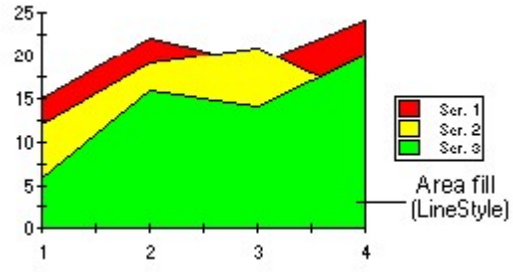
专用的 2D 图表

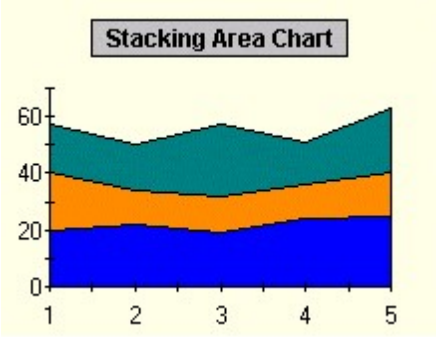
2D 图表可以以几种基本类型中的一种类型或者子类型来展示数据.子类型是对基本图表类型的扩展.例如 Bubble
图表.更加特殊化的图表类型,例如 Gantt
图表,也可以被创建.另外,很多的图表都含有根据属性来转变成为其他图表类型的能力,前提是两种图表类型所需要的数据是兼容的.例如,相同的数据可以在 XY-Plot 中展示,之后再做为一个条状图表来展示.
本章节介绍了 ClChart 中可用的 2D 图表类型.展示了如何选择专有的图表类型,然后展示了如何在一些图表类型中添加 3D 效果.

区域图表

一个区域图表以数据连接点的形式来绘制每一个序列,然后在点下面填充颜色.每一个序列被绘制在前面的序列的顶部.序列可以单独绘制或者叠加绘制.使用 LineStyle 属性,每一个序列的填充属性都可以被自定义.关于此的更多信息,请参见序列的线条和符号样式(240 页).



使用 ChartGroup 对象的 Stacked 属性来创建一个叠加的区域图表.叠加图表在前一个序列值的顶部叠加显示每一个序列的值.



在设计时设置图表类型为区域图表类型

- 在属性窗口中展开ChartGroups节点,通过点击ellipsos按钮来打开ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Area.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择Chart Properties. 在 Gallery 中选择图表类型为 Area.
- 另外一种可行的方法是在属性面板中选择图表属性,在Gallery中,选择图表类型为

Area.

区域图表编程时的讨论

下表描述了区域图表使用的数据数组.每一个数据序列代表将要被创建的X和Y数组值.给其它数组添加值将不会影响这个图表,但是给这些数组填充数据可能使得在转换图表到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

属性		描述			
X		持有 X 轴线的位置.			
Y		持有 Y 轴线的位置.			
	Y1		对区域图表没有影响.		
	Y2		对区域图表没有影响.		
	Y3		对区域图表没有影响.		

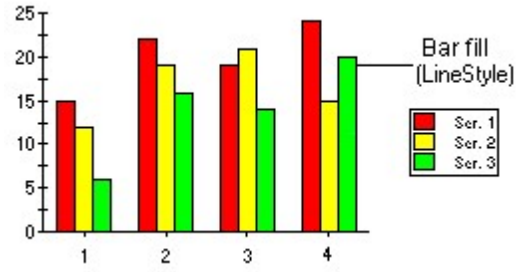
区域图表的 3D 效果

ClChart 的 3D
效果可以用在区域图表或者叠加区域图表上,来创建每一个序列在高度上的假象.通过使用深度,高地,旋转,和阴影效果属性.您可以增强您的区

域图表效果.使它们看起来更加的有视觉深度.
为了访问区域图表的 3D 视图.调整 View3D 对象的属性.View3D 对象是 PlotArea 对象的一个成员,然后 PlotArea 又是 ChartArea 对象的一个成员.通过调整 View3D 对象的属性,深度,高地, 旋转,阴影,您可以自定义 3D 视图.
请注意 Depth 属性是所有 3D 图表类型逻辑的关键. Elevation 和 Rotation 属性修改用户查看图表的方式.所以正是 Depth 属性实际上支配了一个图表是否是 3D 的.通过为 Depth 属性使用一个非 0 值,并且设置 Elevation 和 Rotation 属性的值为 0.您可以创建一个 3D 图表,虽然什么事情起来都没有改变.实际上您只是在看图表的正前方表面,就像一个标准的区域图表的视觉效果一样.
同时请注意,对一些数据进行 3D 视图的展示的效果可能是很令人满意的,但是同时其它的数据需要使用 2D 的面板.对于这些场景,调整附加在每一个 ChartGroup 上的 Use3D 属性.

条状图表

一个条状图表是一个倒置列图表,它的分类轴线是垂直轴线.一个条状/列图表在一个集群中以一个条状显示每一个序列.集群的数目是数据中点的数量.每一个集群显示在每一个序列中的第n个数据点.使用LineStyle属性,每一个序列的填充属性可以被自定义.关于此的更多信息,请参见序列的线条和符号样式(240 页).
下图展示了一个条状图表:



设计时设置图表类型为条状图表

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 bar.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties. 在 Gallery 中选择图表类型为 bar.
- 另外一种可行的方法是在属性面板中选择图表属性,在 Gallery 中,选择图表类型为

Bar.

条状图表编程时的讨论

下表描述了条状图表使用的数据数组.每一个数据序列代表将要被创建的X和Y数组值.给其它数组添加值将不会影响这个图表,但是给这些数组填充数据可能使得在转换图表到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

属性	描述
X	持有 X 轴线的位置.
Y	持有 Y 轴线的位置.
Y1	对条状图表没有影响.
Y2	对条状图表没有影响.
Y3	对条状图表没有影响.

浮动的条状图表

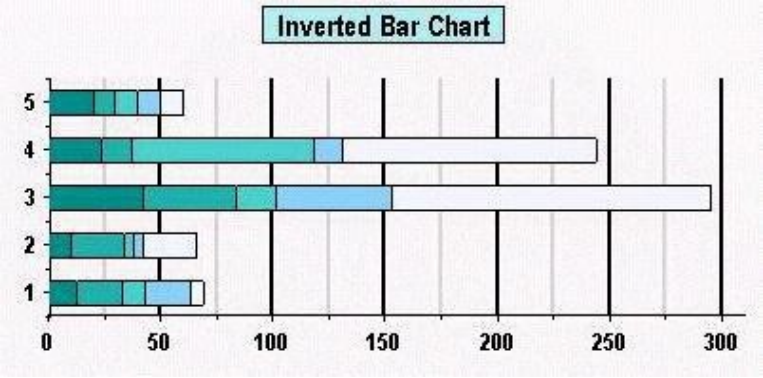
每一个叠加条状图表的序列可以被隐藏(序列将被作为一个空白集)或者移除(当序列不被考虑成是图表数据的一部分时).使用 ChartDataSeries 对象的 Display 属性来决定每一个数据序列是否显示,隐藏,还是移除.
在下面的例子中,第二个数据序列将首先被显示,然后被隐藏,最后被移除.

在设计时可以通过 SeriesList Collection Editor 来访问 Display 属性.该设计器可以通过打开 ChartGroupsCollection Editor 后访问.展开 ChartData 节点,然后点击紧挨着 SeriesList 属性的 ellipsis 按钮.

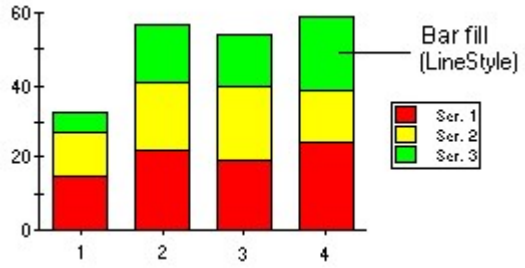
翻转的条状图表

一个条状图表是一个旋转的列图表,它的 X 和 Y 轴线被翻转了.

叠加条状图表



一个叠加条状图表将每一个序列作为叠加条状集群的一个部分来显示.集群的数目是数据中点的数目.每一个条状显示每一个数据序列中的第 n 个数据点. Cylinder, Pyramid,和 Cone Bar 图表可以被叠加,这是通过设置 Stacked 属性为 True 来完成的.使用 LineStyle 属性,每一个序列的
填充属性都可以被自定义.



示例代码
下面的代码创建了一个叠加的条状图表,首先在您的Visual Studio工程中添加C1Chart 控件,
然后添加如下的代码,程序运行后,就可以看到一个叠加条状图表的示例.

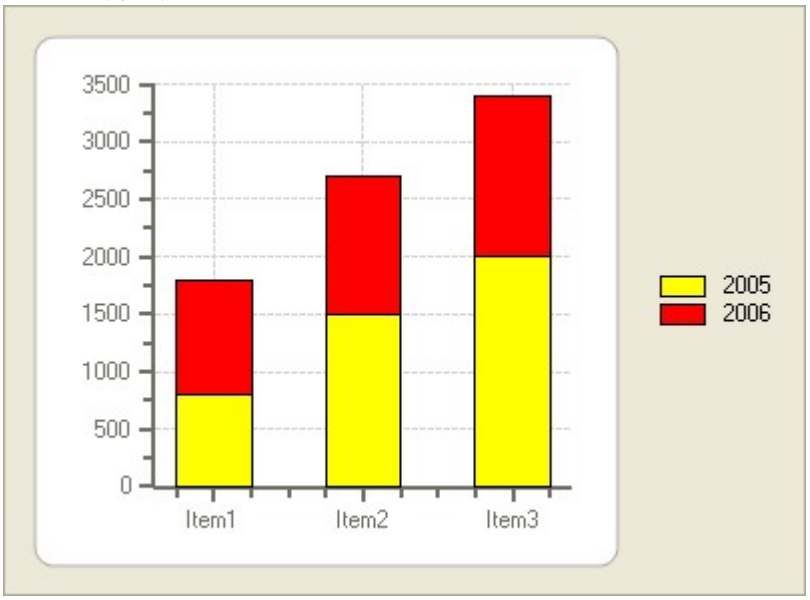
- Visual Basic

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
' Clear previous data
C1Chart1.ChartGroups(0).ChartData.SeriesList.Clear()
' Data
Dim items As String() = New String() {"Item1", "Item2", "Item3"}
Dim sales2005 As Integer() = New Integer() {800, 1500, 2000}
Dim sales2006 As Integer() = New Integer() {1000, 1200, 1400}
' Create first series
Dim ds2005 As C1.Win.C1Chart.ChartDataSeries =
C1Chart1.ChartGroups(0).ChartData.SeriesList.AddNewSeries() ds2005.Label = "2005"
ds2005.LineStyle.Color = Color.Yellow ds2005.X.CopyDataIn(items) ds2005.Y.CopyDataIn(sales2005)
' Create second series
Dim ds2006 As C1.Win.C1Chart.ChartDataSeries =
C1Chart1.ChartGroups(0).ChartData.SeriesList.AddNewSeries() ds2006.Label = "2006"
ds2006.LineStyle.Color = Color.Red ds2006.AutoEnumerate = True ds2006.Y.CopyDataIn(sales2006)
' Set chart type
C1Chart1.ChartGroups(0).ChartType = C1.Win.C1Chart.Chart2DTypeEnum.Bar
C1Chart1.ChartGroups(0).Stacked = True
' Set y-axis minimum
C1Chart1.ChartArea.AxisY.Min = 0
C1Chart1.Legend.Visible = True
' Remove the Axes caption
C1Chart1.ChartArea.AxisX.Text = ""
C1Chart1.ChartArea.AxisY.Text = ""
End Sub
```

- C#

```
private void Form1_Load(object sender, EventArgs e)
{
// Clear previous data

c1Chart1.ChartGroups[0].ChartData.SeriesList.Clear();


// Data

string[] items = new string[] { "Item1", "Item2", "Item3"}; int[] sales2005 = new int[] { 800, 1500, 2000}; int[]
sales2006 = new int[] { 1000, 1200, 1400};


