Chart2D XAML 快速参考

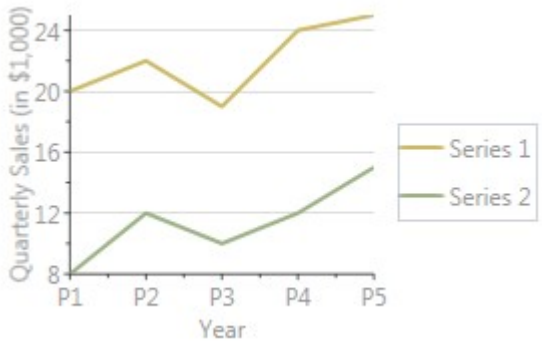
以下XAML演示如何选择图表类型，选择调色板，设置坐标轴以及为一个折线图添加数据系列：

XAML

```
<c1:C1Chart x:Name="_chart" Palette="Module" ChartType="Line"

Foreground="#a0000000" Background="#e0ffffff" >
<c1:C1Chart.View>
<c1:ChartView>
<c1:ChartView.AxisX>
<c1:Axis Title="Year" MajorGridStroke="Transparent" />
</c1:ChartView.AxisX>
<c1:ChartView.AxisY>
<c1:Axis Title="Quarterly Sales (in $1,000)"
MajorGridStroke="#40000000" AnnoFormat="n0" />
</c1:ChartView.AxisY>
</c1:ChartView>
</c1:C1Chart.View>
<c1:C1ChartLegend Position="Right" />
<c1:C1Chart.Data>
<c1:ChartData ItemNames="P1 P2 P3 P4 P5">
<c1:DataSeries Label="s1" Values="20, 22, 19, 24, 25"
ConnectionStrokeThickness="6" />
<c1:DataSeries Label="s2" Values="8, 12, 10, 12, 15" />
</c1:ChartData>
</c1:C1Chart.Data>
</c1:C1Chart>
```

设置基本的折线图形

以下XAML展示如何声明C1Chart控件，设置ChartType，Theme，以及Palette属性以定义基本的图表外观XAML可以被添加在标签内下图表由下面的XAML代码生成：



XAML

```
<Window x:Class="WpfApplication1.Window1" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" xmlns:c1="http://schemas.componentone.com/xaml/c1chart"
Title="Window1" Height="300" Width="348" >
<Grid>
<!-声明C1Chart控件 设置，Theme，以及Palette 属性以定义基本的图表外观->
<c1:C1Chart
Name="C1Chart1"
ChartType="Line"
Foreground="#a0000000"
Background="#e0ffffff"
Theme ="Vista"

Palette ="Aspect" >
<!-定义图表View，包括图表坐标轴。->
<c1:C1Chart.View>
<c1:ChartView>
<!-定义X-轴（标题，网格）->
<c1:ChartView.AxisX>
<c1:Axis
Title="Year"
MajorGridStroke="Transparent"/>
</c1:ChartView.AxisX>
<!-定义Y-轴（标题，网格，标注格式）->
<c1:ChartView.AxisY>
```

```
<c1:Axis
Title="Quarterly Sales (in $1,000)"
MajorGridStroke="#40000000"
AnnoFormat="n0" />
</c1:ChartView.AxisY>
</c1:ChartView>
</c1:C1Chart.View>
<!-定义图表数据，包含数据系列。->
<c1:C1Chart.Data>
<c1:ChartData>
<!-- ItemNames 定义沿着X-轴方向的标签-->
<c1:ChartData.ItemNames>P1 P2 P3 P4 P5</c1:ChartData.ItemNames>
<!-每一个DataSeries指定一个标签（显示在图例上），以及系列数据->
<c1:DataSeries Label="Series 1" RenderMode="Default" Values="20 22 19 24 25" />
<c1:DataSeries Label="Series 2" RenderMode="Default" Values="8 12 10 12 15"
/>
</c1:ChartData>
</c1:C1Chart.Data>
<!-添加一个 ChartLegend，停靠到图表的右侧，以便显示一个包含数据系列以及其样式的图例区。->
<c1:C1ChartLegend DockPanel.Dock="Right" />
</c1:C1Chart>
</Grid>
</Window>
```

