperties)
BackImage	Gets or sets the background image of the data point. (Inherited from DataPointCustomProperties)
BackImageAlignment	Gets or sets the alignment of the background image, which is used with the Unscaled drawing mode. (Inherited from DataPointCustomProperties)
BackImageTransparentColor	Gets or sets a color that will be replaced with a transparent color when the background image is drawn. (Inherited from DataPointCustomProperties)
BackImageWrapMode	Gets or sets the drawing mode of the background image. (Inherited from DataPointCustomProperties)
BackSecondaryColor	Gets or sets the secondary background color. (Inherited from DataPointCustomProperties)
BorderColor	Gets or sets the border color of the data point. (Inherited from DataPointCustomProperties)
BorderDashStyle	Gets or sets the border style of the data point. (Inherited from DataPointCustomProperties)

BorderWidth	Gets or sets the border width of the data point. (Inherited from DataPointCustomProperties)
ChartArea	Gets or sets the name of the ChartArea object used to plot the data series, if any.
ChartType	Gets or sets the chart type of a series.
ChartTypeName	Gets or sets the name of the chart type used to draw the series.
Color	Gets or sets the color of the data point. (Inherited from DataPointCustomProperties)
CustomProperties	Gets or sets the custom properties of the data point. Custom properties can be specified in the following format: AttrName1=Value1, AttrName2=Value2 ... (Inherited from DataPointCustomProperties)
CustomPropertiesExtended	Gets or sets the extended custom properties of the data point. Extended custom properties can be specified in the following format: AttrName1=Value1, AttrName2=Value2... (Inherited from DataPointCustomProperties)
EmptyPointStyle	Gets or sets the drawing style of points marked as empty.
Enabled	Gets or sets a flag that indicates whether the series will be visible on the rendered chart.
Font	Gets or sets the font of the data point. (Inherited from DataPointCustomProperties)
IsValueShownAsLabel	Gets or sets a flag that indicates whether to show the value of the data point on the label. (Inherited from DataPointCustomProperties)
IsVisibleInLegend	Gets or sets a flag that indicates whether the item is shown in the legend. (Inherited from DataPointCustomProperties)
IsXValueIndexed	Gets or sets a flag that indicates whether data point indices will be used for the X-values.
Item[Int32]	Gets the DataPointCustomProperties object by its position in the index. (Inherited from DataPointCustomProperties)
Item[String]	Gets or sets the DataPointCustomProperties object by name. (Inherited from DataPointCustomProperties)
Label	Gets or sets the text of the data point label. (Inherited from DataPointCustomProperties)
LabelAngle	Gets or sets the angle of the data point label.

	(Inherited from DataPointCustomProperties)
LabelBackColor	Gets or sets the background color of the data point label. (Inherited from DataPointCustomProperties)
LabelBorderColor	Gets or sets the border color of the data point label. (Inherited from DataPointCustomProperties)
LabelBorderDashStyle	Gets or sets the border style of the label. (Inherited from DataPointCustomProperties)
LabelBorderWidth	Gets or sets the border width of the label. (Inherited from DataPointCustomProperties)
LabelForeColor	Gets or sets the text color of the label. (Inherited from DataPointCustomProperties)
LabelFormat	Gets or sets the format of the data point label. (Inherited from DataPointCustomProperties)
LabelToolTip	Gets or sets the tooltip for the data point label. (Inherited from DataPointCustomProperties)
Legend	Gets or sets the name of the series associated with the Legend object.
LegendText	Gets or sets the text of the item in the legend. (Inherited from DataPointCustomProperties)
LegendToolTip	Gets or sets the tooltip of the item in the legend. (Inherited from DataPointCustomProperties)
MarkerBorderColor	Gets or sets the border color of the marker. (Inherited from DataPointCustomProperties)
MarkerBorderWidth	Gets or sets the border width of the marker. (Inherited from DataPointCustomProperties)
MarkerColor	Gets or sets the marker color. (Inherited from DataPointCustomProperties)
MarkerImage	Gets or sets the marker image. (Inherited from DataPointCustomProperties)
MarkerImageTransparentColor	Gets or sets the color that will be replaced with a transparent color when the marker image is drawn. (Inherited from DataPointCustomProperties)
MarkerSize	Gets or sets the size of the marker. (Inherited from DataPointCustomProperties)
MarkerStep	Gets or sets a value that determines how often to display data point markers.

MarkerStyle	Gets or sets the marker style. (Inherited from DataPointCustomProperties)
Name	Gets or sets the unique name of a Series object.
Palette	Gets or sets the color palette of a Series object.
Points	Gets a collection of data points.
ShadowColor	Gets or sets the shadow color of a series.
ShadowOffset	Gets or sets the shadow offset, in pixels, of a series.
SmartLabelStyle	Gets or sets the style of the smart labels.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
ToolTip	Gets or sets the tooltip. (Inherited from DataPointCustomProperties)
XAxisType	Gets or sets the X-axis type of the series.
XValueMember	Gets or sets the member of the chart data source used to data bind to the X-value of the series.
XValueType	Gets or sets the value types plotted along the X-axis.
YAxisType	Gets or sets the Y-axis type of a series.
YValueMembers	Gets or sets member columns of the chart data source used to bind data to the Y-values of the series.
YValuesPerPoint	Gets or sets the maximum number of Y-values that DataPoint objects in a Series are allowed to have.
YValueType	Gets or sets the type of Y-value data in the data points stored by a series.

Methods

DeleteCustomProperty(String)	Deletes the data point custom property with the specified name. (Inherited from DataPointCustomProperties)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the Series and optionally releases the managed resources.

Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetCustomProperty(String)	Returns the data point custom property with the specified name. (Inherited from DataPointCustomProperties)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IsCustomPropertySet(String)	Determines whether a custom property with the specified name was set. (Inherited from DataPointCustomProperties)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResetIsValueShownAsLabel()	Resets a custom property to its default value. (Inherited from DataPointCustomProperties)
ResetIsVisibleInLegend()	Resets a custom property to its default value. (Inherited from DataPointCustomProperties)
SetCustomProperty(String, String)	Sets a custom property of the data point. (Inherited from DataPointCustomProperties)
SetDefault(Boolean)	Sets the default properties of the data point. Indicates whether the previous property values must be cleared. (Inherited from DataPointCustomProperties)
Sort(IComparer<DataPoint>)	Sorts the points in the series using IComparer interface.
Sort(PointSortOrder)	Sorts the data points in the series, given the specified sort order.
Sort(PointSortOrder, String)	Sorts the points in the series, given the specified sort order and value to use for the sort.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Series class.

Overloads

Series()	Initializes a new instance of the Series class.
Series(String)	Initializes a new instance of the Series class with the specified series name.
Series(String, Int32)	Initializes a new instance of the Series class with the specified name and maximum number of Y-values.

Series()

Initializes a new instance of the Series class.

C#

```
public Series ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series(String)

Initializes a new instance of the Series class with the specified series name.

```
C#

public Series (string name);
```

Parameters

name String

The name of the Series object that will be created. This must be a unique name; otherwise, an exception will be thrown.

Remarks

This constructor can be used to create a new instance of the Series class.

> ⓘ **Note**
>
> This series is only capable of storing **DataPoint** objects that have one Y-value. If a data point that has multiple Y-values is added to this series, an exception will be thrown.

To create a series at run time, use the Add method.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series(String, Int32)

Initializes a new instance of the Series class with the specified name and maximum number of Y-values.

```
C#

public Series (string name, int yValues);
```

Parameters

`name` String

The name of the Series object that will be created.

`yValues` Int32

The maximum number of Y-values allowed for the DataPoint objects that belong to this series.

Remarks

Use this constructor to create and initialize a new instance of the Series class.

> ⓘ **Important**
>
> This series will only be able to store **DataPoint** objects that do not have more Y-values than the maximum set by the `yValues` parameter. If you add a data point that has more Y-values than the set maximum, an exception will be thrown.

To create a Series object at runtime, the `Add` and `Insert` methods of the Series collection property can also be used.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.AxisLabel Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the axis label text for the series.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AxisLabel { get; set; }
```

Property Value

String

The axis label text for the series. The default value is a zero-length string.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.ChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the ChartArea object used to plot the data series, if any.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.SeriesAreaNameConverter))]
public string ChartArea { get; set; }
```

Property Value

String

A ChartArea object, which is used to plot the series.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

When a series is added at both design time and run time, it is automatically assigned to the "Default" ChartArea object, or the first available chart area in the ChartAreaCollection if "Default" does not exist.

Set this property to a zero-length string if you do not want the series to be plotted.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.ChartType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the chart type of a series.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.SeriesChartType
ChartType { get; set; }
```

Property Value

SeriesChartType

A SeriesChartType enumeration value that indicates the chart type that will be used to represent the series. The default value is Column.

Attributes BindableAttribute

Remarks

This property determines the chart type of the Series object; the value is case-sensitive. See the SeriesChartType enumeration for a list of all possible chart types for a series.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.ChartTypeName Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the chart type used to draw the series.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.ChartTypeConverter))]
public string ChartTypeName { get; set; }
```

Property Value

String

The name of the chart type.

Attributes BindableAttribute, BrowsableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.EmptyPointStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the drawing style of points marked as empty.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.DataPointCustomProperties
EmptyPointStyle { get; set; }
```

Property Value

DataPointCustomProperties

The drawing style of points marked as empty.

Attributes BindableAttribute

Remarks

The EmptyPointStyle property determines the drawing style of points marked as empty, for example, in the border color, in the marker images, background color or images, and so forth.

Certain properties are only applicable for specific chart types. For example, the BackImage property does not apply to line charts but does apply to an area chart.

To display an image that indicates an empty point, use the MarkerImage and MarkerStyle properties, and not the BackImage property.

The AxisLabel property determines the axis label for empty points. Note that for stacked chart types, the axis label is determined by the AxisLabel property of the first plotted series that has this property set.

Data points can be manually marked as an empty by setting the IsEmpty property to `true`. Alternatively, one of the `InsertEmptyPoints` methods in the DataManipulator class can be used to check for missing data and insert empty points if a data point is missing.

> **Note**
>
> In the case of line-type charts, the line color for lines that connect to an empty point is determined by the **Color** property setting.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the series will be visible on the rendered chart.

C#

```
[System.ComponentModel.Bindable(true)]
public bool Enabled { get; set; }
```

Property Value

Boolean

`true` if the series will be visible on the rendered chart; otherwise, `false`. The default value is `true`.

Attributes BindableAttribute

Remarks

Gets or sets a value that determines if the series will be visible on the rendered ChartArea.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.IsXValueIndexed Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether data point indices will be used for the X-values.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsXValueIndexed { get; set; }
```

Property Value

Boolean

`true` if the indices of data points that belong to the series will be used for X-values; otherwise, `false`. The default value is `false`.

Attributes BindableAttribute

Remarks

All data points in a series use sequential indices, and if this property is `true` then data points will be plotted sequentially, regardless of their associated X-values. This means that there will be no "missing" data points.

For example, assume there are three (3) data points in a series having X-values of 1, 2 and 4. If this property was `false`, there would be a data point missing at the X-axis location labeled "3". However, if you set this property to `true`, the three data points will be plotted at points 1, 2, and 4, sequentially, and there will be no missing data point. The X-axis location labeled "3" will not appear on the chart.

This is useful when you do not want to have missing data points for intervals that you know you will not have data for, such as weekends.

Important

If you are displaying multiple series and at least one series uses indexed X-values, then all series must be aligned - that is, have the same number of data points - and the corresponding points must have the same X-values.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.Legend Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the series associated with the Legend object.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.SeriesLegendNameConverter))]
public string Legend { get; set; }
```

Property Value

String

The name of a Legend object.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

When a series is added at both design time and run time, it is automatically assigned to the default Legend object.

If you do not want the series shown in a legend, set this property to a zero-length string, or set the IsVisibleInLegend property to `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.MarkerStep Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that determines how often to display data point markers.

C#

```
[System.ComponentModel.Bindable(true)]
public int MarkerStep { get; set; }
```

Property Value

Int32

An `integer` value that determines how often to display data point markers. The default value is one (1).

Attributes BindableAttribute

Remarks

The MarkerStep property specifies the frequency with which markers are displayed. For example, if you set this property to two (2), every other data point will use a marker.

This property is only used if markers are enabled using the MarkerStyle property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unique name of a Series object.

C#

```
[System.ComponentModel.Bindable(true)]
public override string Name { get; set; }
```

Property Value

String

The unique name of the data series.

Attributes BindableAttribute

Remarks

The name of a data series must be unique; otherwise, an exception will be thrown.

This property can be used as the indexer when you retrieve a Series object from the SeriesCollection object.

If this property is set to a zero-length string, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.Palette Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color palette of a Series object.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartColorPalette
Palette { get; set; }
```

Property Value

ChartColorPalette

An ChartColorPalette enumeration value that determines the palette for the data series.

Attributes BindableAttribute

Remarks

A palette is used to automatically assign colors to data elements if the data points themselves do not have colors assigned to them and if the data Series object that the data points belong to also does not have an assigned color.

If no series or data point colors are explicitly specified, the colors used depend on the chart type and the Chart control palette or Series palette, if one is specified. See the table below for further details:

Chart Type	Series Palette Used?	Colors Displayed by Data Points
Any shape chart (for example, Pie and Doughnut charts)	No	First available colors used from Chart control palette. One color per data point.

Chart Type	Series Palette Used?	Colors Displayed by Data Points
All charts except shape charts.	No	First available colors from Chart control palette. One color per data series.
Any shape chart (for example, Pie and Doughnut charts)	Yes	First available colors used from Series palette. One color per data point.
All charts except shape charts.	Yes	First available colors used from Series palette. One color per data series.

This Palette property has precedence over the Palette property of the root Chart object, with respect to the display of data.

NOTE: If the `Palette` property for both the Chart and Series objects is set to None, and no color is assigned to the relevant data series/data points, the default palette "BrightPastel" or PaletteCustomColors is used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.Points Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a collection of data points.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DataPointCollection
Points { get; }
```

Property Value

DataPointCollection

A collection of data points.