// Create first series
C1.Win.C1Chart.ChartDataSeries ds2005 =

c1Chart1.ChartGroups[0].ChartData.SeriesList.AddNewSeries(); ds2005.Label = "2005";


ds2005.LineStyle.Color = Color.Yellow; ds2005.X.CopyDataIn( items); ds2005.Y.CopyDataIn( sales2005);
// Create second series
C1.Win.C1Chart.ChartDataSeries ds2006 =

c1Chart1.ChartGroups[0].ChartData.SeriesList.AddNewSeries(); ds2006.Label = "2006";


ds2006.LineStyle.Color = Color.Red; ds2006.AutoEnumerate = true; ds2006.Y.CopyDataIn( sales2006);
// Set chart type

c1Chart1.ChartGroups[0].ChartType = C1.Win.C1Chart.Chart2DTypeEnum.Bar; c1Chart1.ChartGroups[0].Stacked = true;


// Set y-axis minimum
c1Chart1.ChartArea.AxisY.Min = 0; c1Chart1.Legend.Visible = true; // Remove the Axes caption
c1Chart1.ChartArea.AxisX.Text = ""; c1Chart1.ChartArea.AxisY.Text = "";
}
```

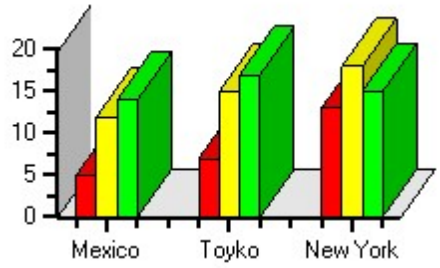
上面的示例代码会创建如下的图表:

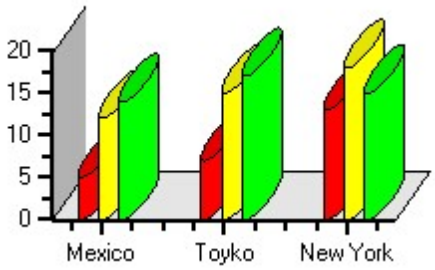


特殊的条形图表属性

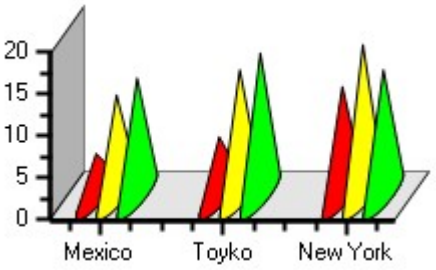
一个条形图表将序列作为一个集群中的条形来绘制.您可以在一个单独的行上以 2D
条形图表显示每一个序列,或者在多于一个的行上使用3D条状图来显示每一个序列.3D条状图表提供了一个很令人感兴趣的视图,您可以在 3D
条形图或者列图表的前面查看图表而不是常见的那一面的视图.条形图表和叠加条形图表中集群的大小和空间可以被自定义.另外,您可以改变条形图表的外观到以下的任何一种形状:圆筒,圆锥,或者金字塔形状.
外观
虽然条形图表一般情况下使用矩形条状图(默认方式)来展示,但是您同时可以使用圆筒,圆锥,
或者金字塔形状来表述图表,从而达到一个不同的效果.为了改变条状图表的形状,从默认形状到圆筒,圆锥,金字塔形.使用 Appearance 属性.
注意:下面的圆筒和圆锥条状图表看其来比圆环更加的椭圆,是因为图表的深度,为了让它们看起来更加的圆,您可以减少 Depth 属性.关于 3D 条状图表效果的更多信息,请参见条状图表 3D 效果(89 页).

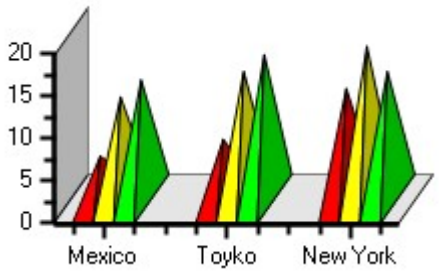
下图展示了 Appearance 属性值的效果:
Appearance = Default Appearance = Cylinder



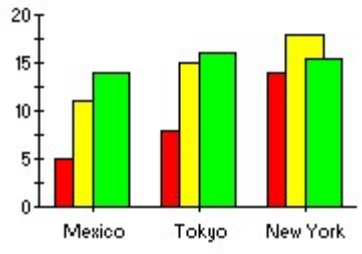


Appearance = Cone Appearance = Pyramid

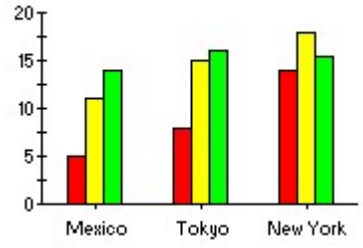


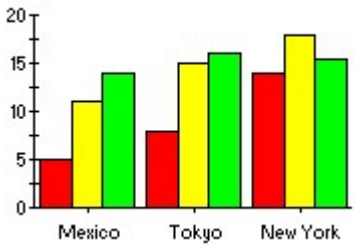


集群叠加使用 ClusterOverlap 属性来设置在一个集群中条形图叠加的数目.这个值代表了条形叠加的百分比,合法的取值范围为从 0 到 100.下图展示了一个 ClusterOverlap 属性为 50 百分比的条形图表.

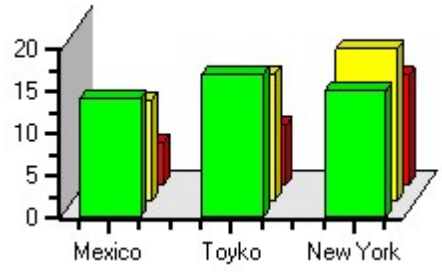


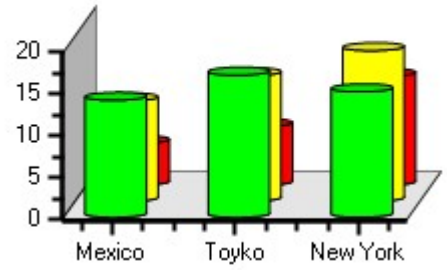
集群宽度
使用 ClusterWidth 属性来设置每一个条形集群使用的空间.这个值代表可用空间的百分比,合法的取值范围为从 0 到 100.
ClusterWidth = 50% ClusterWidth = 90%



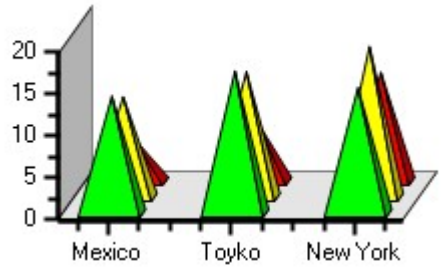


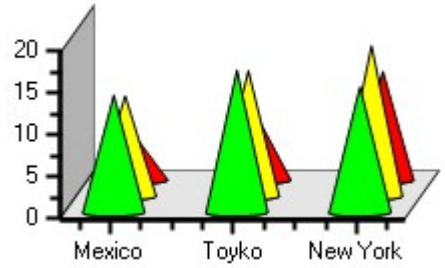
Bar 类的属性 ClusterWidth 和 ClusterOverlap 可以在设计时中通过 ChartGroupsCollection Editor 的 bar 节点来访问.
多行
当 Use3D 属性被设置为 True 时,会启用 3D 图表效果.您可以在 3D 图表中使用 MultiRow属
性来在集群中为每一个条形或者列显示一个新行.
下面展示了每一种条形图类型中 MultiRow 属性的效果.
MultiRow = True MultiRow = True



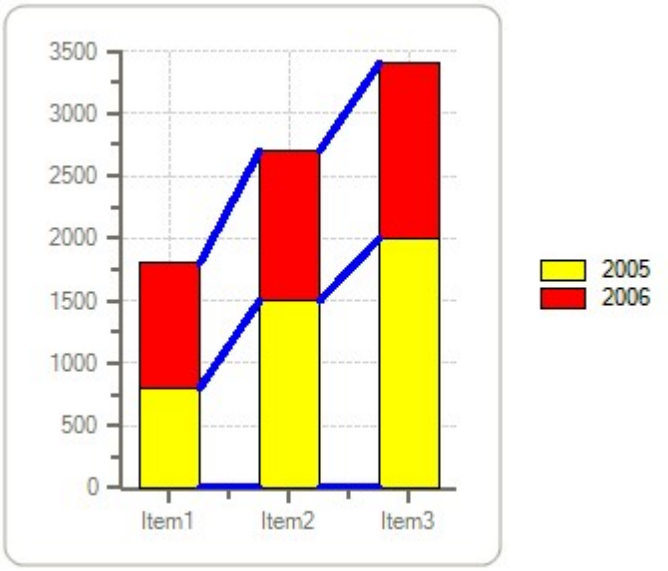


MultiRow = True MultiRow = True





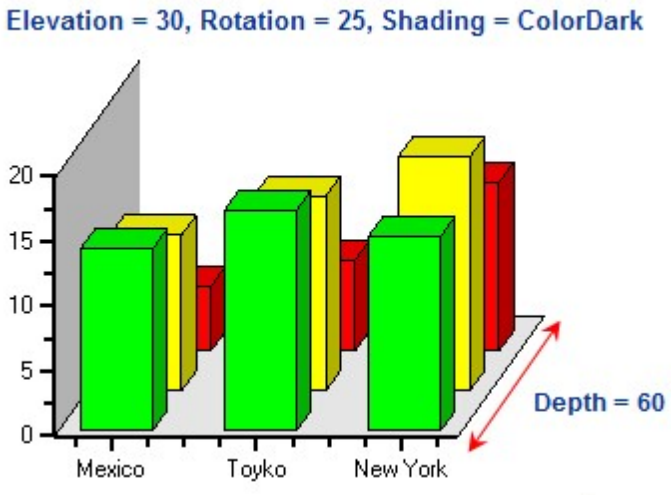
叠加 2D 条形图表和列图表的线条
需要设置BarLines属性为True,当您想要在叠加2D条状图和列图表的点和点之间的数据序列矩形中显示线条的话.当您设置 BarLines 属性为 True 时,您可以使用 BarLineColor 和
BarLineThickness 属性来为条形图线条指定线条的颜色和线条厚度.如下图所示:



通过在叠加 2D 条形图和列图表中使用条形线条,可以更加容易地参看到量化的信息.因为它使用条形线条来对比在
X-轴线(列图表)或者 Y 轴线(条形图)中的值.

条形图表 3D 效果

您可以在条形图和叠加条形图中使用 C1Chart 的 3D 效果来增加您的图表的外观.通过使用 depth, elevation, rotation, 和 shading 属性,您可以在给每一个数据序列创建深度效果.
当 Depth
属性被用来修改条形图表的深度时,它并不影响列的形状.但是,它影响圆筒,圆锥,和金字塔的形状.金字塔的宽度会在X轴线上的条形数目变化时自动改变.但是条形图的深度会受到
Depth 属性的影响.
下图展示了使用 3D 效果来让图表变得更加的在视觉上有吸引力:



为了访问条形图表的 3D 视图.调整 View3D 对象的属性.View3D 对象是 PlotArea 对象的一个成员,然后 PlotArea 又是 ChartArea 对象的一个成员.通过调整 View3D 对象的属性,深度,高地, 旋转,阴影,您可以自定义 3D 视图.
请注意 Depth 属性是所有 3D 图表类型逻辑的关键. Elevation 和 Rotation 属性修改用户查看图表的方式.所以正是 Depth

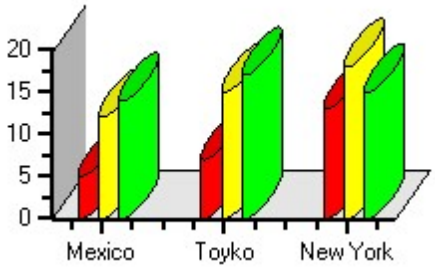
属性实际上支配了一个图表是否是 3D 的.通过为 Depth 属性使用一个非 0 值,并且设置 Elevation 和 Rotation 属性的值为 0.您可以创建一个 3D
图表,虽然什么事情起来都没有改变.实际上您只是在看图表的正前方表面,就像一个标准的区域图表的视觉效果一样.
同时请注意,对一些数据进行 3D 视图的展示的效果可能是很令人满意的,但是同时对于其它的数据也需要使用 2D
的面板.对于这些场景,调整附加在每一个 ChartGroup 上的 Use3D 属性.

条形图表的变种

条形图表有很多的变种,您可以使用圆筒,圆锥,和金字塔图表来显示数据序列.这些图表跟3D
条形图和列图表的功能一样:对比数据序列.唯一个差别在外观上,它们使用圆筒,圆锥或者金字塔来表示代表条形,而不是使用矩形.下面的主题提供了关于圆筒,圆锥和金字塔的更多信息.

圆筒图表

一个圆筒图表是条形和列图表的变形.它使用圆筒来代表条形或者列.圆筒图表在两个端点都创建相同的长圆形箱.像所有的条形和列图表一样,圆筒图表非常适合用来对比单独的项目或者组项目.



改变图表类型

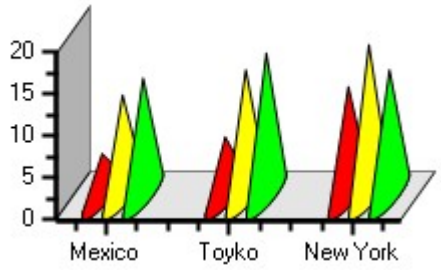
- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Bar,然后设置外观属性为 Cylinder.
- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties.

在 Gallery 中选择 Cylinder.

- 另外一种可行的方法是在 C1Chart 工具栏中选择 Cylinder.

圆锥图表

一个圆锥图表是 3D
条形和列图表的变形.它使用圆锥来代表条形或者列.圆锥图表本质上是一个旋转的三角形.它有一个扁圆形的底部和一个在更高点的弯曲侧.



在设计时其设置条形图表类型为圆锥

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Cone.

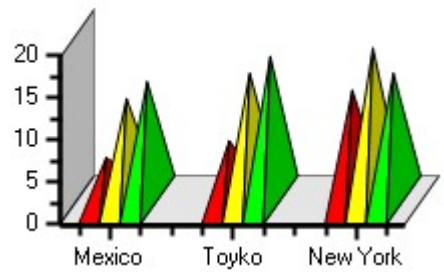
- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties.

在 Gallery 中选择图表类型为 Cone.

- 另外一种可行的方法是在 C1Chart 工具栏中选择 Cone.

金字塔图表

一个金字塔图表是 3D