设置一个甘特图表

为了创建一个甘特图表，请使用以下XAML标记代码：

XAML

```
<clchart:C1Chart Margin="0" Name="c1Chart1"
xmlns:sys="clr-namespace:System;assembly=mscorlib">
<clchart:C1Chart.Resources>
<x:Array x:Key="start" Type="sys:DateTime" >
<sys:DateTime>2008-6-1</sys:DateTime>
<sys:DateTime>2008-6-4</sys:DateTime>
<sys:DateTime>2008-6-2</sys:DateTime>
</x:Array>
<x:Array x:Key="end" Type="sys:DateTime">
<sys:DateTime>2008-6-10</sys:DateTime>
<sys:DateTime>2008-6-12</sys:DateTime>
<sys:DateTime>2008-6-15</sys:DateTime>
</x:Array>
</clchart:C1Chart.Resources>
<clchart:C1Chart.Data>
<clchart:ChartData>
<clchart:ChartData.Renderer>
<clchart:Renderer2D Inverted="True" ColorScheme="Point"/>
</clchart:ChartData.Renderer>
<clchart:ChartData.ItemNames>Task1 Task2 Task3</clchart:ChartData.ItemNames>
<clchart:HighLowSeries HighValuesSource="{StaticResource end}"
LowValuesSource="{StaticResource start}"/>
</clchart:ChartData>
</clchart:C1Chart.Data>
<clchart:C1Chart.View>
<clchart:ChartView>
<clchart:ChartView.AxisX>
<clchart:Axis IsTime="True" AnnoFormat="d"/>
</clchart:ChartView.AxisX>
</clchart:ChartView>
</clchart:C1Chart.View>
</clchart:C1Chart>
```

创建一个堆叠面积图

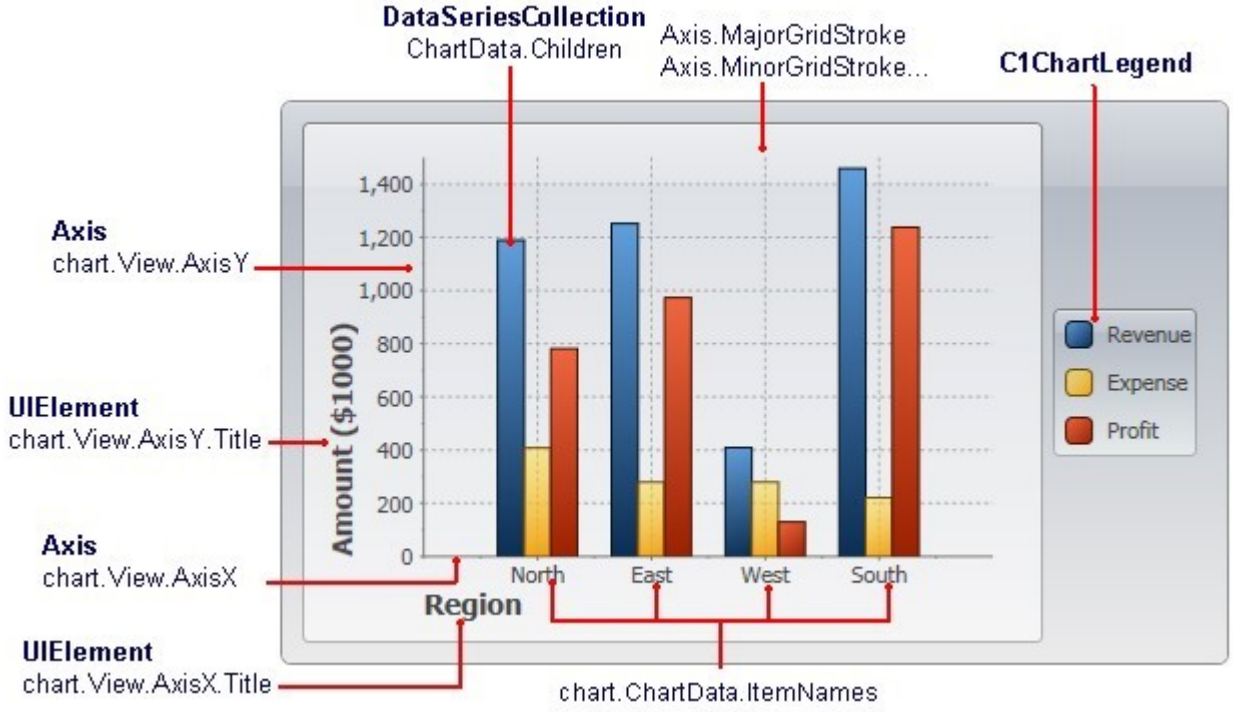
为创建一个堆叠面积图，您应当设置C1Chart.ChartType（在线文档'ChartType 属性'）属性，而不是DataSeries.ChartType（在线文档'ChartType 属性'），如下面的XAML代码所示：

XAML

```
<c1:C1Chart ChartType="AreaStacked" >
<c1:C1Chart.Data>
<c1:ChartData ItemNames="P1 P2 P3 P4 P5">
<c1:DataSeries Label="Series 1" Values="20 22 19 24 25" />

<c1:DataSeries Label="Series 2" Values="8 12 10 12 15" />
</c1:ChartData>
</c1:C1Chart.Data>
<c1:C1ChartLegend DockPanel.Dock="Right" />
</c1:C1Chart>
```

