ComponentOne

Chart3D for WPF and Silverlight

GrapeCity US

GrapeCity
201 South Highland Avenue, Suite 301
Pittsburgh, PA 15206
Tel: 1.800.858.2739 | 412.681.4343
Fax: 412.681.4384
Website: https://www.grapecity.com/en/
E-mail: us.sales@grapecity.com

Trademarks

Warranty

ComponentOne warrants that the media on which the software is delivered is free from defects in material and workmanship, assuming normal use, for a period of 90 days from the date of purchase. If a defect occurs during this time, you may return the defective media to ComponentOne, along with a dated proof of purchase, and ComponentOne will replace it at no charge. After 90 days, you can obtain a replacement for the defective media by sending it and a check for $25 (to cover postage and handling) to ComponentOne.

Except for the express warranty of the original media on which the software is delivered is set forth here, ComponentOne makes no other warranties, express or implied. Every attempt has been made to ensure that the information contained in this manual is correct as of the time it was written. ComponentOne is not responsible for any errors or omissions. ComponentOne's liability is limited to the amount you paid for the product. ComponentOne is not liable for any special, consequential, or other damages for any reason.

Copying and Distribution

Table of Contents

Product Summary

Help with WPF and Silverlight Edition

Getting Started

- For information on installing **ComponentOne Studio WPF Edition**, licensing, technical support, namespaces and creating a project with the control, please visit Getting Started with WPF Edition.
- For information on installing **ComponentOne Studio Silverlight Edition**, licensing, technical support, namespaces and creating a project with the control, please visit Getting Started with Silverlight Edition.

Key Features

Chart3D for WPF and Silverlight provides the following unique features:

- **Surface Chart Types**

  Chart3D for WPF and Silverlight includes 6 different surface chart types. You can choose among surface charts with wire frames, contour levels, and zones when you create your chart. Customize whether contours and meshes appear along each axis. See Chart Types for more information.

- **Show a Legend**

  Display a chart legend for zoned charts. See Adding a Chart Legend for more information.

- **Set Rotation and Elevation**

  Rotate the chart to any angle by setting the Azimuth and Elevation properties. See Rotating and Elevating a Chart for more information.

- **Floors and Ceilings**

  The contours and zones determined for the chart can be displayed on the ceiling or floor of the plot cube. Display contours and zones on the top and bottom of the chart to give a 2D representation in addition to the 3D model. See Adding Floors and Ceilings for examples.

- **Show 2D Projection**

  Easily create flat (or two-dimensional) 3D charts such as heat maps. See Creating a Two-Dimensional Chart for more information.

- **Create Custom Axis Annotations**

  Customize the exact layout and values of the axis labels using the AnnoTemplate property. See Custom Axis Annotation Template for more information.

Quick Start

Step 1: Adding Chart3D to your Project

In this step you'll either begin in Visual Studio to create a chart application using **Chart3D for WPF and Silverlight**.

1. Create a new WPF or Silverlight application in Visual Studio.
2. Double-click the C1Chart3D control in the Toolbox to add it to your window. The XAML markup will look similar to the following:

XAML - WPF

```
<Window x:Class="WFPChart3DQuickstart.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:c1chart3d="http://schemas.componentone.com/xaml/c1chart"
        Title="MainWindow" Height="350" Width="525" >
    <Grid>
        <c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left"
Margin="10,10,0,0" Name="c1Chart3D1" VerticalAlignment="Top" Width="200">
            <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
        </c1chart3d:C1Chart3D>
    </Grid>
</Window>
```

XAML - Silverlight

```
<UserControl xmlns:c1chart3d="clr-
namespace:C1.Silverlight.Chart3D;assembly=C1.Silverlight.Chart3D"
x:Class="MySilverlightApplication.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="400">
    <Grid x:Name="LayoutRoot" Background="White">
        <c1chart3d:C1Chart3D />
    </Grid>
</UserControl>
```

In the next step you will add your own data to the chart.