条形和列图表的变形.它使用金字塔来代表条形或者列.金字塔图表非常类型与圆锥图表,处理他们的底部,金字塔图表经常用做地理学上的用途.



在设计时其设置图表类型为金字塔

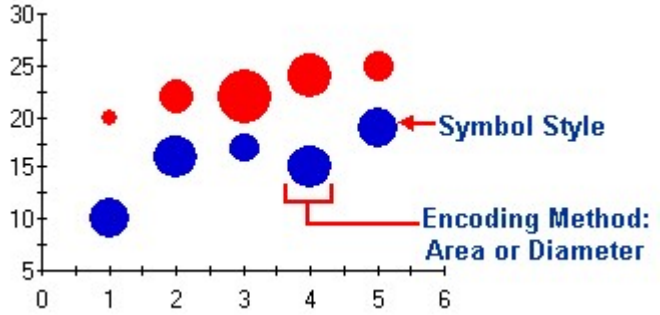
- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Bar,然后设置外观属性为 Pyramid.
- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties.

在 Gallery 中选择图表类型为 Pyramid.

- 另外一种可行的方法是在 C1Chart 工具栏中选择 Pyramid.

气泡图

一个气泡图由两个独立的值组成,这两个值提供了点的Y值和点的大小.气泡图通常通过改变大小来在每一个点上显示一个附加的数据值.Y 数组元素决定了笛卡尔位置(类似于在一个
XY-Plot 图表中),Y1 元素的值决定了每一个点上的气泡的大小.点的大小可以通过区域或者直径进行编码.使用 LineStyle 和 SymbolStyle 属性,符号样式和颜色,连接线的外观,都可以被自定义.
关于此的更多信息,请参见序列的线条和符号样式(240 页).



在设计时其设置图表类型为气泡图

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Bubble.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties.

在 Gallery 中选择图表类型为 XY-Plot,并且图表值类型为 Bubble..

- 另外一种可行的方法是在属性面板中选择图表属性,在 Gallery 中,选择图表类型为

XY-Plot,并且图表值类型为 Bubble.

气泡图表编程时的讨论

下表描述了气泡图表使用的数据数组.每一个数据序列代表将要被创建的 X,Y 和 Y1 数组值.
给其它数组添加值将不会影响这个图表,但是给这些数组填充数据可能使得在转换图表到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

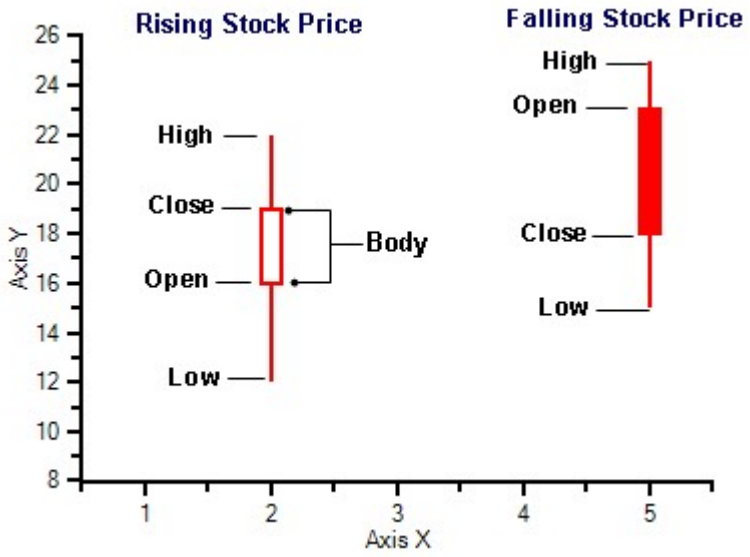
属性	描述
X	持有气泡相对于 X 轴线的位置.
Y	持有气泡相对于 Y 轴线的位置.
Y1	持有气泡的相对位置.
Y2	对条状图表没有影响.
Y3	对条状图表没有影响.

特殊的气泡图表属性

一个气泡图由两个序列组成,为了能够使用不同的点大小来绘制图表.气泡大小的编码方法, 同时它们的最大和最小大小都可以指定.
编码方法
使用 EncodingMethod 属性来设置气泡的大小是否取决于直径和区域.当指定了气泡的大小,
直径和区域都会被作为绘制区域整体的直径和整体区域的百分比来测量.反转这些值(使最小值大于最大值)会为小数值绘制大的气泡,而大数值绘制小的气泡.
EncodingMethod 属性是 Bubble 对象的一个属性,它可以通过 ChartGroupsCollection
Editor 的 Bubble 节点来访问.
最大和最小大小
气泡的最大和最小大小可以通过MaximumSize和MinimumSize属性来分别指定.Bubble对象的一些属性也可以通过 ChartGroups Collection
Editor 中的 Bubble 节点来访问.

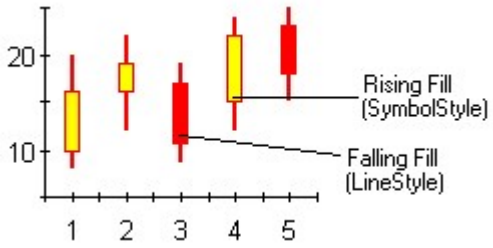
烛图

烛图,Hilo, Hilo 开闭是所有的股票图表,它们用在金融行业的应用中,来显示一个给定股票在开,闭,高,低时的价格.一个烛图是一个特殊的Hilo 开闭图表,它用来显式开和闭,高和低的关系.类似于 Hilo
开闭图表,烛图使用相同的价格数据(高,低,开,闭值),除了它们包含了一个较厚的类似于蜡烛的主体,并且使用烛体的颜色来提取开和闭值之间关系的附加信息.
例如,长透明的蜡烛表示了购买压力,长的有填充色的蜡烛代表了出售压力.
烛图由以下的元素组成:烛体,灯芯,尾巴.烛体或者主体(开闭值之间的实心条状)代表从开到闭之间的股票价格的变化.在烛体上下的细线,灯芯,尾巴描述了高/低的范围.一个空的烛体或者透明烛体指示了一个上升的股票价格(闭时的价格高于开时).在一个空的烛体中,烛体的底部代表了开时的价格,烛体的顶部代表了闭时的价格.一个填充过的烛体代表了一个下降股票价格(开时的价格高于闭时的价格).在一个填充过的烛体中,烛体的顶部代表开时的价格,烛体的底部代表了闭时的价格.
C1Chart 使用 Y 值作为烛体高值,Y1 作为烛体低值,Y2 作为烛体开时的值,Y3 作为烛体闭时的值.C1Chart 使用线的颜色来自动填充下降烛体.



使用 LineStyle 和 SymbolStyle 属性和 HiLoData 类,每一个序列的填充和线条属性可以被自定义

.



在设计时其设置图表类型为烛图

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Candle.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties. 在 Gallery 中选择图表类型为 Stock,并且图表子类型为 Candle.
- 另外一种可行的方法是在属性面板中选择图表属性, 在 Gallery 中选择图表类型为

Stock,并且图表子类型为 Candle.

烛图编程时的讨论

下表描述了烛图使用的数据数组.每一个数据序列代表将要被创建的 Y,Y1,Y2 和 Y3 数组值.

属性	描述
X	持有 X 轴线的位置.
Y	持有烛图的高值.
Y1	持有烛图的低值.
Y2	持有烛图的开值.
Y3	持有烛图的闭值.

特殊的烛图属性

当 ChartType 属性被设置为 Candle 时,您可以使用以下的属性来指定一个上升或者下降的
股票价格.

- FillFalling
- FillTransparent 展示下降价格

设置FillFalling属性为True. ChartDataSeries.LineStyle属性的值决定了填充时使用的颜色.
展示上升价格
设置 FillTransparent 为 True.这将会使得开烛图的烛体透明或者为空.如果您想为上升值指定一个特殊的颜色,那么将 FillTransparent
属性设置为 False,然后为 SymbolStyle
属性设置一个颜色.请注意推荐使用另外一种颜色,这样会让线条样式有一个颜色,而符号样式有另一种颜色.

甘特图

一个甘特图用来描述大量任务的日程表,并且指出工程完成过程中的关键活动.
甘特图跟 Bar 和 Hilo 图表有以下的相似之处:

- 类似于 Bar 图表,甘特图使用条形,但是它通常作为翻转和颠倒条形图来显示.
- 类似于 Hilo 图表,每一个序列中的 Y 和 Y1 数组中元素代表了高值和低值.甘特图使用 Y 和 Y1
  元素来代表一个任务的开始和完成时间.

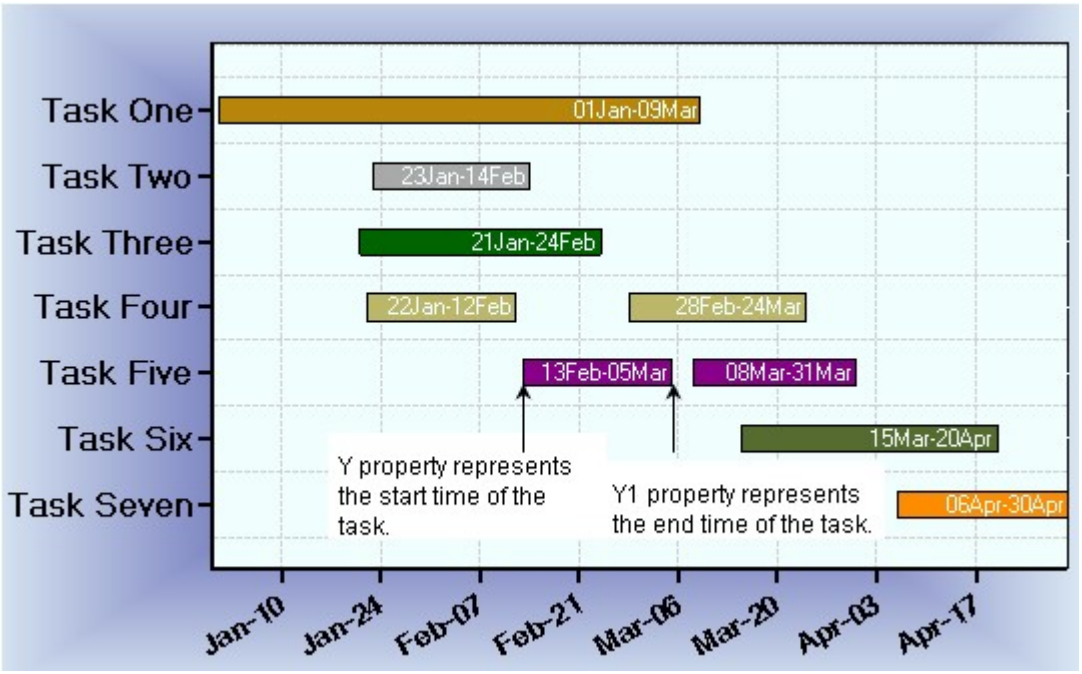
一个甘特图使用以下的方式来显示一个日程表:

- 活动/任务

活动/任务被显示在图表的左边,一个日程表被显示在图表的顶部或者底部.

- 任务有效期每一个工程的任务都被作为一个条形来显示,条形的开始处指定了活动或者任务的开始时间,条形的结束处指定了活动或者
  任务的结束或者完成时间.
- 紧急活动或者成就典型地,在甘特图中,特殊的图表或者颜色被用来代步在每一个任务的活动期间内的关键活动或者成就.

在一个甘特图中,您可以通过给图表数据序列添加一个新的序列来创建一个新的任务.在每一个序列中,都有一些属性可以用来在任务中给您的数据创建信息.例如,您可以使用 Y 属性来给您的任务的开始时间填充数据,并且可以使用 Y1 属性来记录您的任务的结束时间的数据.您可以从 ChartDataArray 对象的 DataType 属性中用在您的甘特图中的数据类型.
下面的甘特图展示了每一个任务的开始和结束时间.甘特图可以在一个任务中展示一个或者多个工程.在一些情况下,一个任务被指定了多个的子任务.每一个条状代表一个任务或者一个子任务.在一个轴线上包含一些条状可以跨越X和Y值的一个较大的范围.甘特图的数据在水平显示时会显得更加的吸引人,相比于垂直显示.所以,图表被倒置了,X 轴线是垂直的,而 Y 轴线是水平
的.C1Chart 提供了 ChartArea对象的Inverted 属性和 Axis的 Reversed
属性.请注意每一个水平条状被一个单色填充.通常地,这代表一个完成的任务.同时每一个水平图表含有一个指示了开始和技术时间的标签.这个标签可以在运行时或者设计时通过使用 ChartDataSeries 对象的 Lable 属性来添加到水平条形上.



在设计时其设置图表类型为甘特图

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Gantt.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties. 在 Gallery 中选择图表类型为 Gantt.
- 另外一种可行的方法是在属性面板中选择图表属性, 在 Gallery 中选择图表类型为

Candle.

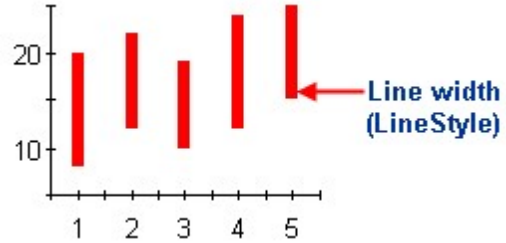
甘特图编程时的考虑

下表列出了在甘特图中的数据序列中的每一个元素和它们的效果.每一个数据序列都需要使用一个 X数组和两个含有 Y 和Y1 的 Y 数组.给其它数组添加值不会影响这个图表,但是给这些数组填充数据可能使得在转换甘特图到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

属性	描述
X	持有 X 轴线的位置.
Y	持有工程的开始时间的开始值.