概念以及主要属性

为了通过C1Chart（在线文档 'C1Chart 类'）控件创建并格式化一个图表，理解主要的属性到图表元素的映射关系非常有用。下图演示了该映射关系：



参与创建一个典型的图表的步骤如下：

1. 选择图表类型（ChartType（在线文档 'ChartType 属性'）属性）

C1Chart支持大约三十种图表类型，包括条形图，柱状图，折线图，面积图，饼图，雷达图，极坐标图，蜡烛图以及其他一些图表类型。何种图表类型是最优选择在很大程度上取决于数据本身的特质，这一点我们将在后续章节进行讨论。

1. 设置坐标轴（AxisX以及chart.View.AxisY属性）

设置坐标轴通常包括制定坐标轴标题，主要和次要间隔的刻度线，显示在刻度线旁边标签的内容以及格式。

1. 添加一个或多个数据系列（chart.Data.Children集合）该步骤包括为图表上的每一个系列创建并填充一个DataSeries对象，并添加该对象至chart.Data.Children集合。如果您的数据每一个点金包含一个数值类型的值（Y轴），请使用普通的DataSeries对象。如果您的数据每一个数据点包含两个数值类型的值（X和Y轴），则请使用XYDataSeries对象。
2. 通过Theme以及Palette属性调整图表的外观。

Theme属性允许您在超过十种以上的内置主题中选择一个，以控制图表的整体外观。Palette属性允许您在超过二十种的内置调色板中选取一个以指定数据系列的颜色。总之，这两个属性组合可以提供多达两百中不同的选项以便毫不费力地创建具有专业外观的图表。

图表类型

该章节将具体介绍C1Chart所有可用的图表类型。

使用内置的类型是设置图表外观的最简单的方式。例如：为设置一个堆叠条形图，为ChartType属性指定相关的字符串：

XAML

```
<c1chart:C1Chart ChartType="BarStacked"> ...

</c1chart:C1Chart>
```

所有可用的图表类型由ChartType枚举的成员指定。了解更多关于不同图表类型的信息，请单击下方的"另请参见"链接。

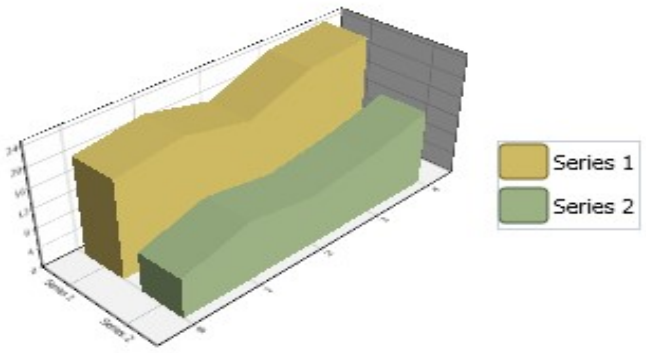
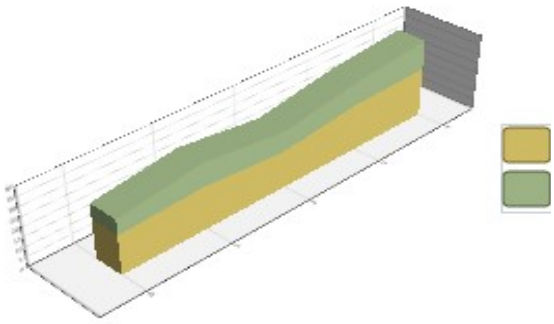
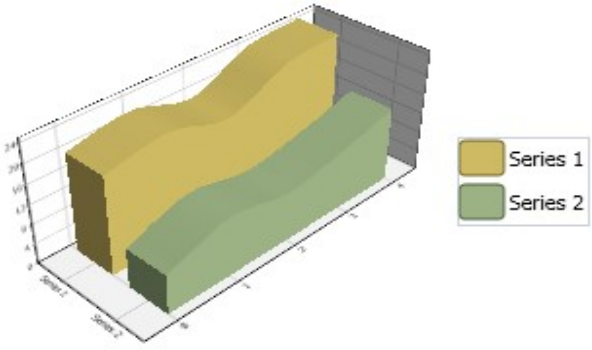
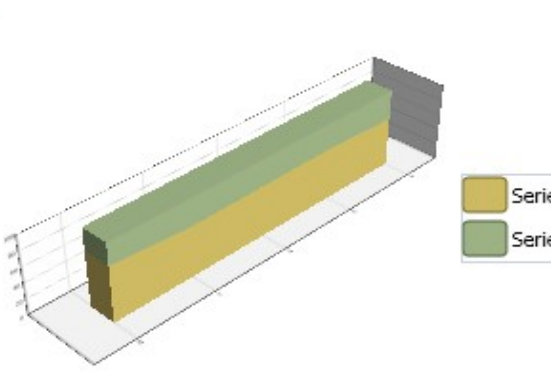
面积图