Attributes BindableAttribute

Remarks

The Points collection property allows you to add and remove data points to a series. This property also provides methods and properties that can be used to manipulate this collection, such as finding data points, performing data-binding, and so forth.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.ShadowColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow color of a series.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.Color ShadowColor { get; set; }
```

Property Value

Color

The shadow color of a series. The default color is Black.

Attributes BindableAttribute

Remarks

Gets or sets the shadow color of a series, which can be to any valid ARGB (alpha, red, green, blue) value.

The alpha value can be used to achieve a realistic shadow effect; try using a color value of "128,0,0,0" for shadows.

To implement a shadow, set the ShadowOffset property to a value other than zero (0).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.ShadowOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow offset, in pixels, of a series.

C#

```
[System.ComponentModel.Bindable(true)]
public int ShadowOffset { get; set; }
```

Property Value

Int32

The shadow offset of the series, in pixels.

Attributes BindableAttribute

Remarks

To implement shadowing, use a value other than zero (0). A positive value will result in a shadow that shifts down and to the right. A negative value will result in a shadow that shifts up and to the left.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.SmartLabelStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the smart labels.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(true)]
public System.Windows.Forms.DataVisualization.Charting.SmartLabelStyle
SmartLabelStyle { get; set; }
```

Property Value

SmartLabelStyle

The style of the smart labels.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.XAxisType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the X-axis type of the series.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.AxisType XAxisType {
get; set; }
```

Property Value

AxisType

A AxisType enumeration value that determines if the series uses the primary or secondary X-axis.

Attributes BindableAttribute

Remarks

For all chart types except horizontally oriented charts such as bar and stacked bar charts, and shape charts such as pie, doughnut, funnel and pyramid charts, the primary X-axis is always the lower axis and the secondary axis is the upper axis. For bar and stacked bar charts, the primary X-axis is the left axis, and the secondary X-axis is the right axis.

When data is plotted using the secondary X-axis, you must use the AxisX2 property to set axis properties, and not the AxisX property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.XValueMember Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the member of the chart data source used to data bind to the X-value of the series.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.SeriesDataSourceMemberConverter))]
public string XValueMember { get; set; }
```

Property Value

String

The member of the chart data source used to data bind to the X-value of the series.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property accepts either the name of the member column or the ordinal value of the member column.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.XValueType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the value types plotted along the X-axis.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartValueType
XValueType { get; set; }
```

Property Value

ChartValueType

A ChartValueType enumeration value that determines the type of X-value data stored by the series. The default value is Auto.

Attributes BindableAttribute

Remarks

The XValueType property determines the type of data stored by the series.

Set this property to DateTime to allow the X-value to be entered as a DateTime value, and not as a Double. Date and time values are internally stored by .NET Framework as Double values. The same holds true for the ToPosition and FromPosition properties of CustomLabel objects.

The `IntervalType` property found in axis label, tick mark, and grid classes determines the interval type.

Note

To correctly apply a **DateTime** format to axis labels that are generated from data point X-values, using the **Format** property, this property must be set to **DateTime**.

This property can also be used to determine the data type to which the X-values are bound.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.YAxisType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the Y-axis type of a series.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.AxisType YAxisType {
get; set; }
```

Property Value

AxisType

An AxisType enumeration value that determines if a series uses the primary or secondary Y-axis.

Attributes BindableAttribute

Remarks

The YAxisType property determines if a series is plotted using the primary or secondary Y-axis.

For all chart types except pie, doughnut, range bar, bar and stacked bar charts, the primary Y-axis is the left vertical axis, and the secondary axis is the right vertical axis. For bar and stacked bar charts, the primary axis is the lower horizontal axis, and the secondary Y-axis is the upper horizontal axis.

When data is plotted using the secondary Y-axis, you must use the AxisY2 property to set axis properties, and not the AxisY property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.YValueMembers Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets member columns of the chart data source used to bind data to the Y-values of the series.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.SeriesDataSourceMemberConverter))]
public string YValueMembers { get; set; }
```

Property Value

String

The member columns of the chart data source used to bind data to the Y-values of the series.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property accepts a comma separated string that contains the names or ordinal values of the columns.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.YValuesPerPoint Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the maximum number of Y-values that DataPoint objects in a Series are allowed to have.

C#

```
[System.ComponentModel.Bindable(true)]
public int YValuesPerPoint { get; set; }
```

Property Value

Int32

The maximum number of Y-values that the data points in a series are allowed to have. The default value is one (1).

Attributes BindableAttribute

Remarks

If you add a DataPoint object that has more Y-values than this property specifies, the excess Y-values will be ignored.

The number of Y-values that are required to properly display a chart depends on the chart type selected.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.YValueType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the type of Y-value data in the data points stored by a series.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.SeriesYValueTypeConverter))]
public System.Windows.Forms.DataVisualization.Charting.ChartValueType
YValueType { get; set; }
```

Property Value

ChartValueType

A ChartValueType enumeration value that determines the type of Y-value data stored by the series. The default value is Auto.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The YValueType property determines the type of data stored by the series. The value of this property is normally set to either Auto or DateTime.

The `IntervalType` property found in axis label, tick mark, and grid classes determines the interval type.

This property can also be used to determine the data type to which the Y-values are bound.

It is possible to plot date expressions as the values of data points, though this is rarely done. If you set this property to DateTime, the Y-value can then entered as a DateTime

value, and not as a Double. Date and time values are internally stored by .NET Framework as Double values.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the unmanaged resources used by the Series and optionally releases the managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Series.Sort Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs a sort on a Series. Note that the DataManipulator class provides sort methods that have greater flexibility than these sort methods.

Overloads

Sort(IComparer<DataPoint>)	Sorts the points in the series using IComparer interface.
Sort(PointSortOrder)	Sorts the data points in the series, given the specified sort order.
Sort(PointSortOrder, String)	Sorts the points in the series, given the specified sort order and value to use for the sort.

Sort(IComparer<DataPoint>)

Sorts the points in the series using IComparer interface.

C#

```
public void Sort
(System.Collections.Generic.IComparer<System.Windows.Forms.DataVisualization.Charting.DataPoint> comparer);
```

Parameters

`comparer` IComparer<DataPoint>

The IComparer interface.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Sort(PointSortOrder)

Sorts the data points in the series, given the specified sort order.

C#

```
public void Sort
(System.Windows.Forms.DataVisualization.Charting.PointSortOrder
pointSortOrder);
```

Parameters

`pointSortOrder` PointSortOrder

The sort order.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Sort(PointSortOrder, String)

Sorts the points in the series, given the specified sort order and value to use for the sort.

C#

```
public void Sort
(System.Windows.Forms.DataVisualization.Charting.PointSortOrder
pointSortOrder, string sortBy);
```

Parameters

`pointSortOrder` PointSortOrder

The sort order.

`sortBy` String

The value used for the sort (X, Y, Y2 ...).

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SeriesChartType Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a chart type for a Series.

C#

```
public enum SeriesChartType
```

Inheritance Object → ValueType → Enum → SeriesChartType

Fields

Area	13	Area chart type.
Bar	7	Bar chart type.
BoxPlot	28	Box plot chart type.
Bubble	2	Bubble chart type.
Candlestick	20	Candlestick chart type.
Column	10	Column chart type.
Doughnut	18	Doughnut chart type.
ErrorBar	27	Error bar chart type.
FastLine	6	FastLine chart type.
FastPoint	1	FastPoint chart type.
Funnel	33	Funnel chart type.
Kagi	31	Kagi chart type.
Line	3	Line chart type.
Pie	17	Pie chart type.

Point	0	Point chart type.
PointAndFigure	32	PointAndFigure chart type.
Polar	26	Polar chart type.
Pyramid	34	Pyramid chart type.
Radar	25	Radar chart type.
Range	21	Range chart type.
RangeBar	23	RangeBar chart type.
RangeColumn	24	Range column chart type.
Renko	29	Renko chart type.
Spline	4	Spline chart type.
SplineArea	14	Spline area chart type.
SplineRange	22	Spline range chart type.
StackedArea	15	Stacked area chart type.
StackedArea100	16	Hundred-percent stacked area chart type.
StackedBar	8	Stacked bar chart type.
StackedBar100	9	Hundred-percent stacked bar chart type.
StackedColumn	11	Stacked column chart type.
StackedColumn100	12	Hundred-percent stacked column chart type.
StepLine	5	StepLine chart type.
Stock	19	Stock chart type.
ThreeLineBreak	30	ThreeLineBreak chart type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SeriesCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of Series objects.

C#

```
public class SeriesCollection :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<
System.Windows.Forms.DataVisualization.Charting.Series>
```

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<Series> → SeriesCollection

Remarks

The SeriesCollection class represents a collection of Series objects, which in turn represent a grouping of related data points and attributes for the series. This collection class is implemented as the Series collection property of the root Chart object.

Series objects can be appended, inserted or removed from the collection at both design time and run time.

When a series is added to the collection, it is automatically assigned to the ChartArea object named "Default" or, if "Default" does not exist, to the first available chart area in the chart area collection.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)

Item[String]	Gets or sets the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)
NamePrefix	Gets the name prefix that is used to create unique chart element names. (Inherited from ChartNamedElementCollection<T>)

Methods

Add(String)	Adds a Series object to the end of the collection.
Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds the chart element by the specified name. (Inherited from ChartNamedElementCollection<T>)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function.

	(Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(String)	Finds the index of the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index. (Inherited from ChartNamedElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection. (Inherited from ChartNamedElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection. (Inherited from ChartNamedElementCollection<T>)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection. (Inherited from ChartNamedElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element. (Inherited from ChartNamedElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection.

	(Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a

	transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.

Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource>(IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource>(IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEqualityComparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.

FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>,	Correlates the elements of two sequences based on equality of keys and groups the

Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum

	Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.

Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.

Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.

Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by

	invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.

ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEqualityComparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.

AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement>(IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SeriesCollection.Add(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a Series object to the end of the collection.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Series Add (string
name);
```

Parameters

name String

The name of the Series to be added.

Returns

Series

The new Series object appended to the collection.

Remarks

Items are always added as the last item in the collection.

The `name` argument refers to the `Name` property of the object being added.

Note If you use this method to add Series objects to the SeriesCollection class, the associated DataPoint objects of the newly added Series object cannot have more than one Y-value per data point.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides methods and properties used to perform intelligent, rule-based data point label positioning to resolve overlapping data point label issues.

C#

```
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.NoNameExpandableObjectConverter))]
public class SmartLabelStyle
```

Inheritance Object → SmartLabelStyle

Derived System.Windows.Forms.DataVisualization.Charting.AnnotationSmartLabelStyle

Attributes TypeConverterAttribute

Remarks

When smart labels are enabled, the chart repositions overlapping data point labels, based on a set of user-defined rules.

Callout lines will be drawn to link the label with its data point. The visual attributes of the callout line can be controlled. These visual attributes include color, line style, and line width. The CalloutStyle and CalloutLineAnchorCapStyle properties control the embellishments, such as arrowheads, that appear at the end of the callout line.

The CalloutLineColor property can be set to any Color object. If you set the color of the callout line, you will also automatically set the color of the CalloutLineAnchorCapStyle and the embellishment defined by the CalloutStyle.

The CalloutLineWidth property controls the pixel width of the callout line. Use of this property will also scale the embellishments of the CalloutLineAnchorCapStyle and the CalloutStyle.

The movement of smart labels can be controlled through the use of several motion-related properties. These define the rules of movement for the direction, distance and whether the data point label is allowed to leave the plotting area. The default movement rules will normally resolve most issues. The MaxMovingDistance defines a value that specifies the maximum distance in pixels from the data point that data point labels are allowed to be moved in case of overlapping. The MinMovingDistance property defines a value that specifies the minimum distance in pixels from the data point that data point labels will be moved in case of overlapping. When a data point label needs to be repositioned, the label will be moved by at least the number of pixels specified.

The IsOverlappedHidden property indicates whether the labels are hidden when overlapping issues cannot be resolved. If the IsOverlappedHidden property is `true`, and the data point cannot be drawn within the area specified by MinMovingDistance and MaxMovingDistance in the directions allowed by MovingDirection, the data point label will not be drawn.

The MovingDirection property defines the direction(s) in which the data point label can be moved. This property accepts one LabelAlignmentStyles value or a bitwise-OR of multiple LabelAlignmentStyles values. The default is the bitwise-OR of all LabelAlignmentStyles values, to allow repositioning in all possible directions except to the center of the data point.

Smart labels are not supported for the following chart types:

- Bar charts: bar and hundred-percent-stacked bar charts.
- Shape charts: pie, doughnut, polar and radar charts.
- Range charts: range, range column and range bar charts.
- Stacked area charts: stacked area and hundred-percent-stacked area charts.

Constructors

SmartLabelStyle()	Initializes a new instance of the SmartLabelStyle class.

Properties

AllowOutsidePlotArea	Gets or sets a flag that specifies whether a SmartLabelStyle object can be drawn outside the plotting area.
CalloutBackColor	Gets or sets the background color of the label callout.

CalloutLineAnchorCapStyle	Gets or sets the anchor cap style of the label callout line.
CalloutLineColor	Gets or sets the color of the label callout line.
CalloutLineDashStyle	Gets or sets the style of the label callout line.
CalloutLineWidth	Gets or sets the width of the label callout line.
CalloutStyle	Gets or sets the callout style of the repositioned SmartLabelStyle object.
Enabled	Gets or sets a flag that indicates whether a SmartLabelStyle algorithm is enabled.
IsMarkerOverlappingAllowed	Gets or sets a flag that specifies whether the point labels are allowed to overlap a point marker.
IsOverlappedHidden	Gets or sets a flag that indicates whether overlapped labels that cannot be repositioned will be hidden.
MaxMovingDistance	Gets or sets the maximum distance, in pixels, that the overlapped SmartLabelStyle object can be moved from the marker.
MinMovingDistance	Gets or sets the minimum distance, in pixels, that the overlapped SmartLabelStyle object can be moved from the marker.
MovingDirection	Gets or sets the direction(s) in which the overlapped SmartLabelStyle object is allowed to be moved.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the SmartLabelStyle class.

C#

```
public SmartLabelStyle ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.AllowOutsidePlotArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether a SmartLabelStyle object can be drawn outside the plotting area.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual
System.Windows.Forms.DataVisualization.Charting.LabelOutsidePlotAreaStyle
AllowOutsidePlotArea { get; set; }
```

Property Value

LabelOutsidePlotAreaStyle

A LabelOutsidePlotAreaStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.CalloutBackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the label callout.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color CalloutBackColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property is only applicable when the CalloutStyle property is set to a value of Box.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.CalloutLineAnchorCapStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the anchor cap style of the label callout line.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual
System.Windows.Forms.DataVisualization.Charting.LineAnchorCapStyle
CalloutLineAnchorCapStyle { get; set; }
```

Property Value

LineAnchorCapStyle

A LineAnchorCapStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.CalloutLineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the label callout line.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public virtual System.Drawing.Color CalloutLineColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.CalloutLineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the label callout line.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual
System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
CalloutLineDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.CalloutLineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of the label callout line.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual int CalloutLineWidth { get; set; }
```

Property Value

Int32

An `integer` value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.CalloutStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the callout style of the repositioned SmartLabelStyle object.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual
System.Windows.Forms.DataVisualization.Charting.LabelCalloutStyle
CalloutStyle { get; set; }
```

Property Value

LabelCalloutStyle

A LabelCalloutStyle enumeration value.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.Enabled Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether a SmartLabelStyle algorithm is enabled.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual bool Enabled { get; set; }
```

Property Value

Boolean

`true` if an algorithm will be applied to prevent data point labels from overlapping. `False` if overlapping of data point labels will not be prevented. The default value is `true`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.IsMarkerOverlappingAllowed Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the point labels are allowed to overlap a point marker.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual bool IsMarkerOverlappingAllowed { get; set; }
```

Property Value

Boolean

`true` if data point labels are permitted to overlap point markers. `False` if data point labels will be prevented from overlapping the point markers. The default value is `false`.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.IsOverlappedHidden Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether overlapped labels that cannot be repositioned will be hidden.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual bool IsOverlappedHidden { get; set; }
```

Property Value

Boolean

`true` if labels will be hidden if they overlap. `false` if overlapped labels will not be hidden. The default value is `true`.

Attributes BindableAttribute

Remarks

If the point cannot be drawn within the area specified by the properties MinMovingDistance and MaxMovingDistance, in the directions allowed by the MovingDirection, property, and this property is `true`, the data point label will not be drawn.

> **Note**
>
> Smart labels only work when the **Angle** property is set to zero.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.MaxMovingDistance Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the maximum distance, in pixels, that the overlapped SmartLabelStyle object can be moved from the marker.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual double MaxMovingDistance { get; set; }
```

Property Value

Double

A `double` value.