Step 2: Adding Data

Chart3D for WPF and Silverlight supports data values defined as a two-dimensional array of z-values where the first index corresponds to X and the second index corresponds to Y. One of the most common uses of a **Chart3D** is plotting 3D functions. Let's plot a function defined as the following:

z(x,y) = x*x - y*y;

in the range

-1 <= x <= 1

-1 <= y <= 1

To do this, follow these steps:

1. Select **View | Code** in your project.
2. Add the following statement at the top of the page:

C# - WPF

```
using C1.WPF.Chart3D;
```

C# - Silverlight

```
using C1.Silverlight.Chart3D
```

3. Add the following code to define the data:

C#

```
public MainWindow()
        {
            InitializeComponent();
            c1Chart3D1.Children.Clear();

            // create 2D array 10x10
            int xlen = 10, ylen = 10;
            var zdata = new double[xlen, ylen];
            double stepx = 2.0 / (xlen - 1);
            double stepy = 2.0 / (ylen - 1);
            // calculate function for all points in the range
            for (int ix = 0; ix < xlen; ix++)
                for (int iy = 0; iy < ylen; iy++)
                {
                    double x = -1.0 + ix * stepx; //  -1 <= x <= 1
                    double y = -1.0 + iy * stepy; //  -1 <= x <= 1
                    zdata[ix, iy] = x * x - y * y;
                }
            // create data series
            var ds = new GridDataSeries();
            ds.Start = new Point(-1, -1); // start for x,y
            ds.Step = new Point(stepx, stepy); // step for x,y
            ds.ZData = zdata; // z-values
            // add series to the chart
            c1Chart3D1.Children.Add(ds);
        }
```

In the next step you will change the appearance of the chart.

Step 3: Changing the Chart Appearance

In this step you will change the chart type to **SurfaceZone**, set the appearance for the chart floor, rotate the chart, and change its elevation angle.

1. Select the C1Chart3D control in your window and click **View | Properties Window**.
2. Change the chart type by clicking the drop-down arrow next to the ChartType property and selecting **SurfaceZone**.
3. Click the drop-down arrow next to the FloorAppearance property and select **ZoneContour**.
4. Enter **170** next to the Elevation property.
5. Set the Azimuth property to 20.

   The XAML markup will now look similar to the following:

XAML - WPF

```
<Window x:Class="WFPChart3DQuickstart.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525"
xmlns:c1chart3d="http://schemas.componentone.com/xaml/c1chart">
    <Grid>
        <c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left"
Margin="10,10,0,0" Name="c1Chart3D1" VerticalAlignment="Top" Width="200"
ChartType="SurfaceZone" CeilAppearance="None" FloorAppearance="ZoneContour"
Elevation="170" Azimuth="20">
            <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
        </c1chart3d:C1Chart3D>
    </Grid>
</Window>
```

XAML - Silverlight

```
<UserControl xmlns:c1chart3d="clr-
namespace:C1.Silverlight.Chart3D;assembly=C1.Silverlight.Chart3D"
x:Class="SilverlightApplication12.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="400">
    <Grid x:Name="LayoutRoot" Background="White">
<c1chart3d:C1Chart3D ChartType="SurfaceZone" FloorAppearance="ZoneContour"
Azimuth="20" Elevation="170">
            <c1chart3d:C1Chart3D />
    </Grid>
```

```
</UserControl>
```

In the next step you will add a legend for the chart.

Step 4: Adding a Legend

In this step you will add a legend for the chart using only one property.

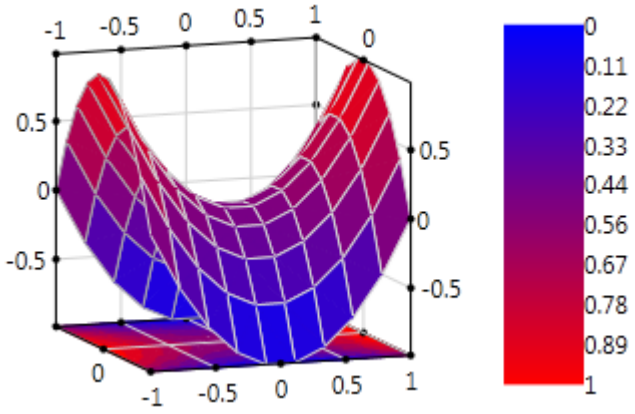
1. Select the C1Chart3D control in your window and click **View | Properties Window**.
2. Add a legend by clicking the drop-down arrow next to the Legend property and selecting C1Chart3DLegend.

In the next step, you will run the project and view your new chart.