一个面积图将每一个数据系列中的数据点绘制为相连的点，并将该点下方的部分填充。每一个数据系列将该在前一个系列的上方。
数据系列可以独立绘制或者堆叠在一起。在WPF版，您还可以创建三维的面积图。WPF及Silverlight版Chart支持一以下面积图类型：
Area3D
Area3Dsmoothed
Area3Dstacked
Area3Dstacked100pc
AreaSmoothed
AreaStacked
AreaStacked100pc
您可以使用以下标记创建一个面积图：

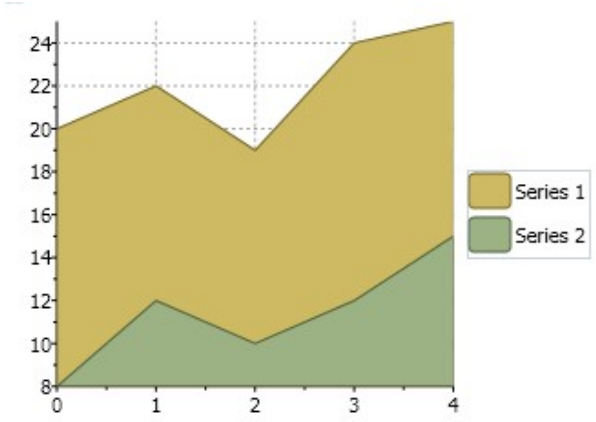
XAML

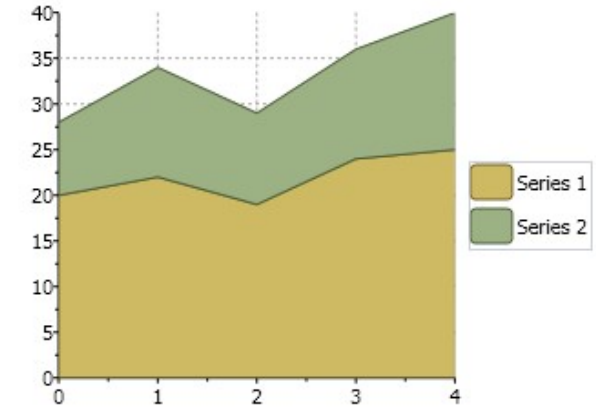
```
<c1:C1Chart ChartType="Area" >
<c1:C1Chart.Data>
<c1:ChartData ItemNames="P1 P2 P3 P4 P5">
<c1:DataSeries Label="Series 1" Values="20 22 19 24 25" />

<c1:DataSeries Label="Series 2" Values="8 12 10 12 15" />
</c1:ChartData>
</c1:C1Chart.Data>
<c1:C1ChartLegend DockPanel.Dock="Right" />
</c1:C1Chart>
```

三维面积图（仅三维面积图（仅WPF版支持）版支持）
使用AreaShape3D（在线文档 ’AreaShape3D
类’）类以访问三维图表绘图元素关联的数据，当鼠标光标悬停在某个绘图区元素上方时获取其值，获取绘图区元素的以像素表示的尺寸大小，指定是否数据点通过平滑曲线连接。以下三维面积图类型在WPF版本中可用：

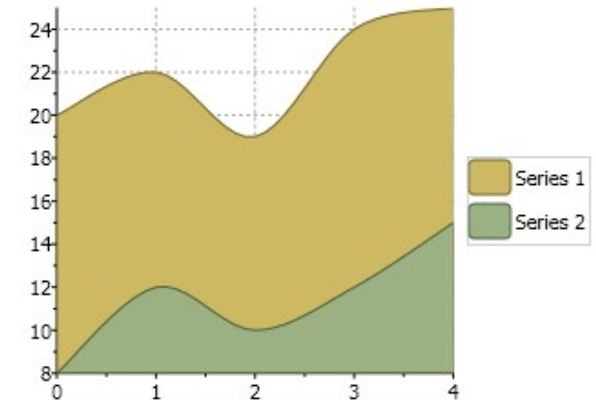
Area3D	Area3Dstacked
	
Area3Dsmoothed	Area3Dstacked100pc
	

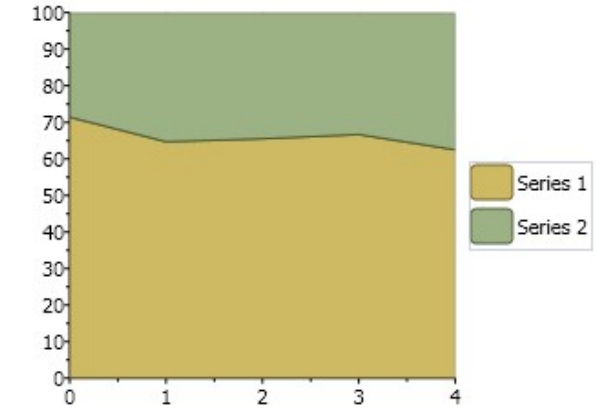
标准面积图（标准面积图（WPF及及Silverlight））
AreaAreaStacked