Attributes BindableAttribute

Remarks

This property specifies the maximum distance in pixels from the data point that data point labels are allowed to be moved in case of overlapping.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.MinMovingDistance Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the minimum distance, in pixels, that the overlapped SmartLabelStyle object can be moved from the marker.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual double MinMovingDistance { get; set; }
```

Property Value

Double

A `double` value.

Attributes BindableAttribute

Remarks

This property specifies the minimum distance in pixels from the data point that data point labels are allowed to be moved in case of overlapping.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

SmartLabelStyle.MovingDirection Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the direction(s) in which the overlapped SmartLabelStyle object is allowed to be moved.

C#

```
[System.ComponentModel.Bindable(true)]
public virtual
System.Windows.Forms.DataVisualization.Charting.LabelAlignmentStyles
MovingDirection { get; set; }
```

Property Value

LabelAlignmentStyles

A LabelAlignmentStyles enumeration value or multiple bitwise OR'd LabelAlignmentStyles values. The default is the bitwise OR of all LabelAlignmentStyles values to allow repositioning in all directions except to the center of the data point.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StartFromZero Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Indicates whether an axis segment should start from zero when a scale break is used.

C#

```
public enum StartFromZero
```

Inheritance Object → ValueType → Enum → StartFromZero

Fields

Auto	0	The Chart control calculates whether to start the axis scale at zero or at a more appropriate rounded value.
No	2	Do not start the axis segment scale from zero.
Yes	1	Start the axis segment scale from zero.

Remarks

The StartFromZero enumeration is used to set the value of the StartFromZero property and indicates whether an axis segment should start from zero.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Calculates statistical formulas.

C#

```
public class StatisticFormula
```

Inheritance Object → StatisticFormula

Remarks

Formulas can be loosely grouped into the following categories: distributions, including inverse distributions; tests; and basic statistical formulas. Each group has common characteristics with respect to their input and return values.

Methods

Anova(Double, String)	Performs an ANOVA test to determine whether there is a statistically significant difference between the mean values of two or more groups of data.
BetaFunction(Double, Double)	Returns the beta function value for two given values.
Correlation(String, String)	Returns the correlation value for two data series.
Covariance(String, String)	Returns the covariance value for two data series.
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FDistribution(Double, Int32, Int32)	Returns the cumulative F distribution function probability.
FTest(Double, String, String)	Performs a two-sample F Test using the F distribution. This test is

	used to determine if the samples have different variances.
GammaFunction(Double)	Returns the gamma function value for the given variable.
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
InverseFDistribution(Double, Int32, Int32)	Returns the inverse of the F cumulative distribution.
InverseNormal Distribution(Double)	Returns the inverse of the standard normal cumulative distribution.
InverseTDistribution(Double, Int32)	Returns the inverse t-value of the Students t-distribution as a function of probability and degree of freedom.
Mean(String)	Returns the average of all data points stored in the specified series.
Median(String)	Returns the median of all data points in the specified series.
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NormalDistribution(Double)	Returns the probability for the standard normal cumulative distribution function.
TDistribution(Double, Int32, Boolean)	Returns the probability for the T distribution (Student's distribution).
ToString()	Returns a string that represents the current object. (Inherited from Object)
TTestEqualVariances(Double, Double, String, String)	Performs a T Test using Student's distribution (T distribution) with equal variances.
TTestPaired(Double, Double, String, String)	Performs a T Test using Student's distribution (T distribution) with paired samples. This is useful when there is a natural pairing of observations in samples.
TTestUnequal Variances(Double, Double, String, String)	Performs a T Test using Student's distribution (T distribution) with unequal variances.
Variance(String, Boolean)	Returns the variance for a series.
ZTest(Double, Double, Double, Double, String, String)	Performs a Z Test using the Normal distribution.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.Anova(Double, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs an ANOVA test to determine whether there is a statistically significant difference between the mean values of two or more groups of data.

C#

```
public System.Windows.Forms.DataVisualization.Charting.AnovaResult Anova
(double probability, string inputSeriesNames);
```

Parameters

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

`inputSeriesNames` String

A comma-delimited list of input series names.

Returns

AnovaResult

An AnovaResult object.

Remarks

An ANOVA test is used to test the difference between the means of two or more groups of data.

Two or more groups of series data must be specified, and each series must have the same number of data points; otherwise, an exception will be raised.

If a specified input series does not exist in the SeriesCollection at the time of the method call, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.BetaFunction(Double, Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the beta function value for two given values.

C#

```
public double BetaFunction (double m, double n);
```

Parameters

`m` Double

The first value for the beta function.

`n` Double

The second value for the beta function.

Returns

Double

A `double` value that represents the beta function for the two given values.

Remarks

This method returns the beta function value for two given values that are specified by `m` and `n`.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.Correlation(String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the correlation value for two data series.

C#

```
public double Correlation (string firstInputSeriesName, string
secondInputSeriesName);
```

Parameters

`firstInputSeriesName` String

The first input series name.

`secondInputSeriesName` String

The second input series name.

Returns

Double

A `double` value that represents the correlation between two data series.

Remarks

Correlation measures the relationship between two data sets that are scaled to be independent of the unit of measurement. The Correlation method returns the covariance of two data sets divided by the product of their standard deviations, and always ranges from -1 to 1.

Use correlation to determine whether two ranges of data move together; that is, whether large values of one set are associated with large values of the other (positive

correlation), whether small values of one set are associated with large values of the other (negative correlation), or whether values in both sets are unrelated (correlation near zero).

If a specified input series does not exist in the series collection at the time of the method call, an exception will be thrown. An exception will also be raised if the specified series do not have the same number of data points.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.Covariance(String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the covariance value for two data series.

C#

```
public double Covariance (string firstInputSeriesName, string
secondInputSeriesName);
```

Parameters

`firstInputSeriesName` String

The first input series name.

`secondInputSeriesName` String

The second input series name.

Returns

Double

A `double` value that represents the covariance between two data series.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.FDistribution(Double, Int32, Int32) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the cumulative F distribution function probability.

C#

```
public double FDistribution (double value, int firstDegreeOfFreedom, int
secondDegreeOfFreedom);
```

Parameters

`value` Double

The F value.

`firstDegreeOfFreedom` Int32

First degree of freedom.

`secondDegreeOfFreedom` Int32

Second degree of freedom.

Returns

Double

A `double` value that represents the cumulative F distribution function probability.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.FTest(Double, String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs a two-sample F Test using the F distribution. This test is used to determine if the samples have different variances.

C#

```
public System.Windows.Forms.DataVisualization.Charting.FTestResult FTest
(double probability, string firstInputSeriesName, string
secondInputSeriesName);
```

Parameters

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

`firstInputSeriesName` String

First input series name.

`secondInputSeriesName` String

Second input series name.

Returns

FTestResult

An FTestResult object.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.GammaFunction(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the gamma function value for the given variable.

C#

```
public double GammaFunction (double value);
```

Parameters

`value` Double

The input variable.

Returns

Double

A `double` value that represents the gamma function.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.InverseFDistribution(Double, Int32, Int32) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the inverse of the F cumulative distribution.

C#

```
public double InverseFDistribution (double probability, int
firstDegreeOfFreedom, int secondDegreeOfFreedom);
```

Parameters

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

`firstDegreeOfFreedom` Int32

First degree of freedom.

`secondDegreeOfFreedom` Int32

Second degree of freedom.

Returns

Double

A `double` value from the inverse F distribution function.

Exceptions

InvalidOperationException

Condition: Cannot continue calculation using specified parameter values because of limitation.

Remarks

This method returns the F value for the given F distribution probability.

The F distribution can be used in an F-test that compares the degree of variability in two data sets.

This method may return an exception depending on the parameter values.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.InverseNormalDistribution(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the inverse of the standard normal cumulative distribution.

C#

```
public double InverseNormalDistribution (double probability);
```

Parameters

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

Returns

Double

A `double` value from the inverse standard normal cumulative distribution function.

Remarks

This method returns the F value for the given F distribution probability.

It is assumed that the distribution has a mean of 0 (zero) and a standard deviation of one.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.InverseTDistribution(Double, Int32) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the inverse t-value of the Students t-distribution as a function of probability and degree of freedom.

C#

```
public double InverseTDistribution (double probability, int
degreeOfFreedom);
```

Parameters

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

`degreeOfFreedom` Int32

The degree of freedom.

Returns

Double

A `double` value that represents the inverse T distribution.

Remarks

This method returns the t value for the given T distribution probability.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.Mean(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the average of all data points stored in the specified series.

C#

```
public double Mean (string inputSeriesName);
```

Parameters

`inputSeriesName` String

Input series name.

Returns

Double

A `double` value that represents the average of all data points.

Remarks

Use this method to calculate the mean (that is, the average) of the points stored in a series.

If the specified input series does not exist in the SeriesCollection at the time of the method call, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.Median(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the median of all data points in the specified series.

C#

```
public double Median (string inputSeriesName);
```

Parameters

`inputSeriesName` String

Input series name.

Returns

Double

A `double` value that represents the median.

Remarks

Use this method to calculate the median of the points stored in a series.

The median is the middle value of a sample set, where half of the members are greater in size and half the members are lesser in size.

If the specified input series does not exist in the SeriesCollection at the time of the method call, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.NormalDistribution(Double) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the probability for the standard normal cumulative distribution function.

C#

```
public double NormalDistribution (double zValue);
```

Parameters

`zValue` Double

The Z value for which the probability is required.

Returns

Double

A `double` value from the standard normal cumulative distribution function.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.TDistribution(Double, Int32, Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the probability for the T distribution (Student's distribution).

```
C#

public double TDistribution (double value, int degreeOfFreedom, bool
oneTail);
```

Parameters

`value` Double

The T value.

`degreeOfFreedom` Int32

Degree of freedom.

`oneTail` Boolean

If `true`, a one-tailed distribution is used; otherwise, a two-tailed distribution is used.

Returns

Double

A `double` value that represents a T Distribution cumulative function.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.TTestEqualVariances(Double, Double, String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs a T Test using Student's distribution (T distribution) with equal variances.

C#

```
public System.Windows.Forms.DataVisualization.Charting.TTestResult
TTestEqualVariances (double hypothesizedMeanDifference, double probability,
string firstInputSeriesName, string secondInputSeriesName);
```

Parameters

`hypothesizedMeanDifference` Double

Hypothesized mean difference.

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

`firstInputSeriesName` String

First input series name.

`secondInputSeriesName` String

Second input series name.

Returns

TTestResult

A TTestResult object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.TTestPaired(Double, Double, String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs a T Test using Student's distribution (T distribution) with paired samples. This is useful when there is a natural pairing of observations in samples.

C#

```
public System.Windows.Forms.DataVisualization.Charting.TTestResult
TTestPaired (double hypothesizedMeanDifference, double probability, string
firstInputSeriesName, string secondInputSeriesName);
```

Parameters

`hypothesizedMeanDifference` Double

Hypothesized mean difference.

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

`firstInputSeriesName` String

First input series name.

`secondInputSeriesName` String

Second input series name.

Returns

TTestResult

A TTestResult object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.TTestUnequalVariances(Double, Double, String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs a T Test using Student's distribution (T distribution) with unequal variances.

C#

```
public System.Windows.Forms.DataVisualization.Charting.TTestResult
TTestUnequalVariances (double hypothesizedMeanDifference, double
probability, string firstInputSeriesName, string secondInputSeriesName);
```

Parameters

`hypothesizedMeanDifference` Double

Hypothesized mean difference.

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

`firstInputSeriesName` String

First input series name.

`secondInputSeriesName` String

Second input series name.

Returns

TTestResult

A TTestResult object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.Variance(String, Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Returns the variance for a series.

C#

```
public double Variance (string inputSeriesName, bool sampleVariance);
```

Parameters

`inputSeriesName` String

The input series name.

`sampleVariance` Boolean

`true` if the data is a sample of the population; `false` if the data is the entire population.

Returns

Double

A `double` value that represents the variance.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StatisticFormula.ZTest(Double, Double, Double, Double, String, String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Performs a Z Test using the Normal distribution.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ZTestResult ZTest
(double hypothesizedMeanDifference, double varianceFirstGroup, double
varianceSecondGroup, double probability, string firstInputSeriesName, string
secondInputSeriesName);
```

Parameters

`hypothesizedMeanDifference` Double

Hypothesized mean difference.

`varianceFirstGroup` Double

Variance of the first group.

`varianceSecondGroup` Double

Variance of the second group.

`probability` Double

The alpha value, which is the probability that the hypothesis is rejected. The allowable range is 0-1.

`firstInputSeriesName` String

First input series name.

`secondInputSeriesName` String

Second input series name.

Returns

ZTestResult

A ZTestResult object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the strip lines on a chart.

C#

```
public class StripLine :
System.Windows.Forms.DataVisualization.Charting.ChartElement
```

Inheritance Object → ChartElement → StripLine

Remarks

Strip lines, or strips, are horizontal or vertical ranges that shade the background of a chart in regular or custom intervals. You can use strip lines to:

- Improve readability for looking up individual values on the chart.
- Separate data points when reading the chart.
- Highlight dates that occur at regular intervals, for example, to identify weekend data points.
- Highlight a specific key range of data.
- Add a threshold line at a specific constant value.

A single StripLine object can either be drawn once, or repeatedly, for a given interval. This action is controlled by the Interval property. When a value of zero is assigned to the Interval property, one strip line will be drawn. When a non-zero value is assigned to the Interval property, a strip line will be drawn repeatedly at each given interval. The location where a strip line is drawn is also affected by the IntervalOffset and IntervalOffsetType properties of the strip line.

Strip lines are always associated with an Axis object. They can be added at both design time and run time.

To add a horizontal or vertical line to display a threshold, set the StripWidth property to a value of 0.0. This will result in a line being drawn. You can use the BorderColor, BorderDashStyle and BorderWidth properties for the color, width and style of the line. No chart background properties (`Back*`) are used when the StripWidth property is set to 0.0.

Use the Text property of the strip line to associate text with a strip line. The placement and orientation of this text can be controlled by the TextAlignment property.

When multiple strip lines are defined for the same axis, it is possible that the strip lines will overlap. The Z-order of StripLine objects is determined by their order of occurrence in the StripLinesCollection object. This means that the first occurrence is drawn first; the second occurrence is drawn second, and so on.

Strip lines are not supported for the following chart types: Pie, Doughnut, Funnel, Pyramid, Kagi, ThreeLineBreak, PointAndFigure, Polar and Radar.

Constructors

StripLine()	Initializes a new instance of the StripLine class.

Properties

BackColor	Gets or sets the background color of the strip line.
BackGradientStyle	Gets or sets the gradient style of the strip line.
BackHatchStyle	Gets or sets the hatching style of the strip line.
BackImage	Gets or sets the background image of the strip line.
BackImageAlignment	Gets or sets the background image alignment.
BackImageTransparentColor	Gets or sets a color which will be replaced with a transparent color when the background image is drawn.
BackImageWrapMode	Gets or sets the drawing mode of the background image of the strip line.
BackSecondaryColor	Gets or sets the secondary color of the strip line background.
BorderColor	Gets or sets the border color of a strip line.
BorderDashStyle	Gets or sets the border style of the strip line.
BorderWidth	Gets or sets the border width of the strip line.