Step 5: Running the Project

In this step, you will run the project to see the chart.

Select **Start | Debugging** or press the **F5** key. You will see the **Chart3D** and legend.



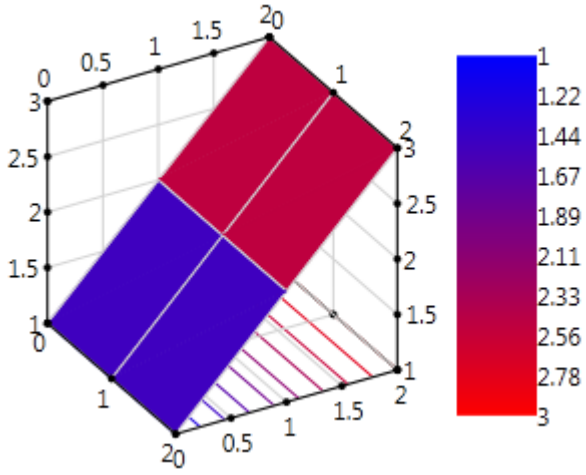
Congratulations! You have completed the **Chart3D for WPF and Silverlight** quick start.

XAML Quick Reference

Example: Set up a 3D Surface Chart

The following XAML shows how to declare the C1Chart3D control, set the ChartType, FloorAppearance, GridDataSeries, and add a C1Chart3DLegend control. The XAML can be added inside the <Grid>…</Grid> tags.

The following chart is produced using the following XAML code:



XAML

```
<Grid>
    <c1chart3d:C1Chart3D Name="c1Chart3D1" ChartType="SurfaceZone"
FloorAppearance="Contour">
        <c1chart3d:C1Chart3D.Legend>
            <c1chart3d:C1Chart3DLegend />
        </c1chart3d:C1Chart3D.Legend>
        <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
    </c1chart3d:C1Chart3D>
</Grid>
```

DirectX rendering

Note: This topic applies only to WPF applications.

The **DirectX Rendering Mode** allows you to create a high performance rendering of surface charts for larger data sets with **Direct3D** rendering. To implement DirectX Rendering, mode, just set the RenderMode property to **Direct3D**.

To use the **Direct3D** render mode in your WPF applications, you should distribute the following assemblies along with your application. During development these assemblies should be found in the same directory as the **C1.WPF.C1Chart3D** assembly but do not need to be directly referenced by the project.

- SharpDX.D3DCompiler.dll
- SharpDX.Direct3D9.dll
- SharpDX.Direct3D10.dll
- SharpDX.dll
- SharpDX.DXGI.dll

System Requirements for **Direct3D Rendering** include:

- DirectX Version 10.0 or higher
- D3D9 Overlay Support

The **Direct3D** mode does not work on some virtual machines (Hyper-V) and through remote desktop connections since the required hardware layer is not supported in such cases.

How to use C1Chart3D for WPF

Data Layout in GridDataSeries

Data in **Chart3D for WPF and Silverlight** is defined in the GridDataSeries class. It provides the following properties related to data:

Property	Description
ZData	Gets or sets two-dimensional array of values on the grid.
Start	Gets or sets the start point of data.
Step	Gets or sets the step of grid data.
ContourData	Gets or sets two-dimensional array of contour data (4-dimensional chart).

The first index of the 2D array in the ZData/ ContourData properties corresponds to the X and the second index corresponds to the Y. The layout for **ZData** is shown in the following table.

	Start.X	Start.X + Step.X	Start.X + 2*Step.X
Start.Y	ZData[0,0]	ZData[1,0]	ZData[2,0]
Start.Y + Step.Y	ZData[0,1]	ZData[1,1]	ZData[2,1]
Start.Y + 2*Step.Y	ZData[0,2]	ZData[1,2]	ZData[2,2]

The element of the array ZData[0,0] corresponds to the point (Start.X, Start.Y), ZData[1,0] is the z-value at (Start.X+Step.X, Start.Y), and so on.

Chart Types

Using built-in types is the simplest way to set up the chart's appearance. For example, to set up a wire-frame surface chart, specify the corresponding string in the ChartType property:

XAML

```
<C1_WPF_C1Chart3D:C1Chart3D ChartType="SurfaceWireframe"/>
```

The available chart types are specified by the members of enumeration Chart3DType.