AreaSmoothedAreaStacked100pc

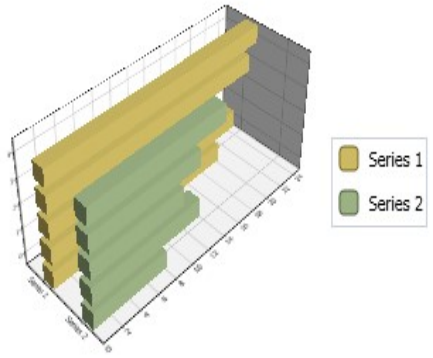
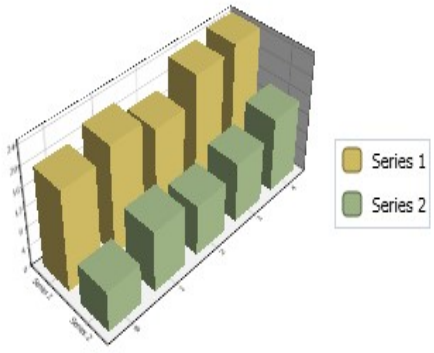
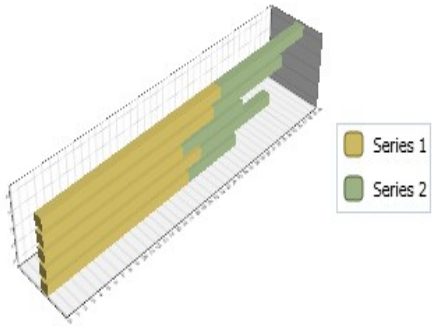
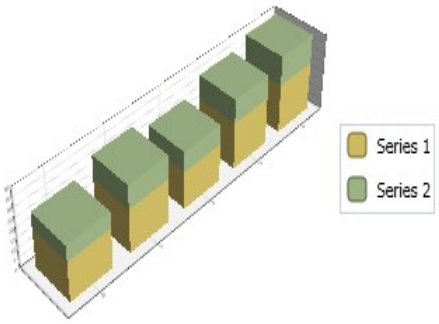




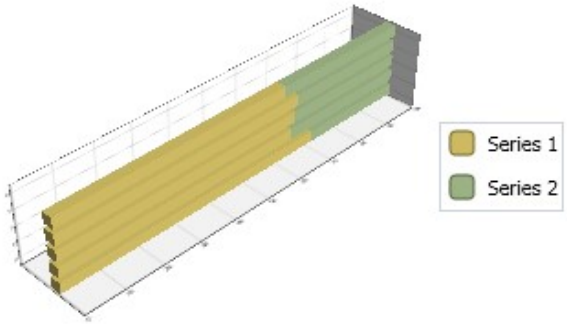
条形图和柱状图

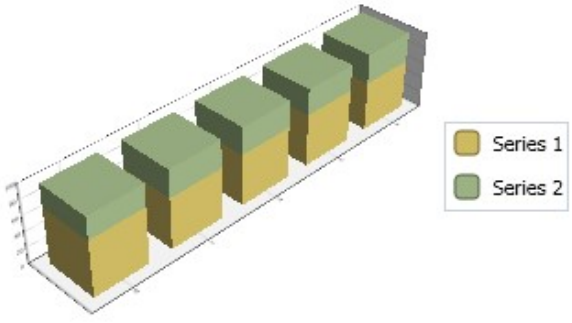
WPF版Chart支持以下条形图类型：
Bar or Column
Bar3D or Column3D

Bar3Dstacked or Column3Dstacked
Bar3Dstacked100pc or Column3Dstacked100pc
BarStacked or ColumnStacked
BarStacked100pc of ColumnStacked100pc 三维条形图和柱形图（仅三维条形图和柱形图（仅WPF））