Y1	持有工程的结束时间的结束值.
Y2	对甘特图无效.
Y3	对甘特图无效.

HiLo 图表

一个 HiLo 图表含有两个独立的值来支持在一个序列中的高值和低值数据.HiLo 图表主要用在金融应用中,来显示一个给定股票的高价格和低价格.每一个序列中的 Y 和 Y1 数组元素代表高值,和低值.使用 LineStyle,每一个序列的填充和线属性都可以被自定义,关于此的更多信息,请参见序列的线和符号样式(240).



在设计时其设置图表类型为 HiLo 图

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 HiLo.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties. 在 Gallery 中选择图表类型为 Stock,并且图表子类型为 Hi-low.
- 另外一种可行的方法是在属性面板中选择图表属性, 在 Gallery 中选择图表类型为

Stock,并且图表子类型为 Hi-low.

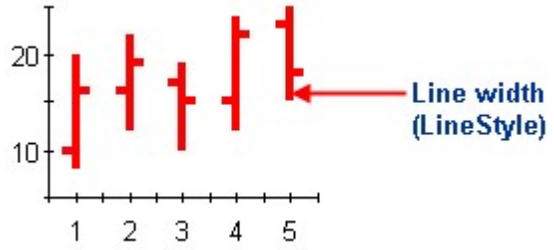
HiLo 图编程时的考虑

下表列出了在 HiLo 图中的数据序列中的每一个元素和它们的效果.每一个数据序列都需要使用一个 X数组和两个含有 Y 和 Y1 的 Y 数组.给其它数组添加值不会影响这个图表,但是给这些数组填充数据可能使得在转换 HiLo
图到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

属性	描述
X	持有 X 轴线的位置.
Y	持有图表的高值.
Y1	持有图表的低值.
Y2	对 HiLo 图无效.
Y3	对 HiLo 图无效.

HiLo 开闭图表

HiLo 开闭图表非常类似于 HiLo
图表,除了它组合四个独立的值来支持在一个序列中的一个点上的高,低,开,闭数据.除了显示一个股票的高值和低值之外,Y2 和 Y3 数组元素特别用来显示股票开盘和闭市时的价格.
使用 LineStyle,每一个序列的填充和线属性都可以被自定义,关于此的更多信息,请参见序列的线和符号样式(240).



在设计时其设置图表类型为 HiLo 开闭图

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为
HiLoOpenClose.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties. 在 Gallery 中选择图表类型为 Stock,并且图表子类型为 Hi-low-open-close.
- 另外一种可行的方法是在属性面板中选择图表属性, 在 Gallery 中选择图表类型为

Stock,并且图表子类型为 Hi-low-open-close.

HiLo 开闭图编程时的考虑

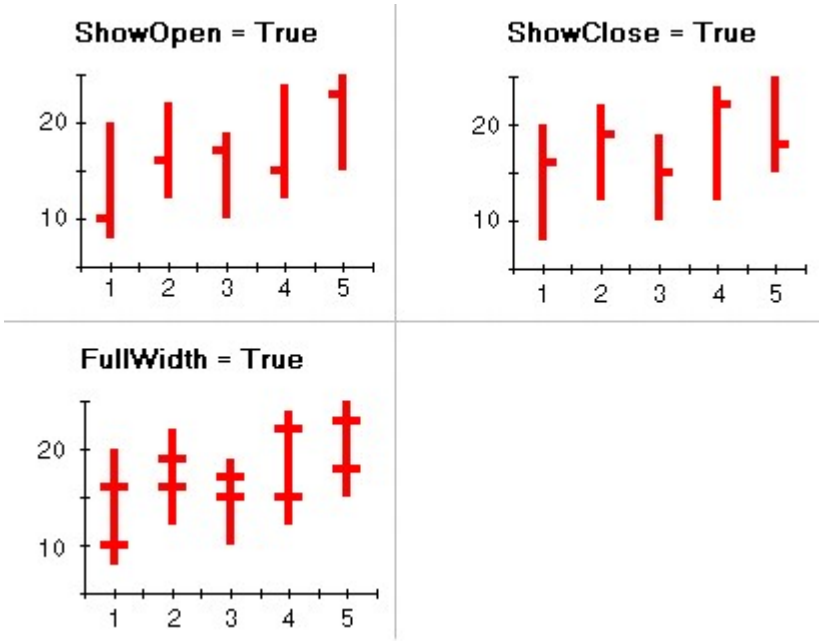
下表列出了 HiLo 开闭图使用的数据数组元素.每一个数据序列都需要一个含有 Y,Y1,Y2,Y3 值的 Y 数组值.

属性	描述		
X	持有 X 轴线的位置.		
Y	持有图表的高值.		
	Y1	持有图表的低值.	
	Y2	持有图表的开值.	
	Y3	持有图表的闭值.	

特殊的 HiLo 开闭图的属性

开发人员可以指定开闭刻度是如何显示在图表上的.
为了显示开闭记号

- 使用 ShowOpen, ShowClose 和 FullWidth 属性来指定开闭刻度是如何被显示的.
- 启用 ShowOpen 来显示开刻度,启用 ShowClose 来显示闭刻度.
- 启用 FullWidth 来在垂直范围的两端都显示开和闭刻度.



格式化线条和刻度

- 查看 HLCandle2005 示例

直方图

一个直方图持有一个未加工的数据值的集合,然后绘制频率分布.它经常被用在分组数据,这些数据由测量未加工的数据的集合而产生,并且绘制落在给定区间内的数据值的个数.请注意,未加工的值不会为直方图生成数据,但是它们被用来生成一个频率.它在显示时类似于条形图.非常重要的一点是直方图使用量化的变量.而同时条形图也经常使用量化的变量.
一个直方图能够过非常简洁地指出一个量化的变量在数据分布上的显著特征.量化变量的重要特征如下:

- 它展现典型的平均值.
- 数据产生一个通用的形状.数据值可以对称地分布在中间的两边.或者它们可以被斜交.
- 如果存在距离数据组比较远的值,那么将它们显示为离群值.
- 数据值可以远离或者靠近典型值.
- 分布可以造成单个的峰值或者多个峰值和谷地.

注意:间隔值代表数据元素的频率.有很多显示频率的技术.频率通常被显示为多边形,列,或者条形图.
在设计时其设置图表类型为直方图

- 在属性窗口中展开 ChartGroups 节点,在编辑器右边的面板上,设置 ChartType 属性为 Histogram.
- 另外一种改变图表类型的方法是展开.Net 属性窗口中的 Chart Properties,然后通过点击ellipsis按钮来打开ChartGroups Collection Editor. 在编辑器的右边面板上.

设置图表类型为 Histogram.

直方图编程时的考虑

不像其它的笛卡尔图表,一个直方图的输入不用指定 X 和 Y 坐标.相反地,一个单值的未加工数据的数组在每一个图表数据序列的 Y 图表数据数组中指定.根据 IntervalCreationMethod 的选择,直方图输入可以使用图表数据序列中的 X 图表数据数组来指定区间范围.
下表列出了直方图中使用的数据数组,并描述了每一个元素的效果.

属性	描述
X	和 XdataBoundaries 一起作为 IntervalCreationMethod 来持有直方图的间隔范围.

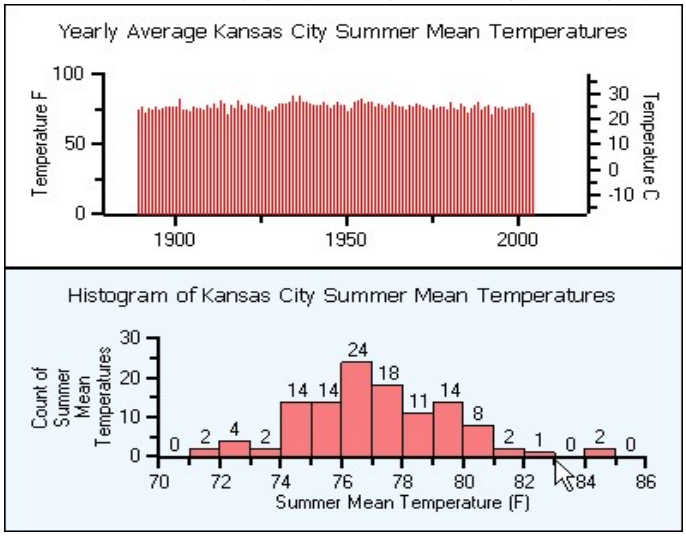
Y	持有未加工的数据值.
Y1	对直方图无效.
Y2	对直方图无效.
Y3	对直方图无效.

直方图的类型

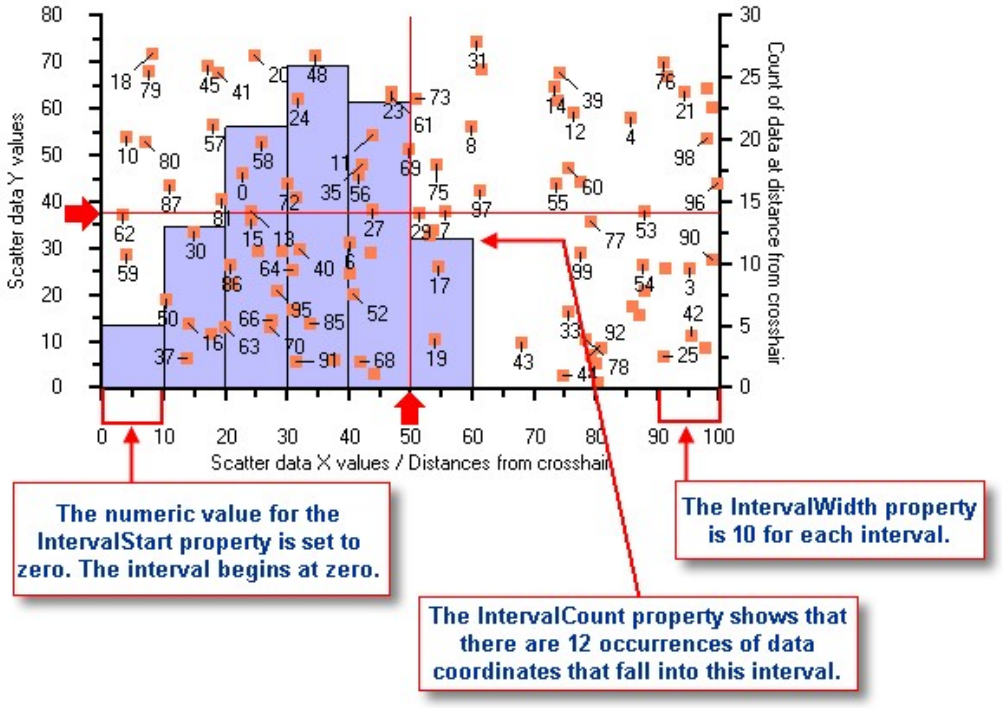
直方图含有集中图表类型,ClChart 提供了以下类型的直方图:

- 直方图
- 频率图
- 步进频率图

直方图
一个直方图的外观非常类似于一个条形图.它们在外观上的一个微小的差异在于,条形图中的矩形经常有一个空间上的间隔,但是直方图上的矩形不会被分割.直方图的列不是分离的,因为列边界代表了真实的X轴线的值.虽然直方图和条状图的外观非常相像,但是它们的功能是不同的.
一个条状图是由数据点创建的,但是直方图从数据的频率分布中创建.
下面的图表展示了一个直方图和一个条形图的区别.两个图表都使用相同的 Y 数据.条形图
(上面的图)显示了每一个年的平均气温.直方图(下面的图)使用相同的输入温度数据,并自动地列出了落在每一个间隔中的温度的数量,然后绘制结果直方图.为了一致性,每一个间隔的数量图表标签并添加在每一个间隔的顶部.



下图展示了一个直方图的详细和属性.它是一个展示一个图表组中含有一百万个随机数据坐标的散点图表,然后在第二个图表组中,展示了一个基于每一个散布的数据点到标记的交叉之间的距离的直方图.



在上面的直方图中, SemiAutomatic 方法被用在 IntervalCreationMethod
属性中来指定超过和低于的限制,还有间隔的数量.当您需要设置不同的间隔边界时 IntervalCreationMethod 属性是
很有用的.您可以选择在 IntervalCreationMethod 属性中选择以下三个方法中的一个:

- SemiAutomatic

当使用 SemiAutomatic 方法时,间隔的超过和低于限制和间隔的数量一起被指定.间隔边界被统一地计算.当您选择 SemiAutomatic 方法时
IntervalStart, IntervalWidth,和 IntervalNumber
属性是可用的. IntervalStart 属性用来获取或者设置第一个间隔开始处的数量值.在上面的直方图中,
IntervalStart属性被设置为zero.所以,间隔在值zero处开始.您可以使用IntervalWidth属性来指定间隔的宽度为一个指定的宽度大小.在上面
的示例中, IntervalWidth 属性为每一个间隔设置了 10 的大小.IntervalWidth
属性将会返回在一个直方图中间隔的数量.例如,在上面的直方图中,IntervalWidth 属性设置为 10,所以,最多可以含有 10
个间隔,剩余的四个间隔不会显示在上面的图表中,即使它们含有 zero 数据元素.

- Automatic

当使用 Automatic 方法时,图表使用 maximum 和 minimum 数据值来计算间隔的超过和低于
限制,并且限制间隔位于数据平均值中的 3 个标准偏差中.间隔的数目是可选的.间隔边界被统一地计算.

- XDataBoundaries

当使用XdataBoundaries方法时,数据序列的X值会被用来显式地设置每一个间隔边界.X值被排序过了,
所以重复值会被过滤掉.每一个结果的上升值被用来决定下一个间隔的边界.所以,第一个和第二个结果的
X值定义了第一个间隔,然后每一个连续的X值指定了下一个间隔的结束.请注意指定N个间隔需要N+1个唯一的X值.
为了在设计时访问 IntervalCreationMethod 属性,展开 ChartData 节点,并且点击紧挨着
SeriesList 节点的 ellipsis 按钮, ChartDataSeries Collection Editor 将会出现.定位在 Histogram节点上并展开它.选择IntervalCreat
ionMethod属性并选择从IntervalCreationMethod 属性的下拉列表框中选择三个方法中的一个. IntervalCreationMethod
属性可以通过以下的代码来访问:

- Visual Basic