The list of available built-in chart types is presented in the table below.

Name	**Description**
SurfaceWireframe	Wire-frame surface chart
SurfaceWireframeContour	Wire-frame surface chart with contour levels

Surface	Surface chart
SurfaceContour	Surface chart with contour levels
SurfaceZone	Surface chart with zones
SurfaceZoneContour	Surface chart with contour zones

Surface charts examine the distribution of data and draw contour lines to demarcate each of the contour levels. You can set the number of contour levels in the chart using the ContourLevelsCount property.

Chart3D for WPF and Silverlight also supports mesh lines which can be added to the X-axis, Y-axis, or both axes. Use the SurfaceMeshAppearance property to determine where the mesh lines appear.

Adding a Chart Legend

Chart3D for WPF and Silverlight allows you to show a legend for contour or zone charts.The legend element displays information about each data series of the chart. The chart legend displays the mapping between the physical colors and the data series. It is controlled by the Legend property of C1Chart3D.

To add a legend for the chart, follow these steps:

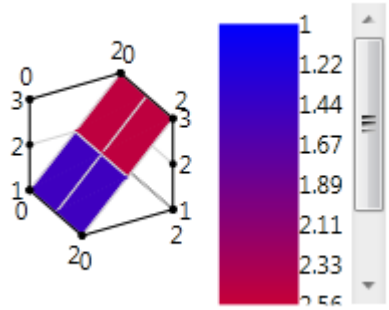
1. Add a **C1Chart3D** control to the window.
2. Set the ChartType to **SurfaceZone** or **SurfaceZoneContour**. The XAML will look similar to the following:

```
XAML
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="137,77,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" ChartType="SurfaceZone">
          <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
</c1chart3d:C1Chart3D>
```

3. Set the Legend property to C1Chart3DLegend. Now the XAML will look like this:

```
XAML
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="178,85,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" ChartType="SurfaceZone">
            <c1chart3d:C1Chart3D.Legend>
                <c1chart3d:C1Chart3DLegend />
            </c1chart3d:C1Chart3D.Legend>
            <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
</c1chart3d:C1Chart3D>
```

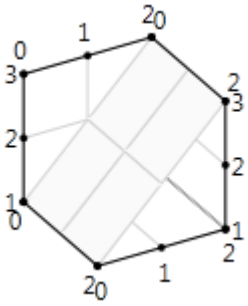
Your chart will look similar to the following:



Note that a legend is visible for contour or zone charts only.

Rotating and Elevating a Chart

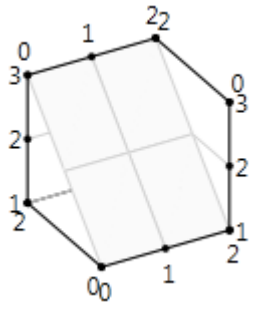
It's easy to rotate a chart using the Azimuth property. The default angle for C1Chart3D is 30 degrees.



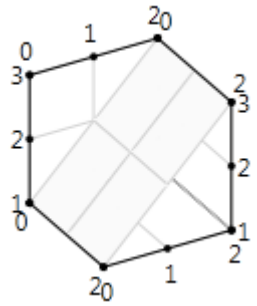
The following XAML sets the **Azimuth** property to 120 degrees.

XAML

```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="96,51,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" Azimuth="120">
```



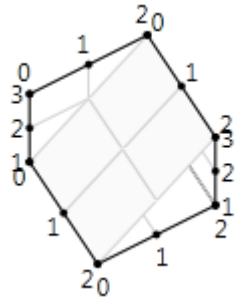
Each chart has an elevation angle, or an incline above the X-axis. When you first add a chart to the window, the elevation angle is 150 degrees, by default.



To change the incline angle, use the Elevation property. The following XAML sets the **Elevation** property to 120.

XAML

```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="96,51,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" Azimuth="30" Elevation="120">
```

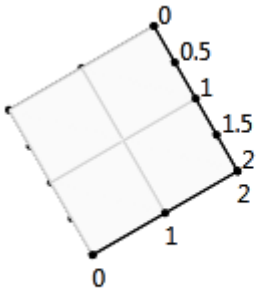


You can also create a two-dimensional chart using the **Elevation** property. See Creating a Two-Dimensional Chart for more information.