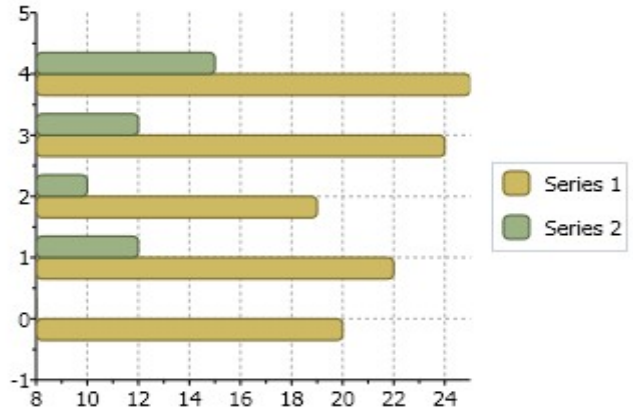
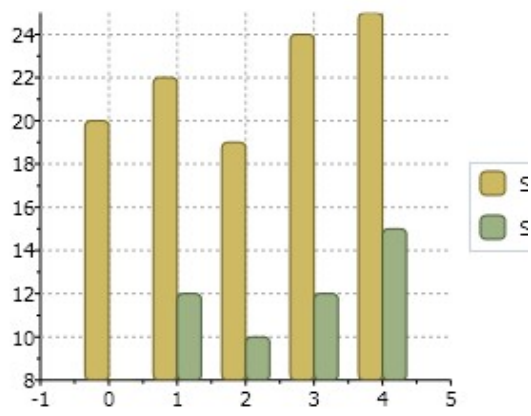
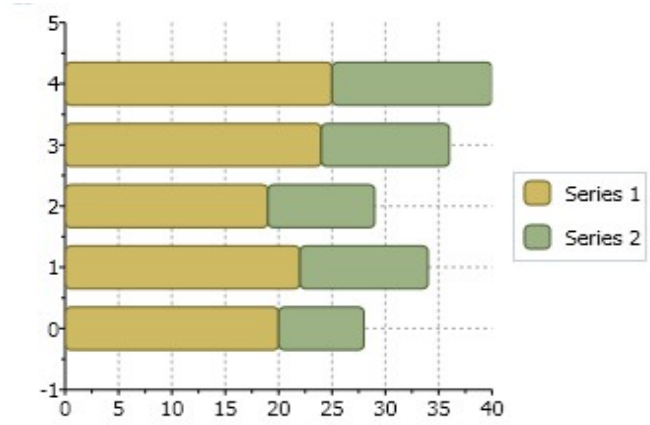
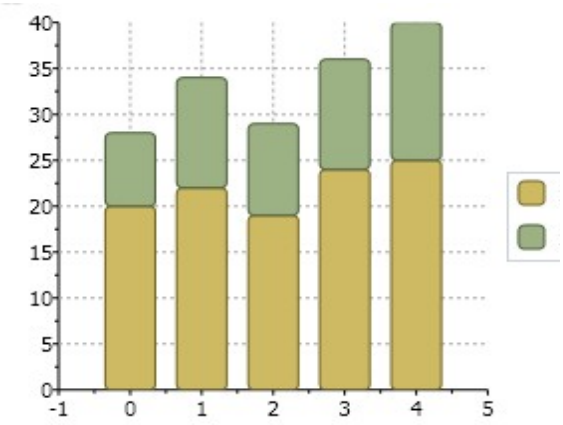
3D条形图和柱形图		
		
3DBarStacked和3DColumnStacked Charts		
		

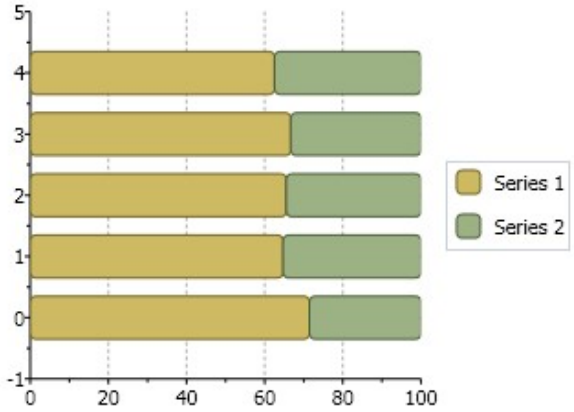
Bar3DStacked100pc和Column3DStacked100pc Charts

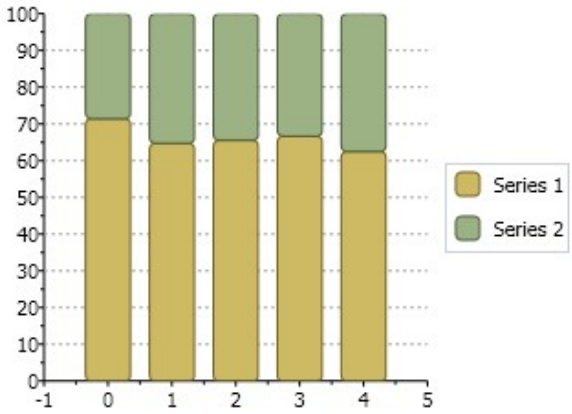




条形图和柱状图条形图和柱状图

条形图和柱状图	
	
BarStacked和ColumnStacked Charts	
	
BarStacked100pc和ColumnStacked100pc Charts	





改变柱状图以及条形图为圆角默认情况下，条形图和柱状图不显示圆角。矩形框圆角的半径可以通过Bar 类型进行设置，例如：

C#

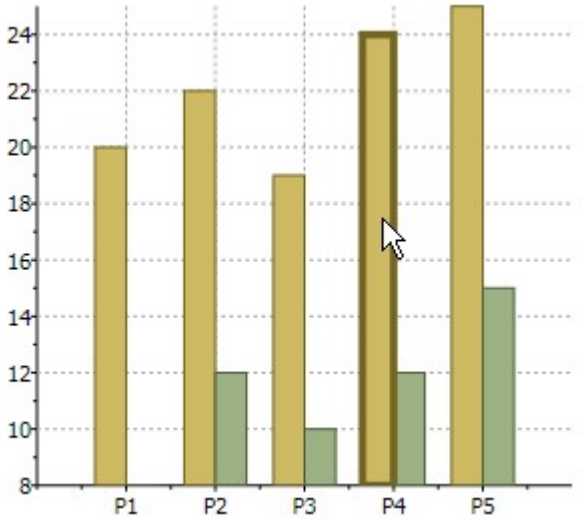
```
ds.Symbol = new Bar() { RadiusX=5, RadiusY=5};
```