```
cds.Histogram.IntervalCreationMethod = IntervalMethodEnum.SemiAutomatic
```

- C#

```
cds.Histogram.IntervalCreationMethod = IntervalMethodEnum.SemiAutomatic;
```

请注意,上面的"cds"变量,是一个 ChartDataSeries 对象的变量名称.在上面的示例代码中,使用了 SemiAutomatic 方法.您还可以通过指定
IntervalMethodEnum 枚举中的其它值来指定使用
Automatic 或者 XdataBoundaries 方法.
您可以通过使用 ChartHistogram 对象的 DisplayType 属性来一个一个直方图,频率图,步进
频率图的形式来显示间隔和数量.如下述代码所示:

- Visual Basic

```
'Displays the chart histogram as a Histogram ch.DisplayType = DisplayTypeEnum.Histogram 'Displays the chart histogram as a
frequency graph ch.DisplayType = DisplayTypeEnum.FrequencyGraph
'Displays the chart histogram as a stepped frequency graph ch.DisplayType = DisplayTypeEnum.SteppedFrequencyGraph
```

- C#

```
//Displays the chart histogram as a Histogram ch.DisplayType = DisplayTypeEnum.Histogram; //Displays the chart histogram as
```

```
a frequency graph ch.DisplayType = DisplayTypeEnum.FrequencyGraph;
//Displays the chart histogram as a stepped frequency graph ch.DisplayType = DisplayTypeEnum.SteppedFrequencyGraph;
```

注意:上面的"ch"变量,是一个 ChartHistogram 对象的变量名称
有些情况下一些数据元素出现在离其它数据元素较远的地方.在这些场景下,距离值被引用
到一个离群值上.在最常见的情况下,离群值不会落在一个间隔中,因为它离其它的数据距离较远.
所以,数据值将会丢失.
为了防止离群值被排除, C1Chart 的 ChartHistogram 对象提供了以下两个属性:

- AboveIntervalCount

AboveIntervalCount属性返回比最后一个间隔的结束值还大的值的数量.

- BelowIntervalCount

BelowIntervalCount属性返回比第一个间隔的开始值还要小的值的数量.
在设计器中您可以在 ChartDataSeries Collection Editor 中的 Histogram 节点中找到 AboveIntervalCount 和 BelowIntervalCount
属性.下面的代码展示了如何使用
AboveIntervalCount 来查看是否有一些离群值.

- Visual Basic

```
Dim overflow As Integer = _ CInt(cds.Histogram.AboveIntervalCount)
Dim msg As String = ""
' this tests to see if there are any outlier values that fall after the last interval.
If overflow > 0 Then
msg = "Number > " + carea.AxisX.Max.ToString() + " = " + overflow.ToString() End If
c1Chart1.ChartLabels("overflow").Text = msg
```

- C#

```
int overflow = (int)cds.Histogram.AboveIntervalCount; string msg = "";
// this tests to see if there are any outlier values that fall after the last interval. if(overflow > 0)
{
msg = "Number > " + carea.AxisX.Max.ToString() + " = " + overflow.ToString();
}

c1Chart1.ChartLabels["overflow"].Text = msg;
```

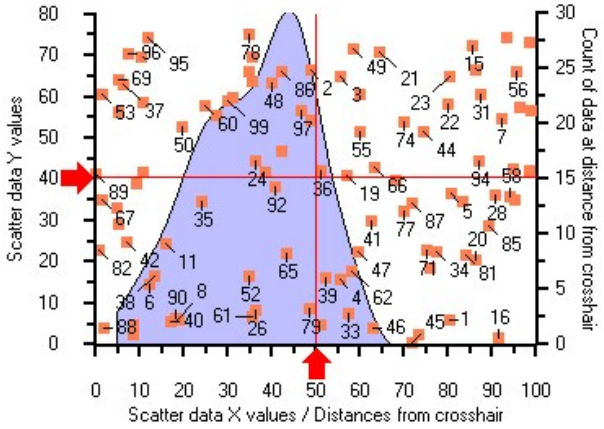
在直方图中,列或者间隔的区域是与其所代表的值是成比例的.当所有的间隔的间隔宽度都相同时,每一个间隔高度代表了在相同单位宽
度下的相同频率.在一些情况下列的宽度在大小上
差异很大.当这种情况发生时,列的高度必须被调整,以保持区域是成比例的.例如,如果您含有几个不统一的列,您可以使用 C1Chart 的
Histogram 对象的 Normalized 属性来使得列或者间隔与其代表的值是成比例的.
直方图对象含有三个属性: NormalDisplay, NormalizationInterval,和 Normalized.
Normalized属性可以被用来将不均匀的间隔变成标准化的间隔,以让每一个间隔高度代表在单位宽度下的相同频率.一个不均匀的间隔
是一个跟其它间隔不统一的间隔.当您设置 Normalized 属性为 True 时,您可以通过 NormalizationInterval
属性来灵活地设置宽度大小.当您想把
IntervalWidth从10设置为20时,将会造成每一个间隔的大小成倍增加.并且同时会造成每一个频率数目(y值)加倍.如果您想自定义您的
IntervalWidth 属性从 2 到 10,您可以将间隔的宽度缩小为原先的 1/5.这样同时会造成每一个频率数目(Y 值)变成原先的 1/5
大小.
频率图
一个频率图是一个没有列的直方图.每一个列的中间点使用一个直线或者曲线连接.所以,列被消除了.在 C1Chart
中,连接一个列的中间点和另一个列的中间点的线可以使线条或者曲线.在您对比多个数据序列时频率图非常有用,因为它让数据变得更
加的可读.
下面的频率图显示了所有数据坐标中从每一个散列的数据点到标记的交叉点之间的距离.散列点上附加的数字是点索引值.
您可以通过设置FitType属性为Line来指定在频率图中使用线条连接.同样地,曲线可以使用 Spline 类型的 FitType
值来指定.一个频率图的特点是它显示了分布的重要特性,并且没有分散在
角落上.下面的代码演示了将频率图的 FitType 属性设置为 Spline.

- Visual Basic

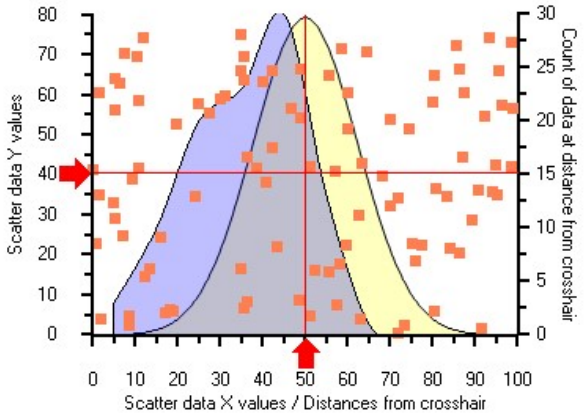
```
cds.FitType = FitTypeEnum.Spline
```

- C#