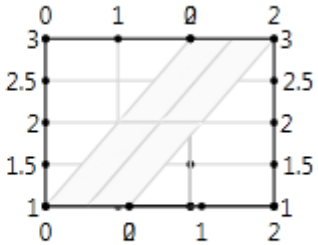
There are patio faces the bed To append

Creating a Two-Dimensional Chart

You can easily create a flat 3D chart, such as a heat map, by changing the Elevation property of a chart. Each chart has an elevation angle, or an incline above the X-axis. When you first add a chart to the window, the elevation angle is 150 degrees, by default. You can flatten the chart by setting the Elevation property to 90 or 180, depending on how you want to flatten it with respect to the X-axis.



Elevation = "90"



Elevation = "180"

The following XAML sets the Elevation property for a chart to 90.

XAML

```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="234,139,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" Elevation="90">
     <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
</c1chart3d:C1Chart3D>
```

Creating a Custom Axis Annotation

The 3D chart has an X, Y, and Z axis that can be controlled by the C1Chart3D object. You can customize the look and feel of the axes through the C1Chart3D object and then set the X, Y, or Z axis object through the AxisX, AxisY, and AxisZ properties for the new Axes settings to appear on the X, Y, and Z axes.

This section describes the most common axis configuration scenarios used to make the chart more readable such as labeling, scaling, and formatting.

Custom Axis Annotation Template

The following sample shows how to use the AnnoTemplate property to create axis labels with color depending on the corresponding axis value.

Add the following XAML to your MainWindow.xaml or your MainPage.xaml:

Note: In a Silverlight application, you'll add <Page.Resources> </Page.Resources> tags instead of the <Window.Resources> tags.

XAML

```
<Window.Resources>
    <!-- instance of converter -->
    <local:LabelToColorConvereter x:Key="cnv" />
  </Window.Resources>
    <Grid>
    <c1chart3d:C1Chart3D Name="c1Chart3D1" >
      <c1chart3d:GridDataSeries ZDataString="-1 -1 -1,0 0 0, 1 1 1" />
      <c1chart3d:C1Chart3D.AxisZ>
        <c1chart3d:Axis3D>
          <!-- set axis annotation template  -->
          <c1chart3d:Axis3D.AnnoTemplate>
            <DataTemplate >
              <!-- DataContext is axis label(string), use converter to set color  -->
                <TextBlock Width="20" Height="12" Text="{Binding}"
Foreground="{Binding Converter={StaticResource cnv}}" />
            </DataTemplate>
          </c1chart3d:Axis3D.AnnoTemplate>
        </c1chart3d:Axis3D>
      </c1chart3d:C1Chart3D.AxisZ>
    </c1chart3d:C1Chart3D>
  </Grid>
```

Select **View | Code** and add the following code. The binding converter selects the brush of the axis label depending on the value.

C#

```
public class LabelToColorConvereter : IValueConverter
{
  static Brush belowZero = new SolidColorBrush(Colors.Blue);
  static Brush aboveZero = new SolidColorBrush(Colors.Red);
  static Brush zero = new SolidColorBrush(Colors.DarkGray);
  public object Convert(object value, Type targetType, object parameter,
    System.Globalization.CultureInfo culture)
  {
    // string representing axis label
    string s = value as string;
    if (!string.IsNullOrEmpty(s))
    {
      var dv = double.Parse(s);
      // return brush depending on the value
      if (dv < 0)
        return belowZero;
      else if (dv > 0)
        return aboveZero;
      else
        return zero;
    }
    return value;
  }

  public object ConvertBack(object value, Type targetType, object parameter,
    System.Globalization.CultureInfo culture)
  {
    throw new NotImplementedException();
  }
}
```

Axis Title

Adding a title to an axis clarifies what is charted along that axis. A title with a specified font can be added to any axis.

To Add an Axis Title:

Use the axis Title property to add a title to an axis.

1. In the Visual Studio **Properties** window, expand the **AxisX**, **AxisY**, or **AxisZ** property.
2. Enter a title next to the **Title** property.
3. To remove the title, delete the text.

Chart3D for WPF and Silverlight Overview

Graph your data in three dimensions!

With **Chart3D for WPF and Silverlight** you can create professional looking 3D surface charts with options for contour levels and zones. Rotate the chart to any angle, show a chart legend and more.