为一个柱状图创建一个鼠标单击事件当单击任意柱状图中任意柱形时，您可以通过MouseDown以及MouseLeave事件添加动画，如下XAML标记代码所示：

XAML

```
<Window x:Class="MouseEvent.Window1" xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:x="http://sch
emas.microsoft.com/winfx/2006/xaml" xmlns:sys="clr-namespace:System;assembly=mscorlib" Title="Window1" Height="300" Width=
"300" xmlns:c1chart="http://schemas.componentone.com/xaml/c1chart" Loaded="Window_Loaded">
<Grid>
<Grid.Resources>
<Style x:Key="sstyle" TargetType="{x:Type c1chart:PlotElement}">
<Setter Property="StrokeThickness" Value="1" /> <Setter Property="Canvas.ZIndex" Value="0" />
<Style.Triggers>
<EventTrigger RoutedEvent="c1chart:PlotElement.MouseDown">
<BeginStoryboard>
<Storyboard>
<Int32Animation Storyboard.TargetProperty=" (Panel.ZIndex)"
To="1" />
<DoubleAnimation
Storyboard.TargetProperty="StrokeThickness"
To="4" Duration="0:0:0.3"
AutoReverse="True"
RepeatBehavior="Forever" />
</Storyboard>
</BeginStoryboard>
</EventTrigger>
<EventTrigger RoutedEvent="c1chart:PlotElement.MouseLeave">
<BeginStoryboard>
<Storyboard>
<DoubleAnimation
Storyboard.TargetProperty="StrokeThickness" />
<Int32Animation Storyboard.TargetProperty="
(Panel.ZIndex)" />
</Storyboard>
</BeginStoryboard>
</EventTrigger>
</Style.Triggers>
</Style>
</Grid.Resources>
<c1chart:C1Chart Margin="0" Name="c1Chart1" ChartType="Column">
<c1chart:C1Chart.Data>
<c1chart:ChartData>
<c1chart:ChartData.ItemNames>P1 P2 P3 P4
P5</c1chart:ChartData.ItemNames>
<c1chart:DataSeries SymbolStyle="{StaticResource sstyle}"
Values="20
22 19 24 25" />
<c1chart:DataSeries SymbolStyle="{StaticResource sstyle}" Values="8
12 10 12 15" />
</c1chart:ChartData>
</c1chart:C1Chart.Data>
```

</c1chart:C1Chart>
</Grid>
</Window>
该主题演示以下内容：该主题演示以下内容：
单击任意柱形，可以注意到矩形框周围的动画效果：



指定数据系列每一个条形或柱形的颜色

您可以在数据系列的PlotElementLoaded（在线文档 'PlotElementLoaded 事件'）事件中指定每一个条形或柱形的颜色，通过以下代码：

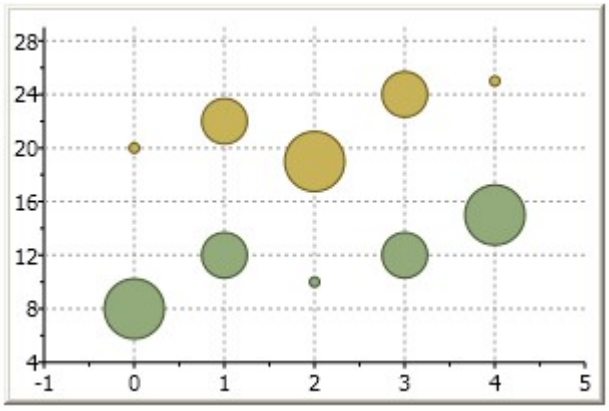
C#

```
<ac:structured-macro ac:name="unmigrated-wiki-markup" ac:schema-version="1" ac:macro-id="ec68df3f-78e1-4b57-a21c-922295bcc64c"><ac:plain-text-body><![CDATA[
```

```
var palette = new Brush[] { Brushes.Red, Brushes.Plum, Brushes.Purple };
dataSeries.PlotElementLoaded += (s, e) =>
PlotElement pe = (PlotElement)s; if (pe.Da
oint.PointIndex >= 0) pe.Fill =
palette[pe.DataPoint.PointIndex %
palette.Length];
]]></ac:plain-text-body></ac:structured-ma
};
```