Font	Gets or sets the font used for the strip line text.
ForeColor	Gets or sets the color of the strip line text.
Interval	Gets or sets the interval for a strip line, and determines if the strip line is drawn once or repeatedly.
IntervalOffset	Gets or sets the offset of grid lines, tick marks, strip lines and axis labels.
IntervalOffsetType	Gets or sets the interval offset type of the strip line.
IntervalType	Gets or sets the interval type of a StripLine object.
Name	Gets the name of the strip line.
StripWidth	Gets or sets the width of a strip line.
StripWidthType	Gets or sets the unit of measurement for the StripWidth property.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the text of the strip line.
TextAlignment	Gets or sets the text alignment of the strip line.
TextLineAlignment	Gets or sets the text line alignment of the strip line.
TextOrientation	Gets or sets the text orientation.
ToolTip	Gets or sets the tooltip of a strip line.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the StripLine and optionally releases the managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance.

	(Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the StripLine class.

C#

```
public StripLine ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color structure. The default color is White.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set this property to any valid ARGB (alpha, red, green, blue) value.

To set a gradient for the strip line, set the BackColor and BackSecondaryColor properties.

If you set this color to "transparent" - in other words, set the alpha value to 0 - no color will be assigned to the background of the element. As a result, the background will be transparent.

If the StripWidth property is set to 0.0, a line will be drawn and the BorderColor property will determine the color of the line, not this property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the gradient style of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration. The default value is None.

Attributes BindableAttribute

Remarks

The BackGradientStyle property determines whether a chart element will display a gradient, as well as the orientation of the gradient, if used.

To use gradient colors:

- Use the BackColor property to set the primary color.
- Use the BackSecondaryColor property to set the secondary color.
- Set this BackGradientStyle property of the chart element to the desired gradient style.

This property is not applicable if the StripWidth property is set to a value of 0.0.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatching style of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration. The default value is None.

Attributes BindableAttribute

Remarks

The hatch pattern consists of a solid background color and lines drawn over the background. The two colors used in the hatch style are set by the BackColor and BackSecondaryColor properties. This property is not applicable if the StripWidth property is set to a value of 0.0.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BackImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public string BackImage { get; set; }
```

Property Value

String

A `string` value that represents an image file's URL. The default is an empty string.

Attributes BindableAttribute

Remarks

The BackImage property defines a path to a background image for many chart elements, and is an absolute or relative URL.

To position this image, use the BackImageAlignment and BackImageWrapMode properties.

You can also specify that one color will be displayed by the image as "transparent" by setting the BackImageTransparentColor property.

An image can be any valid graphics file supported by .NET Framework, including icons.

This property is not applicable if the StripWidth property is set to a value of 0.0.

Background images take precedence over other background properties, such as gradient colors.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BackImageAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image alignment.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.ChartImageAlignmentStyle
BackImageAlignment { get; set; }
```

Property Value

ChartImageAlignmentStyle

A ChartImageAlignmentStyle enumeration value. The default value is TopLeft.

Attributes BindableAttribute

Remarks

This property determines the alignment of the background image for strip lines; it is only applied when the BackImageWrapMode property is set to Unscaled.

For more information about available alignments, see the ChartImageAlignmentStyle enumeration topic.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BackImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color which will be replaced with a transparent color when the background image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that will be displayed as transparent.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property is not applicable if the StripWidth property is set to a value of 0.0.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BackImageWrapMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the drawing mode of the background image of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartImageWrapMode
BackImageWrapMode { get; set; }
```

Property Value

ChartImageWrapMode

A ChartImageWrapMode integer enumeration. The default is Tile.

Attributes BindableAttribute

Remarks

This property determines how the image is displayed; that is, tiled, scaled, and so forth.

If an image is scaled, it will be stretched to fit within the bounding strip line. If the image is tiled, it is repeatedly drawn, starting from the top-left corner.

For more information concerning the available wrapping modes, see the ChartImageWrapMode enumeration topic.

If the BackImageWrapMode property is set to Unscaled, the location where the image is drawn will be determined by the BackImageAlignment property setting.

This property is not applicable if StripWidth is set to a value of 0.0.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color of the strip line background.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value used for the secondary color of a background with hatching or gradient fill. The default color is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property determines the secondary background color of many chart objects, and is used when gradient colors are being applied.

You can set background secondary color to any valid ARGB (alpha, red, green, blue) value.

If hatching is used, the BackSecondaryColor property determines the color of the hatching lines.

To use gradient colors for the background of a chart element:

- Use the BackColor property to set the primary color.
- Use this BackSecondaryColor property to set the secondary color.

- Set the BackGradientStyle property to the desired gradient style.

This property is not applicable if the StripWidth property is set to a value of 0.0.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of a strip line.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderColor { get; set; }
```

Property Value

Color

A Color value. The default color is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set this property to any valid ARGB (alpha, red, green, blue) value.

If the StripWidth property of a StripLine object is set to 0.0, a line will be drawn, and it is this property and not the BackColor property setting that will determine the color of the line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border style of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that determines the border style of the strip line.

Attributes BindableAttribute

Remarks

This property specifies the line style used for a border. The line can be solid, dotted or some variation of solid and dotted.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.BorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public int BorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that determines the width of the strip line's border, in pixels.

Attributes BindableAttribute

Remarks

This property is measured in pixels.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font used for the strip line text.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font value that represents the font of the strip line text.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the strip line text.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value that represents the text color of a strip line. The default value is Black.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can specify any valid ARGB color for this property. This property is only applicable when a text value is specified for the strip line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.Interval Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval for a strip line, and determines if the strip line is drawn once or repeatedly.

C#

```
[System.ComponentModel.Bindable(true)]
public double Interval { get; set; }
```

Property Value

Double

A `double` value that represents the interval between strip lines. The default value is zero.

Attributes BindableAttribute

Remarks

When you set the Interval property to a value of zero, one strip line will be displayed, with a width specified by the StripWidth property. When you set Interval to a value of other than zero, multiple strip lines will be shown. The unit that is used for the Interval property is defined by the IntervalType property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.IntervalOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the offset of grid lines, tick marks, strip lines and axis labels.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.AxisLabelDateValueConverter))]
public double IntervalOffset { get; set; }
```

Property Value

Double

A `double` value that represents the interval offset.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The interval offset of minor tick marks and grid lines is never affected by the value of the IntervalOffset property of the Axis object.

The IntervalOffset property setting defines the units of measurement of the offset.

The Interval can be a negative value.

The default value is `Auto` (`0`).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.IntervalOffsetType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval offset type of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalOffsetType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that indicates the interval type. The default value is Auto.

Attributes BindableAttribute

Remarks

By default, this property is set to Auto; the actual value of this property is determined by the IntervalType property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.IntervalType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the interval type of a StripLine object.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
IntervalType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that indicates the interval type. The default value is Auto.

Attributes BindableAttribute

Remarks

Interval types can be set to Auto, in which case the type depends on: the X-value or Y-value types of the plotted DataPoint objects - depending on which axis the StripLine objects belong to - and the axis scale.

The Interval property determines how often a strip or line is drawn.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the name of the strip line.

C#

```
[System.ComponentModel.Bindable(false)]
[System.ComponentModel.Browsable(false)]
public string Name { get; }
```

Property Value

String

A `string` value that represents the name of the strip line.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.StripWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the width of a strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public double StripWidth { get; set; }
```

Property Value

Double

A `double` value that determines whether a strip or a line is drawn. The default value is 0.0.

Attributes BindableAttribute

Remarks

This property cannot be set to a value less than zero, otherwise an exception will be raised.

A setting of 0.0 will result in a line being drawn; this line will use the BorderColor, BorderDashStyle and BorderWidth property settings for its color, width and style. None of the `Back*` properties in this class are utilized when the value of StripWidth is 0.0.

If you set the StripWidth property to a value greater than zero, a strip will be created. The BorderColor, BorderDashStyle and BorderWidth properties only apply to the border of the strip; all of the `Back*` properties are then used for the background properties of the strip.

If the value of the StripWidth property is greater than 0.0, the units are determined by the relevant axis scale.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.StripWidthType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unit of measurement for the StripWidth property.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
StripWidthType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that indicates the width type. The default value is Auto.

Attributes BindableAttribute

Remarks

This property determines the units used for the StripWidth property. For example, to set a strip line to represent "one week", you could set this property to Weeks and set the StripWidth property to one (1.0).

You can change the value of this property from Auto to a DateTime value when the relevant axis scale uses a DateTime value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.Text Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public string Text { get; set; }
```

Property Value

String

A `string` value that represents the text of a strip line.

Attributes BindableAttribute

Remarks

Use this property to add text to a strip line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.TextAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text alignment of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.StringAlignment TextAlignment { get; set; }
```

Property Value

StringAlignment

A StringAlignment value that represents the alignment of the strip line text.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.TextLineAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text line alignment of the strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.StringAlignment TextLineAlignment { get; set; }
```

Property Value

StringAlignment

A StringAlignment value that represents the alignment of the text line.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.TextOrientation Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text orientation.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.TextOrientation
TextOrientation { get; set; }
```

Property Value

TextOrientation

A TextOrientation value that represents the alignment of the text orientation.

Attributes BindableAttribute

Remarks

When this property is set to a value of Auto, the text is oriented based on the chart type. For all X-Y chart types other than Bar, X-axis strip line text is oriented horizontally from left to right on the bottom of the strip line. For Bar charts, X-axis strip line text is oriented vertically from top to bottom to the right of the strip line.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip of a strip line.

C#

```
[System.ComponentModel.Bindable(true)]
public string ToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip of the strip line.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLine.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the unmanaged resources used by the StripLine and optionally releases the managed resources.

```
C#
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

StripLinesCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of StripLine objects.

C#

```
public class StripLinesCollection :
System.Windows.Forms.DataVisualization.Charting.ChartElementCollection<Syste
m.Windows.Forms.DataVisualization.Charting.StripLine>
```

Inheritance Object → Collection<T> → ChartElementCollection<StripLine> → StripLinesCollection

Remarks

The StripLinesCollection class represents a collection of StripLine objects, and is implemented as the StripLines property of the Axis class.

StripLine objects can be appended, inserted or removed from the collection at both design time and run time.

Note Strip lines can be displayed once or repeatedly for a set interval.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)

Methods

Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts an element into the Collection<T> at the specified index location. (Inherited from ChartElementCollection<T>)

Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the element at the specified index location of the Collection<T>. (Inherited from ChartElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the element at the specified index location. (Inherited from ChartElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)

IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate,	Applies an accumulator function over a sequence. The specified seed value is used as

Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource> (IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.

Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.
Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to

	compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEqualityComparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.
GroupBy<TSource,TKey>(IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey>(IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement>(IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement>(IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult>(IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and

	creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	Correlates the elements of two sequences based on key equality and groups the results. A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>,	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.

Func<TOuter,TInner,TResult>, IEqualityComparer<TKey>)	
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.

Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Nullable<Single>>)	of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.

SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.
SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.

SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>,	Computes the sum of the sequence of Single

Func<TSource,Single>)	values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.
ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement>	Creates a Lookup<TKey,TElement> from an

(IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.
Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.

Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a text annotation.

C#

```
public class TextAnnotation :
System.Windows.Forms.DataVisualization.Charting.Annotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → TextAnnotation

Derived System.Windows.Forms.DataVisualization.Charting.CalloutAnnotation
System.Windows.Forms.DataVisualization.Charting.RectangleAnnotation

Remarks

Note that other annotations, such as rectangle and ellipse annotations, can display inner text.

Constructors

TextAnnotation()	Initializes a new instance of the TextAnnotation class.

Properties

Alignment	Gets or sets the content alignment of an annotation. (Inherited from Annotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)

AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)
AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the alignment of an annotation to the anchor point. (Inherited from Annotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the annotation type.

AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)
AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	This property is not applicable to the text annotation type.
BackGradientStyle	This property is not applicable to the text annotation type.
BackHatchStyle	This property is not applicable to the text annotation type.
BackSecondaryColor	This property is not applicable to the text annotation type.
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
Font	Gets or sets the font for the annotation text.
ForeColor	Gets or sets the text color of an annotation. (Inherited from Annotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsMultiline	Gets or sets a flag that indicates whether the annotation text is multiline.
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from Annotation)
LineColor	This property is not applicable to the text annotation type.
LineDashStyle	This property is not applicable to the text annotation type.
LineWidth	This property is not applicable to the text annotation type.

Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the text of the annotation.
TextStyle	Gets or sets the text style of an annotation. (Inherited from Annotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the mouse. (Inherited from Annotation)
BeginTextEditing()	Begins editing of the annotation text by an end-user.

BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends user placement of an annotation.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, DataPoint)	Anchors an annotation to two data points. (Inherited from Annotation)
StopTextEditing()	Stops end-user editing of the annotation text.
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the TextAnnotation class.

C#

```
public TextAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Remarks

For internal use, should not be used.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the text annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the text annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the text annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the text annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the text annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the text annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the text annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the text annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font for the annotation text.

C#

```
public override System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object used as the font for the annotation text.

Remarks

To set this property, create a new Font object, and then assign it to this property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.IsMultiline Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that indicates whether the annotation text is multiline.