Getting Started

Help with WPF and Silverlight Edition

Getting Started

- For information on installing **ComponentOne Studio WPF Edition**, licensing, technical support, namespaces and creating a project with the control, please visit Getting Started with WPF Edition.
- For information on installing **ComponentOne Studio Silverlight Edition**, licensing, technical support, namespaces and creating a project with the control, please visit Getting Started with Silverlight Edition.

Key Features

Chart3D for WPF and Silverlight provides the following unique features:

- **Surface Chart Types**

  Chart3D for WPF and Silverlight includes 6 different surface chart types. You can choose among surface charts with wire frames, contour levels, and zones when you create your chart. Customize whether contours and meshes appear along each axis. See Chart Types for more information.

- **Show a Legend**

  Display a chart legend for zoned charts. See Adding a Chart Legend for more information.

- **Set Rotation and Elevation**

  Rotate the chart to any angle by setting the Azimuth and Elevation properties. See Rotating and Elevating a Chart for more information.

- **Floors and Ceilings**

  The contours and zones determined for the chart can be displayed on the ceiling or floor of the plot cube. Display contours and zones on the top and bottom of the chart to give a 2D representation in addition to the 3D model. See Adding Floors and Ceilings for examples.

- **Show 2D Projection**

  Easily create flat (or two-dimensional) 3D charts such as heat maps. See Creating a Two-Dimensional Chart for more information.

- **Create Custom Axis Annotations**

  Customize the exact layout and values of the axis labels using the AnnoTemplate property. See Custom Axis Annotation Template for more information.

Quick Start: Chart3D for WPF and Silverlight

Step 1: Adding Chart3D to your Project

In this step you'll either begin in Visual Studio to create a chart application using **Chart3D for WPF and Silverlight**.

1. Create a new WPF or Silverlight application in Visual Studio.
2. Double-click the C1Chart3D control in the Toolbox to add it to your window. The XAML markup will look similar to the following:

XAML - WPF

```
<Window x:Class="WFPChart3DQuickstart.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:c1chart3d="http://schemas.componentone.com/xaml/c1chart"
        Title="MainWindow" Height="350" Width="525" >
    <Grid>
        <c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left"
Margin="10,10,0,0" Name="c1Chart3D1" VerticalAlignment="Top" Width="200">
            <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
        </c1chart3d:C1Chart3D>
    </Grid>
</Window>
```

XAML - Silverlight

```
<UserControl xmlns:c1chart3d="clr-
namespace:C1.Silverlight.Chart3D;assembly=C1.Silverlight.Chart3D"
x:Class="MySilverlightApplication.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="400">
    <Grid x:Name="LayoutRoot" Background="White">
        <c1chart3d:C1Chart3D />
    </Grid>
</UserControl>
```

In the next step you will add your own data to the chart.

Step 2: Adding Data

Chart3D for WPF and Silverlight supports data values defined as a two-dimensional array of z-values where the first index corresponds to X and the second index corresponds to Y. One of the most common uses of a **Chart3D** is plotting 3D functions. Let's plot a function defined as the following:

z(x,y) = x*x - y*y;

in the range

-1 <= x <= 1

-1 <= y <= 1

To do this, follow these steps:

1. Select **View | Code** in your project.
2. Add the following statement at the top of the page:

C# - WPF

```
using C1.WPF.Chart3D;
```

C# - Silverlight

```
using C1.Silverlight.Chart3D
```

3. Add the following code to define the data:

C#

```
public MainWindow()
        {
            InitializeComponent();
            c1Chart3D1.Children.Clear();

            // create 2D array 10x10
            int xlen = 10, ylen = 10;
            var zdata = new double[xlen, ylen];
            double stepx = 2.0 / (xlen - 1);
            double stepy = 2.0 / (ylen - 1);
            // calculate function for all points in the range
            for (int ix = 0; ix < xlen; ix++)
                for (int iy = 0; iy < ylen; iy++)
                {
                    double x = -1.0 + ix * stepx; //  -1 <= x <= 1
                    double y = -1.0 + iy * stepy; //  -1 <= x <= 1
                    zdata[ix, iy] = x * x - y * y;
                }
            // create data series
            var ds = new GridDataSeries();
            ds.Start = new Point(-1, -1); // start for x,y
            ds.Step = new Point(stepx, stepy); // step for x,y
            ds.ZData = zdata; // z-values
            // add series to the chart
            c1Chart3D1.Children.Add(ds);
        }
```

In the next step you will change the appearance of the chart.