```
cds.FitType = FitTypeEnum.Spline;
```



下面的频率图跟上面的频率图是一致的,除了它含有一个正交的曲线来使用正交距离分布来对比距离分布.通过显示的正交曲线,我们对原生数据是否是正交分布的一个合理近似做出判断.
原生数据标签被隐藏了,以显示一个对两个曲线都清晰的图像.



图表中的Histogram对象提供了一个NormalDisplay属性来获取正交(高斯)曲线来显示.您可
以在设计时,通过展开 C1Chart 属性表格中 histogram 节点,并且设置 NormalDisplay 属性为 C1.Win.C1Chart.NormalCurve 来做到这一点.
另外一种可选的方式是通过以下的代码来访问
NormalDisplay 属性:

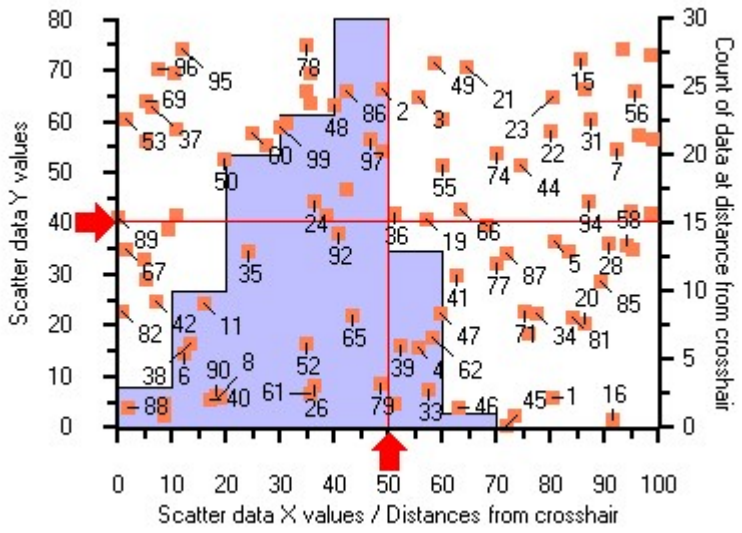
- Visual Basic

```
Dim nc As NormalCurve = cg.Histogram.NormalDisplay
```

- C#

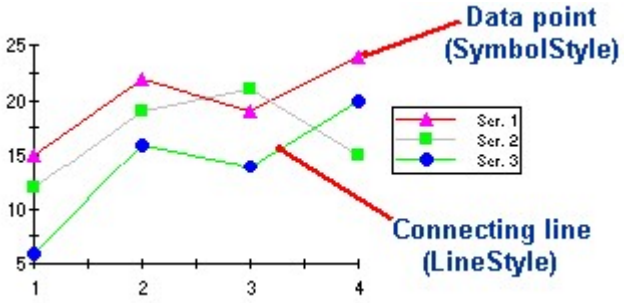
```
NormalCurve nc = cg.Histogram.NormalDisplay;
```

注意:nc,这个文字,是 NormalCurve 对象的名字.
步进频率图
一个步进频率图在功能上类似于一个直方图.这两个图表唯一的区别在外观上.在步进频率图中列之间的线条被消除了,但是在直方图中列之间的线还保持着.



线和 XY-Plot 图表

线和 XY-Plot
图表以数据的连接点方式来绘制每一个序列.通过自定义每一个序列的线和符号样式,连接线可以被消除,从而突出数据值本身,或者点可以被消除来突出点之间的关系.序列可以独立或者叠加绘制.每一个序列的线和符号属性同样可以被自定义.关于更多的信息,请参见序列的线和符号样式(240 页).



使用 ChartGroup 对象的 Stacked
属性来创建一个叠加的绘制图表.叠加图表通过在前一个序列的值的顶部来叠加显示每一个序列的值来显示数据.

在设计时其设置图表类型为线或 XY-Plot 图表类型

- 在属性窗口中展开ChartGroups节点,通过点击ellipsos按钮来打开ChartGroups Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为. XY-Plot 线图表选项在这种方式下是不可用的.
- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties.

在 Gallery 中选择图表类型为 Line 或者 XY-Plot.

- 另外一种可行的方法是在属性面板的底部中选择图表属性, 在 Gallery 中选择图表类型为 Line 或者 XY-Plot

绘制图表编程时的考虑

下表列出了在线和 XY-Plot
图中的数据序列中的每一个元素.每一个数据序列都需要使用一个X数组和Y数组.给其它数组添加值不会影响这个图表,但是给这些数组填充数

据可能使得在转换图表到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

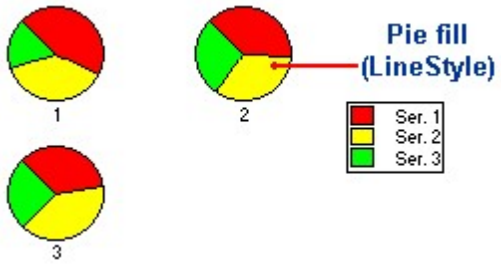
属性		描述			
X		持有 X 轴线的位置.			
Y		持有 Y 轴线的位置.			
Y1		对线和 XY-Plot 图表无效.			
	Y2		对线和 XY-Plot 图表无效.		
	Y3		对线和 XY-Plot 图表无效.		

XY-Plot 图表的 3D 效果

ClChart 的 3D 效果可以用在 XY-Plot 或者叠加 XY-Plot 上来创建每一个序列在高度上的假象.有些时候,这些图表被称为 Ribbon 图表.通过使用深度,海拔,旋转,和阴影属性,您可以增强您的区域图表的效果,通过创建视觉深度来突出它们.
为了访问一个 XY-Plot 图表的 3D 视图,调整 View3D 对象的一些属性 View3D 对象是 PlotArea对象的一个成员,然后PlotArea又是ChartArea对象的一个成员.通过调整View3D对象的属性,深度,高地,旋转,阴影,您可以自定义 3D 视图.
请注意 Depth 属性是所有 3D 图表类型逻辑的关键. Elevation 和 Rotation 属性修改用户查看图表的方式.所以正是 Depth 属性实际上支配了一个图表是否是 3D 的.通过为 Depth 属性使用一个非 0 值,并且设置 Elevation 和 Rotation 属性的值为 0.您可以创建一个 3D 图表,虽然什么事情起来都没有改变.实际上您只是在看图表的正前方表面,就像一个标准的区域图表的视觉效果一样.
同时请注意,对一些数据进行 3D 视图的展示的效果可能是很令人满意的,但是同时其它的数据需要使用 2D 的面板.对于这些场景,调整附加在每一个 ChartGroup 上的 Use3D 属性.

饼状和圆环图表

一个饼状图表以一个饼形中的一个切片的方式来绘制每一个序列.切片的数目就是数据中点的数目.每一个切片显示每一个序列中的第 N 个数据点.使用 LineStyle属性,每一个序列的填充属性可以被自定义.关于此的更多信息,请参见序列的线和符号样式(240 页).



在设计时其设置图表类型为饼状图表类型

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Pie.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties. 在 Gallery 中选择图表类型为 Pie.
- 另外一种可行的方法是在属性面板的底部中选择图表属性, 在 Gallery 中选择图表类型为 Pie

饼状图表编程时的考虑

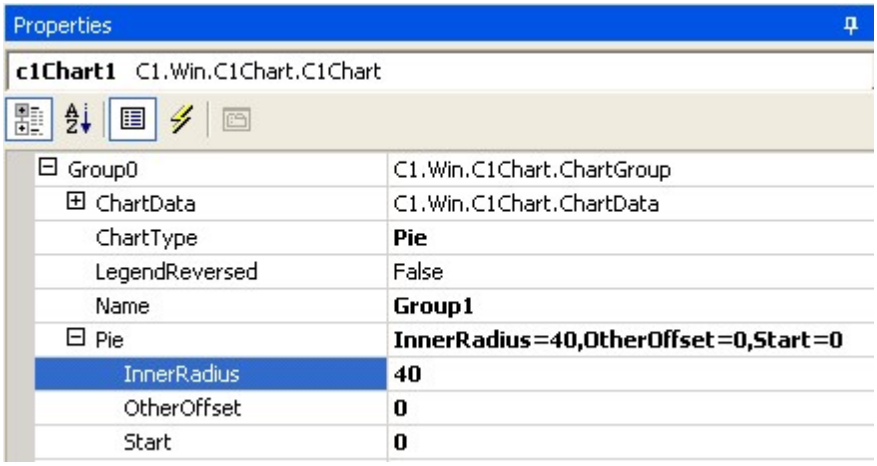
下表列出了在饼状图表中的数据序列中的每一个元素.每一个数据序列都需要使用一个 X 数组和Y数组.给其它数组添加值不会影响这个图表,但是给这些数组填充数据可能使得在转换图表到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

属性	描述
X	持有饼状图表中的以 0 开始的切片索引.
Y	持有由 X 值指定的切片的值.
Y1	对饼状图表无效.
Y2	对饼状图表无效.
Y3	对饼状图表无效.

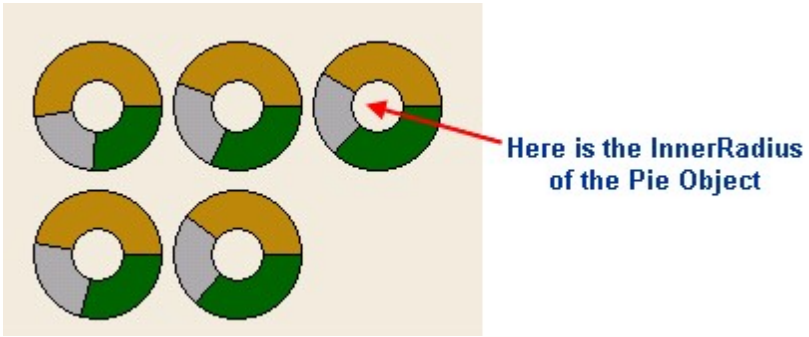
注意:在多个序列中输入数据会导致在一个图表组中显示多个图表切片.

圆环图表

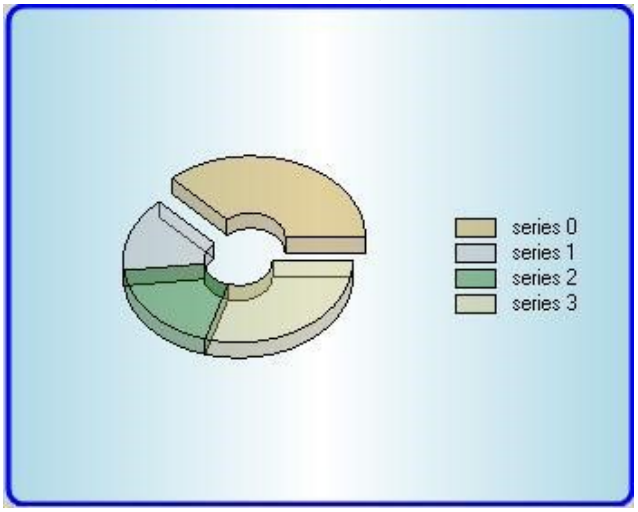
一个 ComponentOne 圆环图表带有非零半径的饼状图表,其在功能上跟一个饼状图表是一致的.但是它可以被用来增加视觉效果,特别是用来显式3D
效果时.同所有的饼状图表一样,每一个圆环以整体的一个片段的方式显式每一个序列上的数据点.如果指定了多个数据点,那么在图表中将会出现多个圆环.
可以通过设置一个饼状图表的 InnerRadius 属性值为非零来创建一个圆环图. InnerRadius
属性的值代表相对于整个饼状半径的百分比.可以在每一个图表组的Pie对象中访问InnerRadius 属性.下面显示了 InnerRadius
属性.在这个示例中 InnerRadius 属性被设置为 40%.



下面的图像展示了将 InnerRadius 属性设置为 40%是的饼状图表的效果.

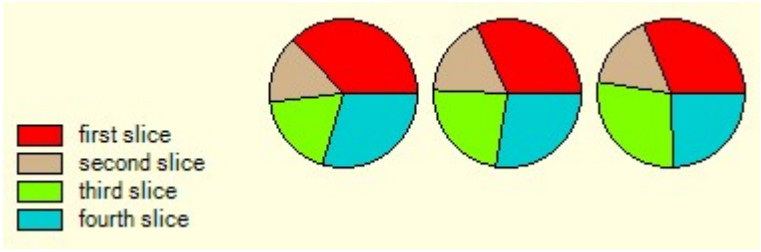


下面是另外的一个圆环图表的示例.在这个示例中 ChartDataSeries 对象的 OffSet 属性设置为 30.

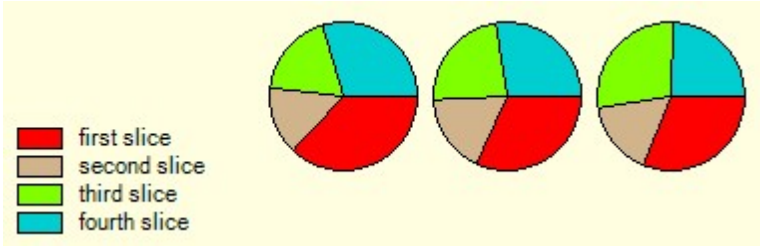


特殊的饼状图表属性

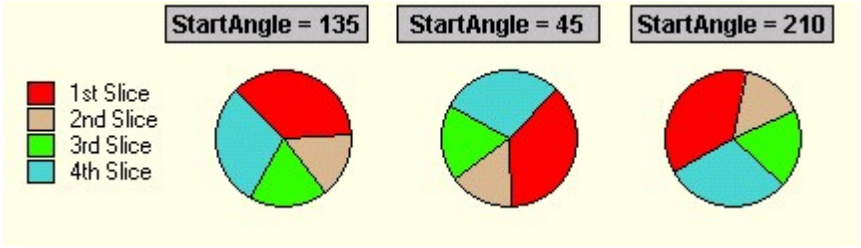
饼状图表与其它图表类型有很大的差异,因为它没有两维网格或者轴线的概念.改变饼状的直径或者分离图表的属性,可以通过 Pie
类的属性来完成.
顺时针或者逆时针方向
设置 Clockwise 属性为True 会以顺时针方向在一个饼状中绘制每一个序列,设置为False
会在饼状中以逆时针方向绘制每一个序列下图展示了两个饼状图表,一个以顺时针方向绘制,另一个以逆时针方向绘制.
顺时针方向(默认)



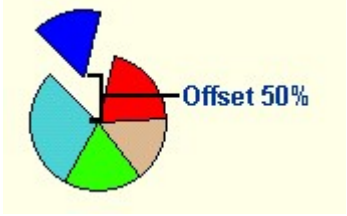
逆时针方向



开始角度 使用 Start 属性来指定第一个序列开始时切片的角度.默认的角度是 0
度.角度代表第一个切片的最左上边缘和饼状的右边水平半径之间的弧度,以反时针方向测量(见上图).Pie 类中的 Start
属性可以在设计时中在 ChartGroupsCollection Editor 中的 Pie 节点中访问.



分离饼状图饼状图表中的一个切片可以通过被分离来强调显示,分离的效果是该切片从饼状图中的其它
切片中伸出.使用序列的Offset属性来设置切片到饼状的中间位置的偏移.偏移以饼状的半径百分比的方式测量.



分离切片可以以编程的方式进行设置,但是仅能在序列上设置.

- Visual Basic