C#

```
public virtual bool IsMultiline { get; set; }
```

Property Value

Boolean

`true` if the annotation text is multiline; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.LineColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the text annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public override System.Drawing.Color LineColor { get; set; }
```

Property Value

Color

A Color value.

Attributes BrowsableAttribute, TypeConverterAttribute

Remarks

This property is not applicable to the text annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.LineDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the text annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override
System.Windows.Forms.DataVisualization.Charting.ChartDashStyle LineDashStyle
{ get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the text annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.LineWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

This property is not applicable to the text annotation type.

C#

```
[System.ComponentModel.Browsable(false)]
public override int LineWidth { get; set; }
```

Property Value

Int32

An `integer` value.

Attributes BrowsableAttribute

Remarks

This property is not applicable to the text annotation type.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.Text Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text of the annotation.

C#

```
public virtual string Text { get; set; }
```

Property Value

String

A `string` value that represents the text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.BeginTextEditing Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Begins editing of the annotation text by an end-user.

C#

```
public void BeginTextEditing ();
```

Remarks

After calling this method, the annotation displays an editing box that allows the user to edit the annotation text.

Call the StopTextEditing method to cancel this mode programmatically. Note that editing also ends when the end-user hits the Enter key if the value of the IsMultiline property is `false`, or when the end-user clicks outside the editing box if the value is `true`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.EndPlacement Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Ends user placement of an annotation.

C#

```
public override void EndPlacement ();
```

Remarks

Ends an annotation placement operation previously started by a BeginPlacement method call. You are not required to call this method, because placement will automatically end when an end-user enters all required points. However, it is useful when an annotation placement operation needs to be aborted for some reason.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAnnotation.StopTextEditing Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Stops end-user editing of the annotation text.

C#

```
public void StopTextEditing ();
```

Remarks

Call this method to cancel a text editing operation that was started with a call to the BeginTextEditing method.

Note that if the end user double-clicks the annotation, the BeginTextEditing method will be automatically called.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextAntiAliasingQuality Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies the level of anti-aliasing quality to be used with the TextAntiAliasingQuality property.

C#

```
public enum TextAntiAliasingQuality
```

Inheritance Object → ValueType → Enum → TextAntiAliasingQuality

Fields

High	1	High anti-aliasing quality.
Normal	0	Normal anti-aliasing quality.
SystemDefault	2	System default anti-aliasing quality.

Remarks

The TextAntiAliasingQuality enumeration represents a quality range for text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextOrientation Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies text orientation in a chart element.

C#

```
public enum TextOrientation
```

Inheritance Object → ValueType → Enum → TextOrientation

Fields

Auto	0	Text orientation is automatically determined, based on the type of chart element in which the text appears.
Horizontal	1	Text is horizontal.
Rotated270	3	Text is rotated 270 degrees and oriented from bottom to top.
Rotated90	2	Text is rotated 90 degrees and oriented from top to bottom.
Stacked	4	Text characters are not rotated and are positioned one below the other.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TextStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies text drawing styles.

C#

```
public enum TextStyle
```

Inheritance Object → ValueType → Enum → TextStyle

Fields

Default	0	Default text drawing style.
Embed	3	Embedded text.
Emboss	2	Embossed text.
Frame	4	Framed text.
Shadow	1	Shadow text.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TickMark Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents the major and minor tick marks along an axis.

C#

```
public class TickMark : System.Windows.Forms.DataVisualization.Charting.Grid
```

Inheritance Object → Grid → TickMark

Remarks

The TickMark class encapsulates the functionality for all chart tick marks.

The TickMark class is exposed as the MajorTickMark and MinorTickMark properties of Axis objects.

By default, major tick marks are displayed at the major grid lines, whereas minor tick marks must be explicitly displayed.

MajorTickMark.Interval is not set (`Double.NaN`) by default, and the tick marks are drawn according to Interval. MinorTickMark.Interval is set to `Auto` (`0`) by default, and the Chart control automatically determines the interval.

A developer can use this class to:

- Enable or disable tick marks.
- Set the color, width, length and line style of tick marks.
- Specify a style that determines whether tick marks are displayed outside the axis, inside the axis or across the axis.
- Specify the starting point - where the first tick mark is drawn - as well as the tick mark interval, which correlates to their frequency.

Constructors

TickMark()	Initializes a new instance of the TickMark class.

Properties

Enabled	Gets or sets a flag that determines whether major or minor grid lines are enabled. (Inherited from Grid)
Interval	Gets or sets the interval between major or minor grid lines. (Inherited from Grid)
IntervalOffset	Gets or sets the offset of grid lines. (Inherited from Grid)
IntervalOffsetType	Gets or sets the interval offset type of major and minor grid lines. (Inherited from Grid)
IntervalType	Gets or sets the interval type for major or minor grid lines. (Inherited from Grid)
LineColor	Gets or sets the line color of a grid. (Inherited from Grid)
LineDashStyle	Gets or sets the line style of a grid. (Inherited from Grid)
LineWidth	Gets or sets the line width of major and minor grid lines. (Inherited from Grid)
Size	Gets or sets the length of tick marks. The length is measured in units that represent a percentage of the height or width of the chart image, depending on the axis.
TickMarkStyle	Gets or sets the tick mark style.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance.

	(Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TickMark Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the TickMark class.

C#

```
public TickMark ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TickMark.Size Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the length of tick marks. The length is measured in units that represent a percentage of the height or width of the chart image, depending on the axis.

C#

```
[System.ComponentModel.Bindable(true)]
public float Size { get; set; }
```

Property Value

Single

A `float` value that represents the size, or length, of tick marks. The default value is 1.0 percent.

Attributes BindableAttribute

Remarks

The size of a tick mark determines the tick mark's length.

The `Size` property is measured in units that represent a percentage of the height or width of the chart image. If the tick mark is displayed along a vertical axis, the unit of measure is a percentage of the chart image's width. If the tick mark is displayed along a horizontal axis, the unit of measure is a percentage of the chart image's height.

A change in the width of the chart image will result in the resizing of all tick marks displayed along vertical axes, for all chart areas. Similarly, a change in the height of the chart image will result in the resizing of all tick marks displayed along horizontal axes, for all chart areas.

To display tick marks for three-dimensional (3D) charts, so that they are just drawn for the width of a wall, set the TickMarkStyle property to AcrossAxis and specify a tick mark size of zero (0) using the Size property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TickMark.TickMarkStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tick mark style.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.TickMarkStyle
TickMarkStyle { get; set; }
```

Property Value

TickMarkStyle

A TickMarkStyle enumeration value. The default value is OutsideArea.

Attributes BindableAttribute

Remarks

The TickMarkStyle property determines the style of tick marks.

Tick marks can be drawn inside the chart area, outside the chart area, across an axis line, or can be disabled.

To display tick marks for three-dimensional (3D) charts so that they are just drawn for the width of a wall, set the TickMarkStyle property to TickMarkStyle.AcrossAxis. Then specify a tick mark size of zero (0) using the Size property.

If the TickMarkStyle property is set to TickMarkStyle.None, tick marks are not displayed. Tick marks can also be toggled on or off by setting the Enabled property to `true` or `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TickMarkStyle Enum

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Specifies a tick mark style when setting the TickMarkStyle property of a TickMark object.

C#

```
public enum TickMarkStyle
```

Inheritance Object → ValueType → Enum → TickMarkStyle

Fields

AcrossAxis	3	Tick marks are drawn across the relevant axis line.
InsideArea	2	Tick marks are drawn inside the chart area.
None	0	Tick marks are disabled.
OutsideArea	1	Tick marks are drawn outside the chart area.

Remarks

The TickMarkStyle enumeration represents the styles for major and minor tick marks.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a title for the entire chart image.

C#

```
public class Title :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElement,
IDisposable
```

Inheritance Object → ChartElement → ChartNamedElement → Title

Implements IDisposable

Remarks

The Title class encapsulates all the functionality of a title for the Chart control. This class is exposed using the Titles collection property of the root Chart object.

There can be any number of titles for an entire chart image.

Titles are able to:

- Be docked and aligned or custom-positioned.
- Be displayed inside or outside the plotted chart areas.
- Use a predefined TextStyle such as Embed or Shadow.
- Use hatching, gradient colors and background images.
- Be sized and positioned anywhere within the chart image.

Constructors

Title()	Initializes a new instance of the Title class.
Title(String)	Initializes a new instance of the Title class with the specified title text.
Title(String, Docking)	Initializes a new instance of the Title class with the specified title text and docking value.
Title(String, Docking, Font, Color)	Initializes a new instance of the Title class with the specified title text, docking value, font and color.

Properties

Alignment	Gets or sets the alignment of the title.
BackColor	Gets or sets the background color of the title.
BackGradientStyle	Gets or sets the orientation for the background gradient of a title. Also determines whether a gradient is used.
BackHatchStyle	Gets or sets the hatching style for the title.
BackImage	Gets or sets the background image of the title.
BackImageAlignment	Gets or sets the alignment of the title background image.
BackImageTransparentColor	Gets or sets a color that will be drawn as transparent when the background image is drawn.
BackImageWrapMode	Gets or sets the drawing mode for the background image of the title.
BackSecondaryColor	Gets or sets the secondary color of the title background.
BorderColor	Gets or sets the border color of the title.
BorderDashStyle	Gets or sets the border style of the title.
BorderWidth	Gets or sets the border width of the title.
DockedToChartArea	Gets or sets the name of the ChartArea where the title will be docked.
Docking	Gets or sets a value that determines where the title is docked.
DockingOffset	Gets or sets the positive or negative offset of the docked title position.
Font	Gets or sets the font for the title.
ForeColor	Gets or sets the color of the title text.

IsDockedInsideChartArea	Gets or sets a flag that specifies whether the title should be docked within a ChartArea object.
Name	Gets or sets the unique name of a Title object.
Position	Gets or sets an ElementPosition object, which can be used to get or set the position of the title.
ShadowColor	Gets or sets the shadow color for the title.
ShadowOffset	Gets or sets the shadow offset, in pixels, of the title.
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
Text	Gets or sets the text for the title.
TextOrientation	Gets or sets the orientation of the text in the title.
TextStyle	Gets or sets the style of the text for the title.
ToolTip	Gets or sets the tooltip for the title.
Visible	Gets or sets the visibility flag of the title.

Methods

Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases the unmanaged resources used by the Title and optionally releases the managed resources.
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the Title class.

Overloads

Title()	Initializes a new instance of the Title class.
Title(String)	Initializes a new instance of the Title class with the specified title text.
Title(String, Docking)	Initializes a new instance of the Title class with the specified title text and docking value.
Title(String, Docking, Font, Color)	Initializes a new instance of the Title class with the specified title text, docking value, font and color.

Title()

Initializes a new instance of the Title class.

C#

```
public Title ();
```

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title(String)

Initializes a new instance of the Title class with the specified title text.

```
C#

public Title (string text);
```

Parameters

`text` String

The text for the title.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title(String, Docking)

Initializes a new instance of the Title class with the specified title text and docking value.

```
C#

public Title (string text,
System.Windows.Forms.DataVisualization.Charting.Docking docking);
```

Parameters

`text` String

The text for the title.

`docking` Docking

A Docking enumeration value that determines where the title will be docked on the chart.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title(String, Docking, Font, Color)

Initializes a new instance of the Title class with the specified title text, docking value, font and color.

C#

```
public Title (string text,
System.Windows.Forms.DataVisualization.Charting.Docking docking,
System.Drawing.Font font, System.Drawing.Color color);
```

Parameters

`text` String

The text for the title.

`docking` Docking

A Docking enumeration value that determines where the title will be docked on the chart.

`font` Font

The title font.

`color` Color

The title color.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.Alignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of the title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.ContentAlignment Alignment { get; set; }
```

Property Value

ContentAlignment

A ContentAlignment enumeration value that represents the title alignment within the text drawing area. The default value is MiddleCenter.

Attributes BindableAttribute

Remarks

The Alignment property has no effect on alignment if the Position property is not set to Auto.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BackColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background color of the title.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackColor { get; set; }
```

Property Value

Color

A Color value that represents the background color of the title. The default color is White.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set this property to any valid ARGB (alpha, red, green, blue) value.

The BackColor value will be the first color used if you have specified gradient colors for the background.

The alpha value controls the opacity of a color. If you set this color to "transparent" - that is, if you use an alpha value of 0 - no color will be assigned to the background of the title. As a result, the background will be transparent.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BackGradientStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the orientation for the background gradient of a title. Also determines whether a gradient is used.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.GradientStyle
BackGradientStyle { get; set; }
```

Property Value

GradientStyle

A GradientStyle enumeration that represents the orientation for the background gradient of a title. The default value is None.

Attributes BindableAttribute

Remarks

Use the BackGradientStyle property to determine whether a title will display a gradient, and to determine the orientation of the gradient if one is used.

For a listing of available patterns, refer to the GradientStyle enumeration.

To use gradient colors:

Set the primary color of the title background with the BackColor property.

Set the secondary color of the title background with the BackSecondaryColor property.

Set this BackGradientStyle property to the desired gradient style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BackHatchStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the hatching style for the title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartHatchStyle
BackHatchStyle { get; set; }
```

Property Value

ChartHatchStyle

A ChartHatchStyle enumeration that specifies the hatching style for the title. The default value is None.

Attributes BindableAttribute

Remarks

The ChartHatchStyle enumeration specifies the hatch pattern used by a brush of type HatchBrush. The hatch pattern consists of a solid background color and lines drawn over the background.

For a listing of available hatching styles, refer to the ChartHatchStyle enumeration.

Note The color of hatching lines is determined by the BackSecondaryColor property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BackImage Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the background image of the title.

C#

```
[System.ComponentModel.Bindable(true)]
public string BackImage { get; set; }
```

Property Value

String

A `string` value that represents the URL of an image file. The default is an empty string.

Attributes BindableAttribute

Remarks

This property defines a path to a background image for a Chart title, and can be an absolute or relative URL.

To position the image, use the BackImageAlignment and BackImageWrapMode properties.

You can also specify one of the colors displayed by the image as "transparent", by setting the BackImageTransparentColor property.

An image be any valid graphics file supported by .NET Framework, including icons.

Background images take precedence over other background properties such as gradient color.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BackImageAlignment Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the alignment of the title background image.

C#

```
[System.ComponentModel.Bindable(true)]
public
System.Windows.Forms.DataVisualization.Charting.ChartImageAlignmentStyle
BackImageAlignment { get; set; }
```

Property Value

ChartImageAlignmentStyle

A ChartImageAlignmentStyle enumeration value that specifies the alignment of the title background image. The default value is TopLeft.

Attributes BindableAttribute

Remarks

This property is used when the BackImageWrapMode property is set to Unscaled.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BackImageTransparentColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a color that will be drawn as transparent when the background image is drawn.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackImageTransparentColor { get; set; }
```

Property Value

Color

A Color value that specifies the color that will be drawn as transparent when the background image is drawn. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This color will be displayed as transparent when the chart is rendered.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BackImageWrapMode Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the drawing mode for the background image of the title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartImageWrapMode
BackImageWrapMode { get; set; }
```

Property Value

ChartImageWrapMode

A ChartImageWrapMode enumeration value that specifies the drawing mode for the background image of the title. The default value is Tile.

Attributes BindableAttribute

Remarks

The BackImageWrapMode property determines how the image is displayed; the image can be tiled, scaled, and so forth.

If an image is scaled, it will be stretched to fit within the bounds of the title. If the image is tiled, it is repeatedly drawn, starting from the top-left corner of the title.

For more information about the available wrapping modes, see the ChartImageWrapMode enumeration topic.

If the BackImageWrapMode property is set to Unscaled, the location where the image is drawn is determined by the BackImageAlignment property setting.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BackSecondaryColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the secondary color of the title background.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BackSecondaryColor { get; set; }
```

Property Value

Color

A Color value used for the secondary color of a background with hatching or gradient fill. The default value is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This BackSecondaryColor property is one of several related properties that determine the secondary background color of Chart objects, and is used when gradient colors are being applied.

You can set BackSecondaryColor to any valid ARGB (alpha, red, green, blue) value.

If hatching is used, the BackSecondaryColor property determines the color of the hatching lines for the title.

To use gradient colors for the background of a title:

- Set the primary color with the BackColor property.
- Set the secondary color with the BackSecondaryColor property.

- Set the BackGradientStyle property to the desired gradient style.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BorderColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border color of the title.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color BorderColor { get; set; }
```

Property Value

Color

A Color value that specifies the border color of the title. The default color is Empty.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

You can set the BorderColor property to any valid ARGB (alpha, red, green, blue) value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BorderDashStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border style of the title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.ChartDashStyle
BorderDashStyle { get; set; }
```

Property Value

ChartDashStyle

A ChartDashStyle enumeration value that determines the border style of the title.

Attributes BindableAttribute

Remarks

The BorderDashStyle property determines the line style to use for a border. The line can be solid, dotted, and so forth.

For other border styles, see the ChartDashStyle enumeration topic.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.BorderWidth Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the border width of the title.

C#

```
[System.ComponentModel.Bindable(true)]
public int BorderWidth { get; set; }
```

Property Value

Int32

An `integer` value that determines the border width, in pixels, of the title.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.DockedToChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the name of the ChartArea where the title will be docked.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.LegendAreaNameConverter))]
public string DockedToChartArea { get; set; }
```

Property Value

String

A `string` value that represents the name of the ChartArea where this title will be docked. The default value is the empty string.

Attributes BindableAttribute, TypeConverterAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.Docking Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a value that determines where the title is docked.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.Docking Docking {
get; set; }
```

Property Value

Docking

A Docking enumeration value that determines where the title is docked. The default value is Top.