气泡图

下图显示当您设置ChartType属性的值为Bubble时，该气泡图的外观：



下面的XAML标记语言将创建一个气泡图：

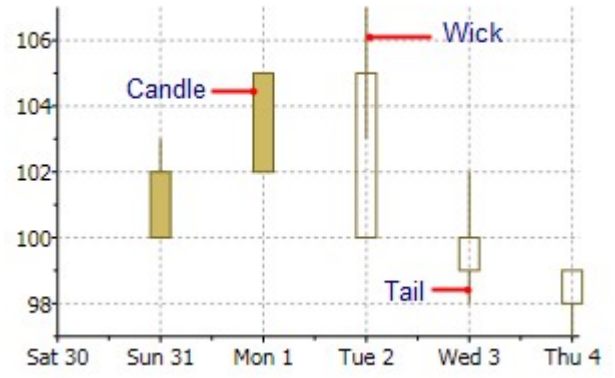
XAML

```
<c1chart:C1Chart ChartType="Bubble" c1chart:BubbleOptions.MinSize="5,5" c1chart:BubbleOptions.MaxSize="30,30" c1chart:Bubb
leOptions.Scale="Area">
<c1chart:C1Chart.Data>
<c1chart:ChartData>
<c1chart:BubbleSeries Values="20 22 19 24 25" SizeValues="1 2 3 2 1" />
<c1chart:BubbleSeries Values="8 12 10 12 15" SizeValues="3 2 1 2 3"/>
</c1chart:ChartData>
</c1chart:C1Chart.Data>
</c1chart:C1Chart>
```

K线图

K线图表示一种特殊的Hi-Lo-Open-Close图表，用于显示开闭值之间的关系以及最高值和最低值。和Hi-Lo-Open-Close

图表类似，K线图使用相同的价格数据（时间，最高值，最低值，开盘价和收盘价），唯一不同的是K线图包含一个粗粗的K线一样的身体。
K线图由以下元素组成：
K线
粗粗的像K线一样的身体使用颜色和尺寸以揭示关于开盘价和收盘价之间关系的额外信息。
一根长长的透明的K线显示了购买压力，而一个较短的，填充有颜色的K线则显示了销售压力。
一根空心的或透明的K线，预示着股票价格的上涨（高于开盘价）。在一根空心的K线形状中，K线体的底部表示开盘价，而K线体的顶部则表示收盘价。
一根填充了颜色的K线则表示了一个正在下跌的股票价格（开盘价高于收盘价）。在一个填充的K线图形中，K线体的顶端代表着开盘价，而K线体的底部则代表收盘价。
烛芯烛芯是K线上方的线，描述的是高价的范围。
尾线
尾线是K线底部的线，描述了低价的范围。
下图通过标签描述了每一个K线图表中的元素：

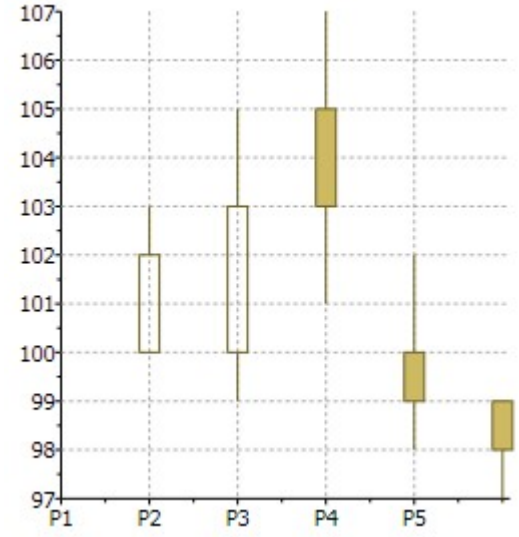


下面的图像表示当您设置了ChartType属性的值为Candle，并指定了XValues（在线文档 'XValues 属性'），HighValues （在线文档 'HighValues 属性'），LowValues（在线文档 'LowValues 属性'），OpenValues（在线文档 'OpenValues 属性'），以及CloseValues（在线文档 'CloseValues 属性'）的值时K线图的外观，如下所示：

XAML

```
<c1chart:C1Chart ChartType="Candle">
<c1chart:C1Chart.Data>
<c1chart:ChartData>
<c1chart:HighLowOpenCloseSeries
XValues="1 2 3 4 5"
HighValues="103 105 107 102 99" LowValues="100 99 101 98 97"
OpenValues="100 100 105 100 99" CloseValues="102 103 103 99 98"
/>
</c1chart:ChartData>
</c1chart:C1Chart.Data>
</c1chart:C1Chart>
```

以上XAML标记代码将生成一个类似于以下图片的图表：



改变蜡烛的宽度

为改变蜡烛的宽度，请使用SymbolSize（在线文档 'SymbolSize 属性'）属性如下：

C#

```
ds.SymbolSize = new Size(5, 5);
```