```
'Get the appropriate ChartData object.
Dim cd As ChartData =
C1Chart1.ChartGroups.ChartGroupsCollection(0).ChartData
'Sets the offset for the first series to 10% of the pie's radius cd.SeriesList(0).Offset = 10 'Resets the exploded
slices. cd.SeriesList(0).Offset = 0
```

- C#

```
//Get the appropriate ChartData object.

ChartData cd = c1Chart1.ChartGroups.ChartGroupsCollection[0].ChartData; //Sets the offset for the first series to 10% of the
pie's radius cd.SeriesList[0].Offset = 10; //Resets the exploded slices. cd.SeriesList[0].Offset = 0;
```

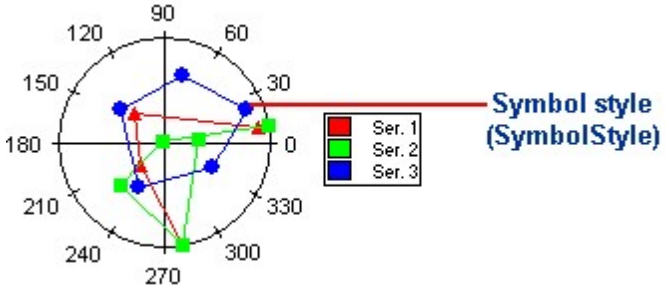
饼状图表的 3D 效果

C1Chart 的 3D 效果可以用在饼状图表上来创建每一个序列在高度上的假象.有些时候,这些图表被称为 Ribbon
图表.通过使用深度,海拔,旋转,和阴影属性,您可以增强您的区域图表的效果, 通过创建视觉深度来突出它们.
为了访问一个饼状图表的 3D 视图,调整 View3D 对象的一些属性 View3D 对象是 PlotArea
对象的一个成员,然后 PlotArea 又是 ChartArea 对象的一个成员.通过调整 View3D 对象的属性, 深度,高地,旋转,阴影,您可以自定义 3D
视图.
请注意 Depth 属性是所有 3D 图表类型逻辑的关键. Elevation 和 Rotation 属性修改用户查看图表的方式.所以正是 Depth
属性实际上支配了一个图表是否是 3D 的.通过为 Depth 属性使用一个非 0 值,并且设置 Elevation 和 Rotation 属性的值为

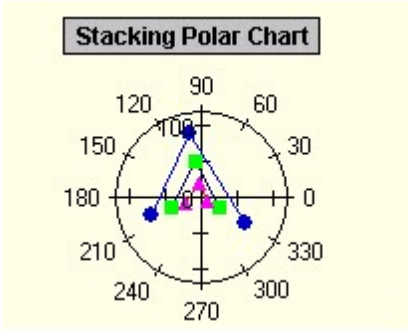
0.您可以创建一个 3D 图表,虽然什么事情起来都没有改变.实际上您只是在看图表的正前方表面,就像一个标准的区域图表的视觉效果一样.
同时请注意,对一些数据进行 3D 视图的展示的效果可能是很令人满意的,但是同时其它的数据需要使用 2D
的面板.对于这些场景,调整附加在每一个 ChartGroup 上的 Use3D 属性.

极地图表

一个极地图表以(theta,r)的形式绘制每一个序列中的 X 和 Y 坐标.其中,theta
是从开始的旋转,r是从开始的距离.theta可以通过角度(默认方式)或者弧度来指定.因为X轴线是一个圆,所以X
轴线的最大和最小值是固定的.序列可以独立或者叠加地显示.
使用 LineStyle 属性,每一个序列的填充属性可以被自定义.关于此的更多信息,请参见序列的线和符号样式(240 页).



使用 ChartGroup 对象的 Stacked
属性来创建一个叠加的极地图表.叠加图表通过在前一个序列的值的顶部来叠加显示每一个序列的值来显示数据.



在设计时其设置图表类型为极地图表类型

- 在属性窗口中展开ChartGroups节点,通过点击ellipsos按钮来打开ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Polar.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties. 在 Gallery 中选择图表类型为 Polar.
- 另外一种可行的方法是在属性面板的底部中选择图表属性, 在Gallery中选择图表类型为 Polar.

极地图表编程时的考虑

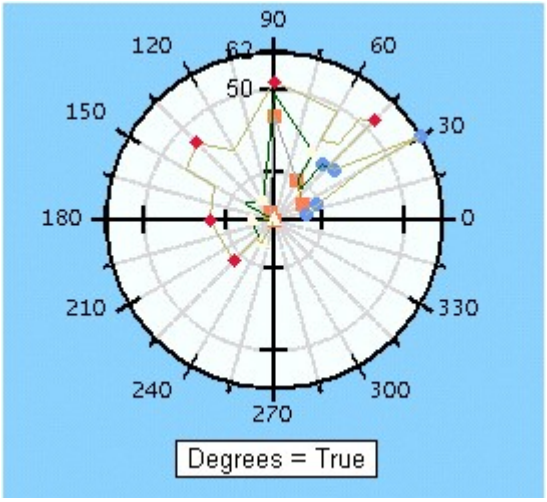
下表列出了在极地图表中的数据序列中的每一个元素.每一个数据序列都需要使用一个 X
数组和Y数组.给其它数组添加值不会影响这个图表,但是给这些数组填充数据可能使得在转换图表到其它图表类型时的工作变得简单,如果其它
图表类型使用这些数组的话.

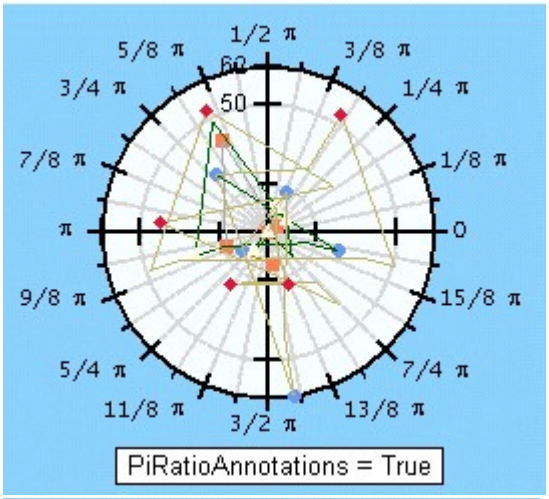
属性	描述
X	在角度或者弧度值中持有 X 轴线的位置.
Y	持有 Y 轴线的位置.
Y1	对极地图表无效.
Y2	对极地图表无效.
Y3	对极地图表无效.

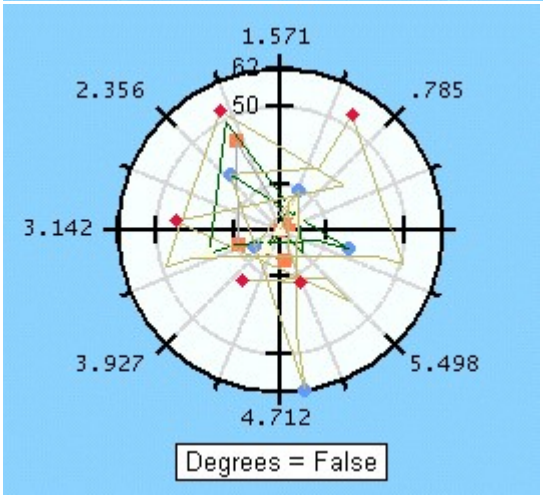
特殊的极地图表属性

在设计时通过 ChartGroupsCollection Editor 的 polar 节点可以访问下面的属性.
图表的角度或者弧度

Polar 类的 Degrees 属性用来设置极地图表的 X 数据值以角度(true)或者弧度(False)的方式反映角度. 在设计时通过 ChartGroupsCollection Editor 的 polar 节点可以访问到 Degrees 属性.
如果 Degrees 属性被设置为 False,那么图表将会反映弧度值.ClChart 提供了以圆周率的比例而不是弧度的方式来注解图表.设置 Polar 类的 PiRatioAnnotations 属性为 True 将会以圆周率的比例的方式来注解 X 值.





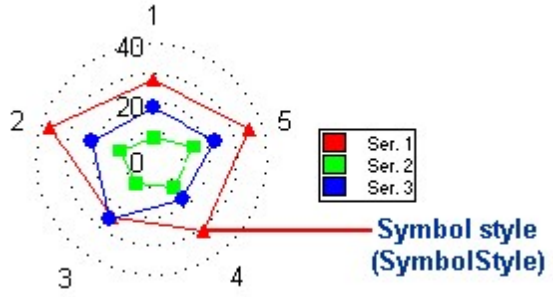


设置开始角度
使用 Polar 类的 Start 属性来指定极地图表的开始角度.默认的角度是 0 度.设置一个比 0 大的角度值,会以指定的量以逆时针方向移动图表的开始处.例如,设置 Start 属性值为 90 度,将会以逆时针方向移动极地图表 90 度.

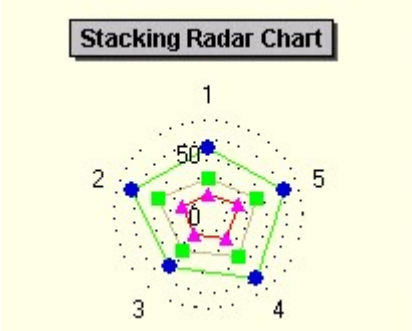
雷达图

一个雷达图以一个雷达警戒线(除了标签外 X 值被忽略)的方式绘制每一个数据中的 Y 值.如果数据含有 n 个唯一的点,那么图表平面会被分割成 n 个相同角度的分段,然后会在每一个 n/360 角度增量处绘制一个雷达警戒线(代表每一个点).默认地,代表第一个点的雷达会被垂直地绘制(在 90 度上).序列可以独立地或者叠加绘制.
雷达图可以在辐射方向上随着 X 轴主要网格线来显示刻度线,刻度线由 Y 轴线的刻度属性控

制.关于刻度线属性的更多信息,请参见轴线刻度线(207 页).
使用 LineStyle 属性,每一个序列的填充属性可以被自定义.关于此的更多信息,请参见序列的线和符号样式(240 页).