Attributes BindableAttribute

Remarks

The Title can be docked to the top, left, bottom or right of either the entire chart image, or the inside or outside of a chart area. If the IsDockedInsideChartArea property is set to `false`, docking applies to the entire chart image. If the IsDockedInsideChartArea property is set to `true`, docking applies to the ChartArea object specified by the DockedToChartArea property

This property has no effect if the Position property is not set to Auto.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.DockingOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the positive or negative offset of the docked title position.

C#

```
[System.ComponentModel.Bindable(true)]
public int DockingOffset { get; set; }
```

Property Value

Int32

An `integer` value that specifies the positive or negative offset of the docked title position.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.Font Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the font for the title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Drawing.Font Font { get; set; }
```

Property Value

Font

A Font object that represents the font for the title. Defaults to "Microsoft Sans Serif, 8pt".

Attributes BindableAttribute

Remarks

Use a new Font object when setting this property. Assign the entire object to the property.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.ForeColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the color of the title text.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ForeColor { get; set; }
```

Property Value

Color

A Color value that is used to draw the color of the title text .

Attributes BindableAttribute, TypeConverterAttribute

Remarks

The ForeColor property is used to set the title text color.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.IsDockedInsideChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets a flag that specifies whether the title should be docked within a ChartArea object.

C#

```
[System.ComponentModel.Bindable(true)]
public bool IsDockedInsideChartArea { get; set; }
```

Property Value

Boolean

`true` if the title will be docked within a chart area; `false` if the title will be docked outside the chart area. The default value is `false`.

Attributes BindableAttribute

Remarks

If the IsDockedInsideChartArea property is set to `false`, the title is docked to the outside edge of the plotted chart areas; if this property is set to `true`, the title will be docked to the inside edge of the chart area. This property will have no effect if the name of a ChartArea object is not assigned to the DockedToChartArea property.

Note This behavior only applies if the Position property is set to Auto. If this is not the case, setting the IsDockedInsideChartArea property has no effect, because the position of the title is then determined by the Position property value.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.Name Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the unique name of a Title object.

C#

```
[System.ComponentModel.Bindable(true)]
public override string Name { get; set; }
```

Property Value

String

A `string` value that represents the unique name of the title object.

Attributes BindableAttribute

Remarks

The name of a title object must be unique; otherwise an exception will be thrown.

This property can be used as the indexer when you retrieve a Title object from the TitleCollection object.

If this property is set to a zero-length string, an exception will be thrown.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.Position Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets an ElementPosition object, which can be used to get or set the position of the title.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Windows.Forms.DataVisualization.Charting.ElementPositionConverter))]
public System.Windows.Forms.DataVisualization.Charting.ElementPosition
Position { get; set; }
```

Property Value

ElementPosition

An ElementPosition value that represents the position of the title. The default value is Auto.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

If the Position property is automatic (that is, Title.Position.Auto = `true`) the title position is calculated automatically by the Chart control, taking into account the Docking, Alignment and IsDockedInsideChartArea property settings.

If it is not automatic, the Docking, DockedToChartArea and IsDockedInsideChartArea properties are ignored, and the title position is solely determined by the value of the returned ElementPosition object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.ShadowColor Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow color for the title.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.TypeConverter(typeof(System.Drawing.ColorConverter))]
public System.Drawing.Color ShadowColor { get; set; }
```

Property Value

Color

A Color value that specifies the shadow color for the title. The default color is Black.

Attributes BindableAttribute, TypeConverterAttribute

Remarks

This property can be set to any valid ARGB (alpha, red, green, blue) value.

The alpha value can be used to achieve a "realistic" shadowing effect; try a color value of "128,0,0,0" for shadows.

To implement a shadow, set the ShadowOffset property to a value other than zero (0).

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.ShadowOffset Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the shadow offset, in pixels, of the title.

C#

```
[System.ComponentModel.Bindable(true)]
public int ShadowOffset { get; set; }
```

Property Value

Int32

An `integer` value that represents the shadow offset, in pixels, of the title.

Attributes BindableAttribute

Remarks

To implement shadowing, set this property to a value other than zero (0). A positive value will result in a shadow that shifts down and to the right; a negative value will result in a shadow that shifts up and to the left.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.Text Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the text for the title.

C#

```
[System.ComponentModel.Bindable(true)]
public string Text { get; set; }
```

Property Value

String

A `string` value that represents the text for the title.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.TextOrientation Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the orientation of the text in the title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.TextOrientation
TextOrientation { get; set; }
```

Property Value

TextOrientation

A TextOrientation enumeration value that determines the orientation of the text in the title.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.TextStyle Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the style of the text for the title.

C#

```
[System.ComponentModel.Bindable(true)]
public System.Windows.Forms.DataVisualization.Charting.TextStyle TextStyle {
get; set; }
```

Property Value

TextStyle

A TextStyle enumeration value that determines the style of the text for the title. The default value is Default.

Attributes BindableAttribute

Remarks

There are five possible styles for title text: Default, Shadow, Emboss, Embed, and Frame. Refer to the table below for a summary of these styles:

Style	Description
Default	Text is displayed using the desired font and color, with no additional effects.
Shadow	Text is displayed using the desired font and color with an added shadow effect on the text.
Emboss	Text is displayed using the desired font and color but drawn with an embossing effect.
Embed	Text is displayed using the desired font and color but drawn with an embedding effect.
Frame	Text is displayed using the desired font and color but drawn as an outline of the text, that is, with no fill.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.ToolTip Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip for the title.

C#

```
[System.ComponentModel.Bindable(true)]
public string ToolTip { get; set; }
```

Property Value

String

A `string` value that represents the tooltip for the title.

Attributes BindableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.Visible Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the visibility flag of the title.

C#

```
public virtual bool Visible { get; set; }
```

Property Value

Boolean

`true` if the title is visible; otherwise, `false`.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Title.Dispose(Boolean) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Releases the unmanaged resources used by the Title and optionally releases the managed resources.

C#

```
protected override void Dispose (bool disposing);
```

Parameters

`disposing` Boolean

`true` to release both managed and unmanaged resources; `false` to release only unmanaged resources.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TitleCollection Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a collection of Title objects.

C#