Step 3: Changing the Chart Appearance

In this step you will change the chart type to **SurfaceZone**, set the appearance for the chart floor, rotate the chart, and change its elevation angle.

1. Select the C1Chart3D control in your window and click **View | Properties Window**.
2. Change the chart type by clicking the drop-down arrow next to the ChartType property and selecting **SurfaceZone**.
3. Click the drop-down arrow next to the FloorAppearance property and select **ZoneContour**.
4. Enter **170** next to the Elevation property.
5. Set the Azimuth property to 20.

   The XAML markup will now look similar to the following:

XAML - WPF

```
<Window x:Class="WFPChart3DQuickstart.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525"
xmlns:c1chart3d="http://schemas.componentone.com/xaml/c1chart">
    <Grid>
        <c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left"
Margin="10,10,0,0" Name="c1Chart3D1" VerticalAlignment="Top" Width="200"
ChartType="SurfaceZone" CeilAppearance="None" FloorAppearance="ZoneContour"
Elevation="170" Azimuth="20">
            <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
        </c1chart3d:C1Chart3D>
    </Grid>
</Window>
```

XAML - Silverlight

```
<UserControl xmlns:c1chart3d="clr-
namespace:C1.Silverlight.Chart3D;assembly=C1.Silverlight.Chart3D"
x:Class="SilverlightApplication12.MainPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
    mc:Ignorable="d"
    d:DesignHeight="300" d:DesignWidth="400">
    <Grid x:Name="LayoutRoot" Background="White">
<c1chart3d:C1Chart3D ChartType="SurfaceZone" FloorAppearance="ZoneContour"
Azimuth="20" Elevation="170">
            <c1chart3d:C1Chart3D />
    </Grid>
```

```
</UserControl>
```

In the next step you will add a legend for the chart.

Step 4: Adding a Legend

In this step you will add a legend for the chart using only one property.

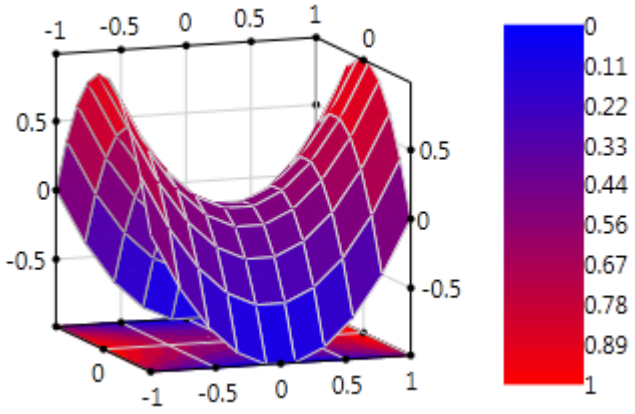
1. Select the C1Chart3D control in your window and click **View | Properties Window**.
2. Add a legend by clicking the drop-down arrow next to the Legend property and selecting C1Chart3DLegend.

In the next step, you will run the project and view your new chart.

Step 5: Running the Project

In this step, you will run the project to see the chart.

Select **Start | Debugging** or press the **F5** key. You will see the **Chart3D** and legend.



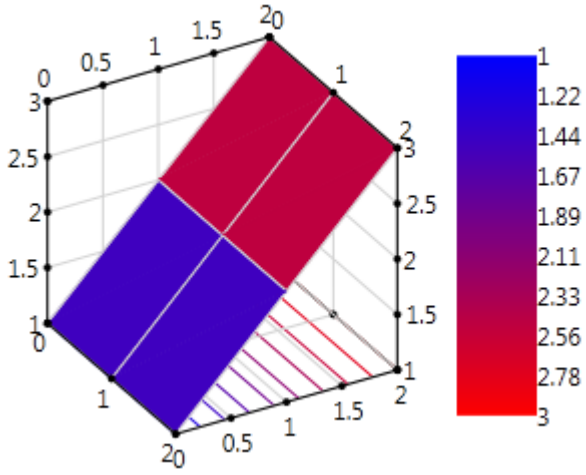
Congratulations! You have completed the **Chart3D for WPF and Silverlight** quick start.