使用 ChartGroup 对象的 Stacked
属性来创建一个叠加的雷达图表.叠加图表通过在前一个序列的值的顶部来叠加显示每一个序列的值来显示数据.



在设计时其设置图表类型为雷达图表类型

- 在属性窗口中展开 ChartGroups 节点,通过点击 ellipsos 按钮来打开 ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Radar.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties. 在 Gallery 中选择图表类型为 Radar.
- 另外一种可行的方法是在属性面板的底部中选择图表属性, 在 Gallery 中选择图表类型为 Radar.

雷达图表编程时的考虑

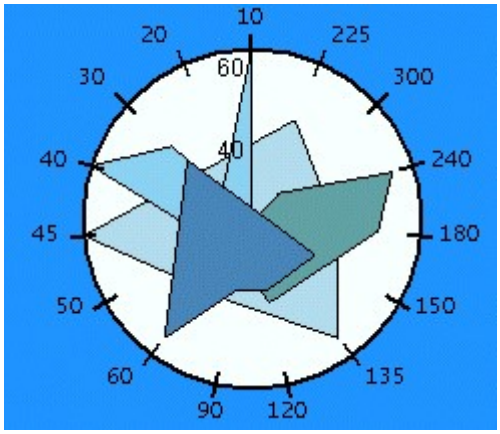
下表列出了在雷达图表中的数据序列中的每一个元素.每一个数据序列仅使用一个 Y 数组.
给其它数组添加值不会影响这个图表,但是给这些数组填充数据可能使得在转换图表到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

属性	描述
X	除了标签外都被忽略
Y	持有雷达图的 Y 轴线数据,Y 数据数组中点的数目决定了图表中雷达警戒线的数目.
Y1	对雷达图表无效.
Y2	对雷达图表无效.
Y3	对雷达图表无效.

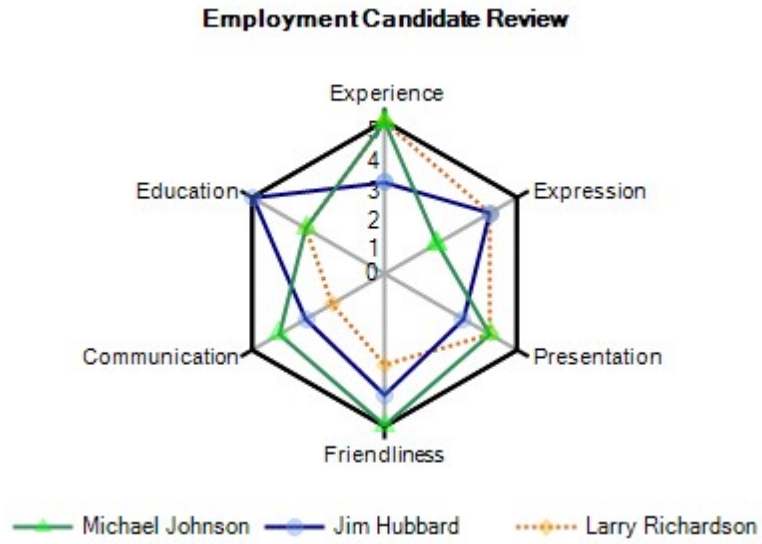
特殊的雷达图属性

雷达图含有一些特殊的属性,这些属性可以用来设置雷达的角度,设置起始角度,创建填充的雷达图,和决定是否在雷达图中使用平面 Y 坐标网格线.
图表的角度或者弧度
Radar类的Degrees属性用来设置雷达图表的X数据值以角度(true)或者弧度(False)的方式反映角度. 在设计时通过 ChartGroupsCollection Editor 的 Radar 节点可以访问到 Degrees 属性.
设置开始角度
使用 Radar 类的 Start 属性来指定雷达图表的开始角度.默认的角度是 0 度.设置一个比 0 大的角度值,会以指定的量以逆时针方向移动图表的开始处.例如,设置 Start 属性值为 90 度,将会以逆时针方向移动雷达图表 90 度.
填充的雷达图
一个填充的雷达图以一个雷达警戒线(除了标签外X值被忽略)的方式绘制每一个数据中的Y 值.如果数据含有 n 个唯一的点,那么图表平面会被分割成 n 个相同角度的分段,然后会在每一个 n/360角度增量处绘制一个雷达警戒线(代表每一个点).每一个序列被绘制在前一个序列的"顶部",

序列可以独立地或者叠加绘制.填充的雷达图和雷达图是类似的,除了在开始出和点之间的空间是被填充的,并且符号不会被显示.为了创建一个填充的雷达图,设置 Radar 类的 Filled 属性为
True.



平面网格线您可以通过设置 FlatGridLines 属性为 True 来在雷达图中显示平面的 Y 坐标网格线.



下面的代码展示了以编程的方式设置该属性的值.

- Visual Basic

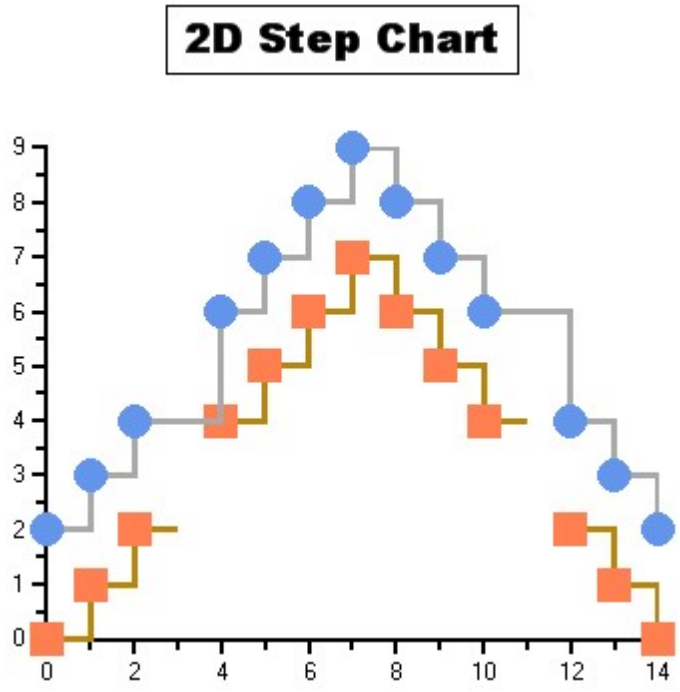
```
C1Chart1.ChartGroups(0).Radar.FlatGridLines = True
```

- C#

```
c1Chart1.ChartGroups[0].Radar.FlatGridLines = true;
```

步进图表

一个步进图表是 XY-Plot 图表的另一种形式.步进图表经常在 Y 值以离散量改变的场景下被使用,在一些 X 上值会有一些突然的改变的时候.一个简单,日常的例子是绘制一个带有时间的支票薄收支平衡的图表.当每一次储蓄发生,每一次使用支票,支票登记薄的平衡(Y 值)会随着时间(X 值)流逝而突然的改变,而非逐步的改变.在一些没有储蓄发生或者使用支票的时期,平衡(Y 值)随着时间流逝依然保持不变.
类似于线和XY绘制,步进图表的外观可以被自定义,通过为每一个序列使用线和符号样式,来改变颜色,符号大小,和线的厚度.符号可以被完全的消除从而增强突出点之间的关系,或者可以包含符号来指出真实的数据值.如果出现数据洞,步进图表表现的像预期一样,在数据洞的 X 值上使用序列线来表明已知的信息. 符号和线会在非空洞的数据再次出现时继续显示.
像大多数的 XY 样式绘制图表一样,步进图表可以在恰当的时机叠加显示.
下面的图表展示了 2D 步进图表:



在设计时其设置图表类型为步进图表类型

- 在属性窗口中展开ChartGroups节点,通过点击ellipsos按钮来打开ChartGroups

Collection Editor.在编辑器右边的面板上,设置 ChartType 属性为 Step.

- 另外一种改变图表类型的方法是右键点击既存的图表,并且选择 Chart Properties.

在 Gallery 中选择图表类型为 Step.

- 另外一种可行的方法是在属性面板中选择图表属性, 在 Gallery 中选择图表类型为 Step.

步进图表编程时的考虑

下表列出了在步进图表中的数据序列中的每一个元素.每一个数据序列仅使用一个 Y 数组.
给其它数组添加值不会影响这个图表,但是给这些数组填充数据可能使得在转换图表到其它图表类型时的工作变得简单,如果其它图表类型使用这些数组的话.

属性	描述
X	持有 X 轴线的位置.
Y	持有 Y 轴线的位置.
Y1	对步进图表无效.
Y2	对步进图表无效.
Y3	对步进图表无效.

步进图表的 3D 效果

ClChart 的 3D 效果可以用在步进图表上来创建每一个序列在高度上的假象.通过使用深度,
海拔,旋转,和阴影属性,您可以增强您的区域图表的效果,通过创建视觉深度来突出它们.
为了访问一个区域图表的 3D 视图,调整 View3D 对象的一些属性 View3D 对象是 PlotArea 对象的一个成员,然后 PlotArea 又是 ChartArea 对象的一个成员.通过调整 View3D 对象的属性, 深度,高地,旋转,阴影,您可以自定义 3D 视图.
请注意 Depth 属性是所有 3D 图表类型逻辑的关键. Elevation 和 Rotation 属性修改用户查看图表的方式.所以正是 Depth 属性实际上支配了一个图表是否是 3D 的.通过为 Depth 属性使用一个非 0 值,并且设置 Elevation 和 Rotation 属性的值为 0.您可以创建一个 3D 图表,虽然什么事情起来都没有改变.实际上您只是在看图表的正前方表面,就像一个标准的区域图表的视觉效果一样.
同时请注意,对一些数据进行 3D 视图的展示的效果可能是很令人满意的,但是同时其它的数据需要使用 2D 的面板.对于这些场景,调整附加在每一个 ChartGroup 上的 Use3D 属性.