```
public class TitleCollection :
System.Windows.Forms.DataVisualization.Charting.ChartNamedElementCollection<
System.Windows.Forms.DataVisualization.Charting.Title>
```

Inheritance Object → Collection<T> → ChartElementCollection<T> → ChartNamedElementCollection<Title> → TitleCollection

Remarks

The TitleCollection class is implemented as the Titles collection property of the root Chart object.

Title objects can be appended, inserted or removed from the collection at design time and run time.

Properties

Count	Gets the number of elements actually contained in the Collection<T>. (Inherited from Collection<T>)
Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
Item[String]	Gets or sets the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
Items	Gets a IList<T> wrapper around the Collection<T>. (Inherited from Collection<T>)

NamePrefix	Gets the name prefix that is used to create unique chart element names. (Inherited from ChartNamedElementCollection<T>)

Methods

Add(String)	Adds a Title object with the specified name to the end of the collection.
Add(T)	Adds an object to the end of the Collection<T>. (Inherited from Collection<T>)
Clear()	Removes all elements from the Collection<T>. (Inherited from Collection<T>)
ClearItems()	Removes all elements from the Collection<T>. (Inherited from ChartElementCollection<T>)
Contains(T)	Determines whether an element is in the Collection<T>. (Inherited from Collection<T>)
CopyTo(T[], Int32)	Copies the entire Collection<T> to a compatible one-dimensional Array, starting at the specified index of the target array. (Inherited from Collection<T>)
Dispose()	Releases the resources used by the ChartElementCollection<T>. (Inherited from ChartElementCollection<T>)
Dispose(Boolean)	Releases the unmanaged resources used by the ChartElementCollection<T> and optionally releases the managed resources. (Inherited from ChartElementCollection<T>)
Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
FindByName(String)	Finds the chart element by the specified name. (Inherited from ChartNamedElementCollection<T>)
GetEnumerator()	Returns an enumerator that iterates through the Collection<T>. (Inherited from Collection<T>)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)

IndexOf(String)	Finds the index of the chart element with the specified name. (Inherited from ChartNamedElementCollection<T>)
IndexOf(T)	Searches for the specified object and returns the zero-based index of the first occurrence within the entire Collection<T>. (Inherited from Collection<T>)
Insert(Int32, T)	Inserts an element into the Collection<T> at the specified index. (Inherited from Collection<T>)
InsertItem(Int32, T)	Inserts the specified item in the collection at the specified index. (Inherited from ChartNamedElementCollection<T>)
Invalidate()	Forces the invalidation of the parent chart element. (Inherited from ChartElementCollection<T>)
IsUniqueName(String)	Determines whether the chart element with the specified name already exists in the collection. (Inherited from ChartNamedElementCollection<T>)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
NextUniqueName()	Finds the unique name for the new element to be added to the collection. (Inherited from ChartNamedElementCollection<T>)
Remove(T)	Removes the first occurrence of a specific object from the Collection<T>. (Inherited from Collection<T>)
RemoveAt(Int32)	Removes the element at the specified index of the Collection<T>. (Inherited from Collection<T>)
RemoveItem(Int32)	Removes the chart element at the specified index of the collection. (Inherited from ChartNamedElementCollection<T>)
ResumeUpdates()	Resumes the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
SetItem(Int32, T)	Replaces the chart element at the specified index with a new chart element. (Inherited from ChartNamedElementCollection<T>)
SuspendUpdates()	Suspends the invalidation of a chart element in the collection. (Inherited from ChartElementCollection<T>)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Explicit Interface Implementations

ICollection.CopyTo(Array, Int32)	Copies the elements of the ICollection to an Array, starting at a particular Array index. (Inherited from Collection<T>)
ICollection.IsSynchronized	Gets a value indicating whether access to the ICollection is synchronized (thread safe). (Inherited from Collection<T>)
ICollection.SyncRoot	Gets an object that can be used to synchronize access to the ICollection. (Inherited from Collection<T>)
ICollection<T>.IsReadOnly	Gets a value indicating whether the ICollection<T> is read-only. (Inherited from Collection<T>)
IEnumerable.GetEnumerator()	Returns an enumerator that iterates through a collection. (Inherited from Collection<T>)
IList.Add(Object)	Adds an item to the IList. (Inherited from Collection<T>)
IList.Contains(Object)	Determines whether the IList contains a specific value. (Inherited from Collection<T>)
IList.IndexOf(Object)	Determines the index of a specific item in the IList. (Inherited from Collection<T>)
IList.Insert(Int32, Object)	Inserts an item into the IList at the specified index. (Inherited from Collection<T>)
IList.IsFixedSize	Gets a value indicating whether the IList has a fixed size. (Inherited from Collection<T>)
IList.IsReadOnly	Gets a value indicating whether the IList is read-only. (Inherited from Collection<T>)
IList.Item[Int32]	Gets or sets the element at the specified index. (Inherited from Collection<T>)
IList.Remove(Object)	Removes the first occurrence of a specific object from the IList. (Inherited from Collection<T>)

Extension Methods

CopyToDataTable<T>(IEnumerable<T>)	Returns a DataTable that contains copies of the DataRow objects, given an input

	IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
CopyToDataTable<T>(IEnumerable<T>, DataTable, LoadOption, FillErrorEventHandler)	Copies DataRow objects to the specified DataTable, given an input IEnumerable<T> object where the generic parameter `T` is DataRow.
Aggregate<TSource>(IEnumerable<TSource>, Func<TSource,TSource,TSource>)	Applies an accumulator function over a sequence.
Aggregate<TSource,TAccumulate>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value.
Aggregate<TSource,TAccumulate,TResult>(IEnumerable<TSource>, TAccumulate, Func<TAccumulate,TSource,TAccumulate>, Func<TAccumulate,TResult>)	Applies an accumulator function over a sequence. The specified seed value is used as the initial accumulator value, and the specified function is used to select the result value.
All<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether all elements of a sequence satisfy a condition.
Any<TSource>(IEnumerable<TSource>)	Determines whether a sequence contains any elements.
Any<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Determines whether any element of a sequence satisfies a condition.
Append<TSource>(IEnumerable<TSource>, TSource)	Appends a value to the end of the sequence.
AsEnumerable<TSource>(IEnumerable<TSource>)	Returns the input typed as IEnumerable<T>.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the average of a sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the average of a sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the average of a sequence of Int32 values that are obtained by invoking a

	transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the average of a sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the average of a sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the average of a sequence of nullable Double values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the average of a sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the average of a sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the average of a sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Average<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the average of a sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Cast<TResult>(IEnumerable)	Casts the elements of an IEnumerable to the specified type.
Concat<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Concatenates two sequences.
Contains<TSource>(IEnumerable<TSource>, TSource)	Determines whether a sequence contains a specified element by using the default equality comparer.
Contains<TSource>(IEnumerable<TSource>, TSource, IEqualityComparer<TSource>)	Determines whether a sequence contains a specified element by using a specified IEqualityComparer<T>.

Count<TSource>(IEnumerable<TSource>)	Returns the number of elements in a sequence.
Count<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns a number that represents how many elements in the specified sequence satisfy a condition.
DefaultIfEmpty<TSource> (IEnumerable<TSource>)	Returns the elements of the specified sequence or the type parameter's default value in a singleton collection if the sequence is empty.
DefaultIfEmpty<TSource> (IEnumerable<TSource>, TSource)	Returns the elements of the specified sequence or the specified value in a singleton collection if the sequence is empty.
Distinct<TSource>(IEnumerable<TSource>)	Returns distinct elements from a sequence by using the default equality comparer to compare values.
Distinct<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Returns distinct elements from a sequence by using a specified IEqualityComparer<T> to compare values.
ElementAt<TSource>(IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence.
ElementAtOrDefault<TSource> (IEnumerable<TSource>, Int32)	Returns the element at a specified index in a sequence or a default value if the index is out of range.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set difference of two sequences by using the default equality comparer to compare values.
Except<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set difference of two sequences by using the specified IEqualityComparer<T> to compare values.
First<TSource>(IEnumerable<TSource>)	Returns the first element of a sequence.
First<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element in a sequence that satisfies a specified condition.
FirstOrDefault<TSource> (IEnumerable<TSource>)	Returns the first element of a sequence, or a default value if the sequence contains no elements.
FirstOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the first element of the sequence that satisfies a condition or a default value if no such element is found.

GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Groups the elements of a sequence according to a specified key selector function.
GroupBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and compares the keys by using a specified comparer.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Groups the elements of a sequence according to a specified key selector function and projects the elements for each group by using a specified function.
GroupBy<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Groups the elements of a sequence according to a key selector function. The keys are compared by using a comparer and each group's elements are projected by using a specified function.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key.
GroupBy<TSource,TKey,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TKey,IEnumerable<TSource>,TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The keys are compared by using a specified comparer.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>,TResult>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. The elements of each group are projected by using a specified function.
GroupBy<TSource,TKey,TElement,TResult> (IEnumerable<TSource>, Func<TSource, TKey>, Func<TSource,TElement>, Func<TKey,IEnumerable<TElement>, TResult>, IEqualityComparer<TKey>)	Groups the elements of a sequence according to a specified key selector function and creates a result value from each group and its key. Key values are compared by using a specified comparer, and the elements of each group are projected by using a specified function.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>)	Correlates the elements of two sequences based on equality of keys and groups the results. The default equality comparer is used to compare keys.
GroupJoin<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>,	Correlates the elements of two sequences based on key equality and groups the results.

Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,IEnumerable<TInner>, TResult>, IEqualityComparer<TKey>)	A specified IEqualityComparer<T> is used to compare keys.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set intersection of two sequences by using the default equality comparer to compare values.
Intersect<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set intersection of two sequences by using the specified IEqualityComparer<T> to compare values.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>)	Correlates the elements of two sequences based on matching keys. The default equality comparer is used to compare keys.
Join<TOuter,TInner,TKey,TResult> (IEnumerable<TOuter>, IEnumerable<TInner>, Func<TOuter,TKey>, Func<TInner,TKey>, Func<TOuter,TInner,TResult>, IEquality Comparer<TKey>)	Correlates the elements of two sequences based on matching keys. A specified IEqualityComparer<T> is used to compare keys.
Last<TSource>(IEnumerable<TSource>)	Returns the last element of a sequence.
Last<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a specified condition.
LastOrDefault<TSource> (IEnumerable<TSource>)	Returns the last element of a sequence, or a default value if the sequence contains no elements.
LastOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the last element of a sequence that satisfies a condition or a default value if no such element is found.
LongCount<TSource>(IEnumerable<TSource>)	Returns an Int64 that represents the total number of elements in a sequence.
LongCount<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns an Int64 that represents how many elements in a sequence satisfy a condition.
Max<TSource>(IEnumerable<TSource>)	Returns the maximum value in a generic sequence.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the maximum Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the maximum Double value.

Max<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the maximum Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Invokes a transform function on each element of a sequence and returns the maximum Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Decimal value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Double value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int32 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Int64 value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the maximum nullable Single value.
Max<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the maximum Single value.
Max<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the maximum resulting value.
Min<TSource>(IEnumerable<TSource>)	Returns the minimum value in a generic sequence.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Invokes a transform function on each element of a sequence and returns the minimum Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Invokes a transform function on each element of a sequence and returns the minimum Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Invokes a transform function on each element of a sequence and returns the minimum Int32 value.
Min<TSource>(IEnumerable<TSource>,	Invokes a transform function on each element

Func<TSource,Int64>)	of a sequence and returns the minimum Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Decimal value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Double value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int32 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Int64 value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Invokes a transform function on each element of a sequence and returns the minimum nullable Single value.
Min<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Invokes a transform function on each element of a sequence and returns the minimum Single value.
Min<TSource,TResult>(IEnumerable<TSource>, Func<TSource,TResult>)	Invokes a transform function on each element of a generic sequence and returns the minimum resulting value.
OfType<TResult>(IEnumerable)	Filters the elements of an IEnumerable based on a specified type.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in ascending order according to a key.
OrderBy<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in ascending order by using a specified comparer.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Sorts the elements of a sequence in descending order according to a key.
OrderByDescending<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IComparer<TKey>)	Sorts the elements of a sequence in descending order by using a specified comparer.
Prepend<TSource>(IEnumerable<TSource>, TSource)	Adds a value to the beginning of the sequence.
Reverse<TSource>(IEnumerable<TSource>)	Inverts the order of the elements in a

	sequence.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,TResult>)	Projects each element of a sequence into a new form.
Select<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,TResult>)	Projects each element of a sequence into a new form by incorporating the element's index.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T> and flattens the resulting sequences into one sequence.
SelectMany<TSource,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TResult>>)	Projects each element of a sequence to an IEnumerable<T>, and flattens the resulting sequences into one sequence. The index of each source element is used in the projected form of that element.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein.
SelectMany<TSource,TCollection,TResult> (IEnumerable<TSource>, Func<TSource,Int32,IEnumerable<TCollection>>, Func<TSource,TCollection,TResult>)	Projects each element of a sequence to an IEnumerable<T>, flattens the resulting sequences into one sequence, and invokes a result selector function on each element therein. The index of each source element is used in the intermediate projected form of that element.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>)	Determines whether two sequences are equal by comparing the elements by using the default equality comparer for their type.
SequenceEqual<TSource> (IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Determines whether two sequences are equal by comparing their elements by using a specified IEqualityComparer<T>.
Single<TSource>(IEnumerable<TSource>)	Returns the only element of a sequence, and throws an exception if there is not exactly one element in the sequence.
Single<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition, and throws an exception if more than one such element exists.

SingleOrDefault<TSource> (IEnumerable<TSource>)	Returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
SingleOrDefault<TSource> (IEnumerable<TSource>, Func<TSource,Boolean>)	Returns the only element of a sequence that satisfies a specified condition or a default value if no such element exists; this method throws an exception if more than one element satisfies the condition.
Skip<TSource>(IEnumerable<TSource>, Int32)	Bypasses a specified number of elements in a sequence and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements.
SkipWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Bypasses elements in a sequence as long as a specified condition is true and then returns the remaining elements. The element's index is used in the logic of the predicate function.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Decimal>)	Computes the sum of the sequence of Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Double>)	Computes the sum of the sequence of Double values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int32>)	Computes the sum of the sequence of Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Int64>)	Computes the sum of the sequence of Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Decimal>>)	Computes the sum of the sequence of nullable Decimal values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Double>>)	Computes the sum of the sequence of nullable Double values that are obtained by

	invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int32>>)	Computes the sum of the sequence of nullable Int32 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Int64>>)	Computes the sum of the sequence of nullable Int64 values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Nullable<Single>>)	Computes the sum of the sequence of nullable Single values that are obtained by invoking a transform function on each element of the input sequence.
Sum<TSource>(IEnumerable<TSource>, Func<TSource,Single>)	Computes the sum of the sequence of Single values that are obtained by invoking a transform function on each element of the input sequence.
Take<TSource>(IEnumerable<TSource>, Int32)	Returns a specified number of contiguous elements from the start of a sequence.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Returns elements from a sequence as long as a specified condition is true.
TakeWhile<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Returns elements from a sequence as long as a specified condition is true. The element's index is used in the logic of the predicate function.
ToArray<TSource>(IEnumerable<TSource>)	Creates an array from a IEnumerable<T>.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function.
ToDictionary<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to specified key selector and element selector functions.
ToDictionary<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Dictionary<TKey,TValue> from an IEnumerable<T> according to a specified key selector function, a comparer, and an element selector function.

ToHashSet<TSource>(IEnumerable<TSource>)	Creates a HashSet<T> from an IEnumerable<T>.
ToHashSet<TSource>(IEnumerable<TSource>, IEqualityComparer<TSource>)	Creates a HashSet<T> from an IEnumerable<T> using the `comparer` to compare keys.
ToList<TSource>(IEnumerable<TSource>)	Creates a List<T> from an IEnumerable<T>.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function.
ToLookup<TSource,TKey> (IEnumerable<TSource>, Func<TSource,TKey>, IEqualityComparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function and key comparer.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to specified key selector and element selector functions.
ToLookup<TSource,TKey,TElement> (IEnumerable<TSource>, Func<TSource,TKey>, Func<TSource,TElement>, IEquality Comparer<TKey>)	Creates a Lookup<TKey,TElement> from an IEnumerable<T> according to a specified key selector function, a comparer and an element selector function.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>)	Produces the set union of two sequences by using the default equality comparer.
Union<TSource>(IEnumerable<TSource>, IEnumerable<TSource>, IEquality Comparer<TSource>)	Produces the set union of two sequences by using a specified IEqualityComparer<T>.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Boolean>)	Filters a sequence of values based on a predicate.
Where<TSource>(IEnumerable<TSource>, Func<TSource,Int32,Boolean>)	Filters a sequence of values based on a predicate. Each element's index is used in the logic of the predicate function.
Zip<TFirst,TSecond,TResult> (IEnumerable<TFirst>, IEnumerable<TSecond>, Func<TFirst,TSecond,TResult>)	Applies a specified function to the corresponding elements of two sequences, producing a sequence of the results.
AsParallel(IEnumerable)	Enables parallelization of a query.
AsParallel<TSource>(IEnumerable<TSource>)	Enables parallelization of a query.
AsQueryable(IEnumerable)	Converts an IEnumerable to an IQueryable.
AsQueryable<TElement> (IEnumerable<TElement>)	Converts a generic IEnumerable<T> to a generic IQueryable<T>.

Ancestors<T>(IEnumerable<T>)	Returns a collection of elements that contains the ancestors of every node in the source collection.
Ancestors<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the ancestors of every node in the source collection. Only elements that have a matching XName are included in the collection.
DescendantNodes<T>(IEnumerable<T>)	Returns a collection of the descendant nodes of every document and element in the source collection.
Descendants<T>(IEnumerable<T>)	Returns a collection of elements that contains the descendant elements of every element and document in the source collection.
Descendants<T>(IEnumerable<T>, XName)	Returns a filtered collection of elements that contains the descendant elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
Elements<T>(IEnumerable<T>)	Returns a collection of the child elements of every element and document in the source collection.
Elements<T>(IEnumerable<T>, XName)	Returns a filtered collection of the child elements of every element and document in the source collection. Only elements that have a matching XName are included in the collection.
InDocumentOrder<T>(IEnumerable<T>)	Returns a collection of nodes that contains all nodes in the source collection, sorted in document order.
Nodes<T>(IEnumerable<T>)	Returns a collection of the child nodes of every document and element in the source collection.
Remove<T>(IEnumerable<T>)	Removes every node in the source collection from its parent node.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TitleCollection.Add(String) Method

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Adds a Title object with the specified name to the end of the collection.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Title Add (string
name);
```

Parameters

name String

The name of the title to add to the collection.

Returns

Title

Index of the added object.

Remarks

A `string` value that represents the zero-based index of the added title.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ToolTipEventArgs Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Used as a parameter object in the GetToolTipText event of the root Chart object.

```
C#
public class ToolTipEventArgs : EventArgs
```

Inheritance Object → EventArgs → ToolTipEventArgs

Remarks

This class is exposed as the `e` parameter in the GetToolTipText event, and provides the mouse position, tooltip text, if any, and details about the chart element over which the mouse cursor is held when the event is raised.

The GetToolTipText event is raised several seconds after the mouse cursor is held over a relevant chart element.

The ToolTipEventArgs class contains the following properties:

- The X and Y properties, which are used to get the mouse coordinates.
- The Text property, which is used to get the tooltip for the chart element that the event is raised for. Note that many of the chart elements have a `ToolTip`, `LegendToolTip`, and/or `LabelToolTip` property that you can use to assign a tooltip. These elements include legends, annotations, data points, series, axes, titles, custom labels, and strip lines. This property can also be used to override the tooltip for a chart element, or to set a tooltip for a chart element that doesn't have a `ToolTip` property.
- The HitTestResult property, which provides information on the chart element for which the event was raised.

Constructors

ToolTipEventArgs(Int32, Int32, String, HitTestResult)	Initializes a new instance of the ToolTipEventArgs class with the specified `x`, `y`, `text` and `result` parameters.

Properties

HitTestResult	Gets a HitTestResult object, which provides information about the chart element that the GetToolTipText event was raised for.
Text	Gets or sets the tooltip, if any, of the chart element over which the mouse cursor is held. This property can also set a tooltip for this chart element.
X	Gets the X coordinate of the point the mouse cursor is held over when the GetToolTipText event is raised.
Y	Gets the Y coordinate of the point the mouse cursor is held over when the GetToolTipText event is raised.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ToolTipEventArgs(Int32, Int32, String, HitTestResult) Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ToolTipEventArgs class with the specified `x`, `y`, `text` and `result` parameters.

C#

```
public ToolTipEventArgs (int x, int y, string text,
System.Windows.Forms.DataVisualization.Charting.HitTestResult result);
```

Parameters

`x` Int32

X-coordinate of mouse.

`y` Int32

Y-coordinate of mouse.

`text` String

The tooltip text.

`result` HitTestResult

The HitTestResult object.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ToolTipEventArgs.HitTestResult Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets a HitTestResult object, which provides information about the chart element that the GetToolTipText event was raised for.

C#

```
public System.Windows.Forms.DataVisualization.Charting.HitTestResult
HitTestResult { get; }
```

Property Value

HitTestResult

A HitTestResult object that represents information for the element the GetToolTipText event was raised for.

Remarks

This property is exposed in the GetToolTipText event of the root Chart object.

The HitTestResult property can be used to get the object the event was raised for, the type of the chart element the event was raised for, and so forth. For further details, see the HitTestResult class overview.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ToolTipEventArgs.Text Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the tooltip, if any, of the chart element over which the mouse cursor is held. This property can also set a tooltip for this chart element.

C#

```
public string Text { get; set; }
```

Property Value

String

A `string` value that represents the tooltip text of the element the GetToolTipText event was raised for.

Remarks

This property is exposed in the GetToolTipText event of the root Chart object.

- The Text property, which is used to get the tooltip for the chart element that the event is raised for. Note that many of the chart elements have a `ToolTip`, `LegendToolTip`, and/or `LabelToolTip` property that you can use to assign a tooltip. These elements include legends, annotations, data points, series, axes, titles, custom labels, and strip lines.
- If you get this property, it will always return a zero-length string for any chart element items that do not have a tooltip.

You can set this property to:

- Override the tooltip for any of the chart elements capable of assigning tooltips.
- Assign a tooltip to a chart element that does not support tooltips.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ToolTipEventArgs.X Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the X coordinate of the point the mouse cursor is held over when the GetToolTipText event is raised.

```
C#
public int X { get; }
```

Property Value

Int32

An `integer` value that represents the X coordinate of the mouse cursor.

Remarks

This property is exposed in the GetToolTipText event of the root Chart object.

Use the X property to get the X screen coordinate of the point the mouse cursor is held over when the GetToolTipText event is raised.

Note that the X coordinate is measured in pixels; the point (0,0) corresponds to the top-left corner of the chart.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ToolTipEventArgs.Y Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Y coordinate of the point the mouse cursor is held over when the GetToolTipText event is raised.

C#

```
public int Y { get; }
```

Property Value

Int32

An `integer` that represents the Y coordinate of the mouse cursor.

Remarks

This property is exposed in the GetToolTipText event of the root Chart object.

Use the Y property to get the Y screen coordinate of the point the mouse cursor is held over when the GetToolTipText event is raised.

Note that the Y coordinate is measured in pixels; the point (0,0) corresponds to the top-left corner of the chart.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Stores the results of a T Test statistical formula.

C#

```
public class TTestResult
```

Inheritance Object → TTestResult

Constructors

TTestResult()	Initializes a new instance of the TTestResult class.

Properties

DegreeOfFreedom	Gets the degree of freedom.
FirstSeriesMean	Gets the mean of the first series.
FirstSeriesVariance	Gets the variance of the first series.
ProbabilityTOneTail	Gets the Probability T one tail value.
ProbabilityTTwoTail	Gets the Probability T two tail value.
SecondSeriesMean	Gets the mean of the second series.
SecondSeriesVariance	Gets the variance of the second series.
TCriticalValueOneTail	Gets the Critical T one tail value.
TCriticalValueTwoTail	Gets the Critical T two tail value.
TValue	Gets the T value.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the TTestResult class.

C#

```
public TTestResult ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.DegreeOfFreedom Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the degree of freedom.

C#

```
public double DegreeOfFreedom { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.FirstSeriesMean Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the mean of the first series.

C#

```
public double FirstSeriesMean { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.FirstSeriesVariance Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the variance of the first series.

C#

```
public double FirstSeriesVariance { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.ProbabilityTOneTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Probability T one tail value.

C#

```
public double ProbabilityTOneTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.ProbabilityTTwoTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Probability T two tail value.

C#

```
public double ProbabilityTTwoTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.SecondSeriesMean Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the mean of the second series.

C#

```
public double SecondSeriesMean { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.SecondSeriesVariance Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the variance of the second series.

C#

```
public double SecondSeriesVariance { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.TCriticalValueOneTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Critical T one tail value.

C#

```
public double TCriticalValueOneTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.TCriticalValueTwoTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Critical T two tail value.

C#

```
public double TCriticalValueTwoTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

TTestResult.TValue Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the T value.

C#

```
public double TValue { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

VerticalLineAnnotation Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Represents a vertical line annotation.

C#

```
public class VerticalLineAnnotation :
System.Windows.Forms.DataVisualization.Charting.LineAnnotation
```

Inheritance Object → ChartElement → ChartNamedElement → Annotation → LineAnnotation → VerticalLineAnnotation

Constructors

VerticalLineAnnotation()	Initializes a new instance of the VerticalLineAnnotation class.

Properties

Alignment	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
AllowAnchorMoving	Gets or sets a flag that specifies whether the end user is allowed to move the annotation anchor using a mouse. (Inherited from Annotation)
AllowMoving	Gets or sets a flag that specifies whether the end user is allowed to move an annotation using a mouse. (Inherited from Annotation)
AllowPathEditing	Gets or sets a flag that specifies whether the end user is allowed to move the points of a polygon annotation using a mouse. (Inherited from Annotation)
AllowResizing	Gets or sets a flag that specifies whether the end user is allowed to resize an annotation using a mouse. (Inherited from Annotation)

AllowSelecting	Gets or sets a flag that specifies whether the end user is allowed to select an annotation using a mouse. (Inherited from Annotation)
AllowTextEditing	Gets or sets a flag that specifies whether the text in an annotation may be edited when the end user double clicks the text. (Inherited from Annotation)
AnchorAlignment	Gets or sets the position alignment of an annotation to the anchor point. (Inherited from LineAnnotation)
AnchorDataPoint	Gets or sets the data point to which an annotation is anchored. (Inherited from Annotation)
AnchorDataPointName	Gets or sets the name of a data point to which an annotation is anchored. (Inherited from Annotation)
AnchorOffsetX	Gets or sets the offset from the anchor point for the X position of an annotation. (Inherited from Annotation)
AnchorOffsetY	Gets or sets the offset from the anchor point for the Y position of an annotation. (Inherited from Annotation)
AnchorX	Gets or sets the X coordinate to which the annotation is anchored. (Inherited from Annotation)
AnchorY	Gets or sets the Y coordinate to which the annotation is anchored. (Inherited from Annotation)
AnnotationGroup	Gets the group, if any, to which the annotation belongs. (Inherited from Annotation)
AnnotationType	Gets the annotation type.
AxisX	Gets or sets the X axis to which an annotation is attached. (Inherited from Annotation)
AxisXName	Gets or sets the name of the X axis to which an annotation is attached. (Inherited from Annotation)
AxisY	Gets or sets the Y axis to which an annotation is attached. (Inherited from Annotation)

AxisYName	Gets or sets the name of the Y axis which an annotation is attached to. (Inherited from Annotation)
BackColor	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
BackGradientStyle	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
BackHatchStyle	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
BackSecondaryColor	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
Bottom	Gets or sets the bottom boundary of an annotation position. (Inherited from Annotation)
ClipToChartArea	Gets or sets the name of the chart area to which an annotation is clipped. (Inherited from Annotation)
EndCap	Gets or sets a style for the anchor cap at the end of the annotation line. (Inherited from LineAnnotation)
Font	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
ForeColor	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
Height	Gets or sets the height of an annotation. (Inherited from Annotation)
IsInfinitive	Gets or sets a flag that indicates whether an infinite line should be drawn. (Inherited from LineAnnotation)
IsSelected	Gets or sets a flag that determines if an annotation is selected. (Inherited from Annotation)
IsSizeAlwaysRelative	Gets or sets a flag that specifies whether the size of an annotation is always defined in relative chart coordinates. (Inherited from LineAnnotation)
LineColor	Gets or sets the line color of an annotation. (Inherited from Annotation)
LineDashStyle	Gets or sets the line style of an annotation. (Inherited from Annotation)

LineWidth	Gets or sets the line width of an annotation. (Inherited from Annotation)
Name	Gets or sets the unique name of an annotation. (Inherited from Annotation)
Right	Gets or sets the right boundary position of an annotation. (Inherited from Annotation)
ShadowColor	Gets or sets the color of an annotation's shadow. (Inherited from Annotation)
ShadowOffset	Gets or sets the size of an annotation's shadow. (Inherited from Annotation)
SmartLabelStyle	Gets or sets the smart label style of an annotation. (Inherited from Annotation)
StartCap	Gets or sets a style for the anchor cap at the start of an annotation line. (Inherited from LineAnnotation)
Tag	Gets or sets an object associated with this chart element. (Inherited from ChartElement)
TextStyle	This property is not applicable to the line annotation type. (Inherited from LineAnnotation)
ToolTip	Gets or sets the tooltip text for an annotation. (Inherited from Annotation)
Visible	Gets or sets a flag that specifies whether an annotation is visible. (Inherited from Annotation)
Width	Gets or sets the width, in pixels, of an annotation. (Inherited from Annotation)
X	Gets or sets the X coordinate of an annotation. (Inherited from Annotation)
Y	Gets or sets the Y coordinate of the annotation. (Inherited from Annotation)
YAxisName	Gets or sets the name of the Y axis to which an annotation is attached. (Inherited from Annotation)

Methods

BeginPlacement()	Begins the end user placement of an annotation using the

	mouse. (Inherited from Annotation)
BringToFront()	Brings an annotation to the front of all annotations. (Inherited from Annotation)
Dispose()	Releases the resources used by the ChartElement. (Inherited from ChartElement)
Dispose(Boolean)	Releases unmanaged resources and, optionally, managed resources. (Inherited from Annotation)
EndPlacement()	Ends the end user placement of an annotation. (Inherited from Annotation)
Equals(Object)	Determines whether the specified Object is equal to the current ChartElement. (Inherited from ChartElement)
GetHashCode()	Returns a hash function for a particular type. (Inherited from ChartElement)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ResizeToContent()	Resizes an annotation according to its content size. (Inherited from Annotation)
SendToBack()	Sends an annotation to the back of all the annotations. (Inherited from Annotation)
SetAnchor(DataPoint)	Anchors an annotation to the specified data point. (Inherited from Annotation)
SetAnchor(DataPoint, Data Point)	Anchors an annotation to two data points. (Inherited from Annotation)
ToString()	Returns a string that represents the current Object. (Inherited from ChartElement)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

VerticalLineAnnotation Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the VerticalLineAnnotation class.

C#

```
public VerticalLineAnnotation ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

VerticalLineAnnotation.AnnotationType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the annotation type.

C#

```
[System.ComponentModel.Bindable(true)]
[System.ComponentModel.Browsable(false)]
public override string AnnotationType { get; }
```

Property Value

String

The annotation type.

Attributes BindableAttribute, BrowsableAttribute

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ViewEventArgs Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Provides event data that is used as a parameter object in the AxisViewChanged and AxisViewChanging events of the root Chart object.

C#

```
public class ViewEventArgs : EventArgs
```

Inheritance Object → EventArgs → ViewEventArgs

Remarks

This class is exposed as the `e` parameter in the AxisViewChanged and AxisViewChanging events. The AxisViewChanging event is raised just before a new view is displayed, as a result of the end-user clicking and dragging in a chart area. The AxisViewChanged event is raised just after the new view is created.

It is important to note that the position and size of a view cannot be set in the AxisViewChanged event because in this case the view is already created.

The ViewEventArgs class contains the following properties:

- The ChartArea property, which is used to get the ChartArea object in which the view is being displayed. The Name property gets the name of the chart area. Other chart area properties can also be set.
- The Axis property, which is used to get the Axis object with which the view is associated. To determine the type of axis, which can be X, Y, X2 or Y2, use the AxisName property. Other axis properties can also be set.
- The NewPosition property, which represents the new position of a view.
- The NewSize property, which represents the new size of a view.

- The NewSizeType property, which represents the unit of measurement for the size of a view.

Constructors

ViewEventArgs(Axis, Double)	Initializes a new instance of the ViewEventArgs class with the specified `axis` and `newPosition` parameters.
ViewEventArgs(Axis, Double, Double, DateTimeIntervalType)	Initializes a new instance of the ViewEventArgs class with the specified `axis`, `newPosition`, `newSize` and `newSizeType` parameters.

Properties

Axis	Gets the Axis object that a view belongs to.
ChartArea	Gets the ChartArea object that a view belongs to.
NewPosition	Gets the new position of a view along an axis.
NewSize	Gets or sets the size of a view along an axis.
NewSizeType	Gets the unit of measurement for the new size of a view.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ViewEventArgs Constructors

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ViewEventArgs class.

Overloads

ViewEventArgs(Axis, Double)	Initializes a new instance of the ViewEventArgs class with the specified `axis` and `newPosition` parameters.
ViewEventArgs(Axis, Double, Double, DateTimeIntervalType)	Initializes a new instance of the ViewEventArgs class with the specified `axis`, `newPosition`, `newSize` and `newSizeType` parameters.

ViewEventArgs(Axis, Double)

Initializes a new instance of the ViewEventArgs class with the specified `axis` and `newPosition` parameters.

C#

```
public ViewEventArgs
(System.Windows.Forms.DataVisualization.Charting.Axis axis, double
newPosition);
```

Parameters

`axis` Axis

The axis of the scale view.

`newPosition` Double

The start position of the new scale view.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ViewEventArgs(Axis, Double, Double, DateTimeIntervalType)

Initializes a new instance of the ViewEventArgs class with the specified `axis`, `newPosition`, `newSize` and `newSizeType` parameters.

C#

```
public ViewEventArgs
(System.Windows.Forms.DataVisualization.Charting.Axis axis, double
newPosition, double newSize,
System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
newSizeType);
```

Parameters

`axis` Axis

The axis of the scale view.

`newPosition` Double

The start position of the new scale view.

`newSize` Double

The new scale view size.

`newSizeType` DateTimeIntervalType

The unit of measurement for the size of the new scale view.

Applies to

▼ .NET Framework 4.8.1 and other versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ViewEventArgs.Axis Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Axis object that a view belongs to.

C#

```
public System.Windows.Forms.DataVisualization.Charting.Axis Axis { get; }
```

Property Value

Axis

An Axis object that represents the axis that a view belongs to.

Remarks

This property is exposed in the AxisViewChanged and AxisViewChanging events of the root Chart object.

This property can be used to:

- Determine the type of axis - X, X2, Y or Y2 - that the event is being raised for, using the AxisName property.
- Get and set axis properties, such as LineColor.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ViewEventArgs.ChartArea Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the ChartArea object that a view belongs to.

C#

```
public System.Windows.Forms.DataVisualization.Charting.ChartArea ChartArea {
get; }
```

Property Value

ChartArea

A ChartArea object that represents the chart area a view belongs to.

Remarks

This property is exposed in the AxisViewChanged and AxisViewChanging events of the root Chart object.

The ChartArea property can be used to determine the chart area that the view belongs to, by accessing the Name property. This property can also be used to get and set chart area properties.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ViewEventArgs.NewPosition Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the new position of a view along an axis.

C#

```
public double NewPosition { get; set; }
```

Property Value

Double

A `double` value that represents the position of a view along an axis, which is either the bottom of a vertical view or the left position of a horizontal view.

Remarks

This property is exposed in the AxisViewChanged and AxisViewChanging events of the root Chart object, and represents the new position of a view.

The NewPosition property can be set to a new value in the AxisViewChanging event, but cannot be set to a new value for the AxisViewChanged event because in this case the view is already created.

The NewPosition is always measured as an axis value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ViewEventArgs.NewSize Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets or sets the size of a view along an axis.

C#

```
public double NewSize { get; set; }
```

Property Value

Double

A `double` value that represents the size of a view along an axis. The size is either the height of the vertical view or the width of the horizontal view.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ViewEventArgs.NewSizeType Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the unit of measurement for the new size of a view.

C#

```
public System.Windows.Forms.DataVisualization.Charting.DateTimeIntervalType
NewSizeType { get; set; }
```

Property Value

DateTimeIntervalType

A DateTimeIntervalType enumeration value that represents the unit of measurement for the NewSize property, which represents the width or height of a data view. The default value is Auto.

Remarks

When this property is set to Auto, the unit of measurement for the NewSize property is automatically determined by the Chart control, and will be either a number or a day.

Recall that the NewSize property is always measured as an axis value. Make sure that if you explicitly set the size type, the type matches the axis that the scrollbar belongs to. In other words, do not specify Minutes if the axis does not use DateTime values.

Note that the NewSize property can be set to a new value in the AxisViewChanging event, but cannot be set to a new value for the AxisViewChanged event because the view is already created.

Applies to

Product	Versions

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult Class

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Stores the results of a Z Test statistical formula.

C#

```
public class ZTestResult
```

Inheritance Object → ZTestResult

Constructors

ZTestResult()	Initializes a new instance of the ZTestResult class.

Properties

FirstSeriesMean	Gets the mean of the first series.
FirstSeriesVariance	Gets the variance of the first series.
ProbabilityZOneTail	Gets the Probability Z one tail value.
ProbabilityZTwoTail	Gets the Probability Z two tail value.
SecondSeriesMean	Gets the mean of the second series.
SecondSeriesVariance	Gets the variance of the second series.
ZCriticalValueOneTail	Gets the Critical Z one tail value.
ZCriticalValueTwoTail	Gets the Critical Z two tail value.
ZValue	Gets the Z value.

Methods

Equals(Object)	Determines whether the specified object is equal to the current object. (Inherited from Object)
GetHashCode()	Serves as the default hash function. (Inherited from Object)
GetType()	Gets the Type of the current instance. (Inherited from Object)
MemberwiseClone()	Creates a shallow copy of the current Object. (Inherited from Object)
ToString()	Returns a string that represents the current object. (Inherited from Object)

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult Constructor

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Initializes a new instance of the ZTestResult class.

C#

```
public ZTestResult ();
```

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.FirstSeriesMean Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the mean of the first series.

C#

```
public double FirstSeriesMean { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.FirstSeriesVariance Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the variance of the first series.

C#

```
public double FirstSeriesVariance { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.ProbabilityZOneTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Probability Z one tail value.

C#

```
public double ProbabilityZOneTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.ProbabilityZTwoTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Probability Z two tail value.

C#

```
public double ProbabilityZTwoTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.SecondSeriesMean Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the mean of the second series.

C#

```
public double SecondSeriesMean { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.SecondSeriesVariance Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the variance of the second series.

C#

```
public double SecondSeriesVariance { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.ZCriticalValueOneTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Critical Z one tail value.

C#

```
public double ZCriticalValueOneTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.ZCriticalValueTwoTail Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Critical Z two tail value.

C#

```
public double ZCriticalValueTwoTail { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

ZTestResult.ZValue Property

Reference

Definition

Namespace: System.Windows.Forms.DataVisualization.Charting

Assembly: System.Windows.Forms.DataVisualization.dll

Gets the Z value.

C#

```
public double ZValue { get; }
```

Property Value

Double

A `double` value.

Applies to

Product	Versions
.NET Framework	4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1