XAML Quick Reference

Example: Set up a 3D Surface Chart

The following XAML shows how to declare the C1Chart3D control, set the ChartType, FloorAppearance, GridDataSeries, and add a C1Chart3DLegend control. The XAML can be added inside the <Grid>…</Grid> tags.

The following chart is produced using the following XAML code:



XAML

```
<Grid>
    <c1chart3d:C1Chart3D Name="c1Chart3D1" ChartType="SurfaceZone"
FloorAppearance="Contour">
        <c1chart3d:C1Chart3D.Legend>
            <c1chart3d:C1Chart3DLegend />
        </c1chart3d:C1Chart3D.Legend>
        <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
    </c1chart3d:C1Chart3D>
</Grid>
```

Using Chart3D for WPF and Silverlight

Data Layout in GridDataSeries

Data in **Chart3D for WPF and Silverlight** is defined in the GridDataSeries class. It provides the following properties related to data:

Property	Description
ZData	Gets or sets two-dimensional array of values on the grid.
Start	Gets or sets the start point of data.
Step	Gets or sets the step of grid data.
ContourData	Gets or sets two-dimensional array of contour data (4-dimensional chart).

The first index of the 2D array in the ZData/ ContourData properties corresponds to the X and the second index corresponds to the Y. The layout for **ZData** is shown in the following table.

	Start.X	Start.X + Step.X	Start.X + 2*Step.X
Start.Y	ZData[0,0]	ZData[1,0]	ZData[2,0]
Start.Y + Step.Y	ZData[0,1]	ZData[1,1]	ZData[2,1]
Start.Y + 2*Step.Y	ZData[0,2]	ZData[1,2]	ZData[2,2]

The element of the array ZData[0,0] corresponds to the point (Start.X, Start.Y), ZData[1,0] is the z-value at (Start.X+Step.X, Start.Y), and so on.

Chart Types

Using built-in types is the simplest way to set up the chart's appearance. For example, to set up a wire-frame surface chart, specify the corresponding string in the ChartType property:

XAML

```
<C1_WPF_C1Chart3D:C1Chart3D ChartType="SurfaceWireframe"/>
```

The available chart types are specified by the members of enumeration Chart3DType.

The list of available built-in chart types is presented in the table below.

Name	Description
SurfaceWireframe	Wire-frame surface chart
SurfaceWireframeContour	Wire-frame surface chart with contour levels
Surface	Surface chart
SurfaceContour	Surface chart with contour levels
SurfaceZone	Surface chart with zones
SurfaceZoneContour	Surface chart with contour zones

Surface charts examine the distribution of data and draw contour lines to demarcate each of the contour levels. You can set the number of contour levels in the chart using the ContourLevelsCount property.

Chart3D for WPF and Silverlight also supports mesh lines which can be added to the X-axis, Y-axis, or both axes. Use the SurfaceMeshAppearance property to determine where the mesh lines appear.

Adding a Chart Legend

Chart3D for WPF and Silverlight allows you to show a legend for contour or zone charts.The legend element displays information about each data series of the chart. The chart legend displays the mapping between the physical colors and the data series. It is controlled by the Legend property of C1Chart3D.

To add a legend for the chart, follow these steps:

1. Add a **C1Chart3D** control to the window.

2. Set the ChartType to **SurfaceZone** or **SurfaceZoneContour**. The XAML will look similar to the following:

XAML

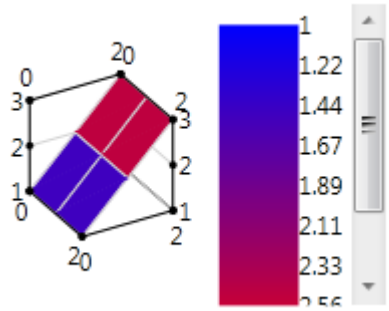
```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="137,77,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" ChartType="SurfaceZone">
          <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
</c1chart3d:C1Chart3D>
```

3. Set the Legend property to C1Chart3DLegend. Now the XAML will look like this:

XAML

```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="178,85,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" ChartType="SurfaceZone">
            <c1chart3d:C1Chart3D.Legend>
                <c1chart3d:C1Chart3DLegend />
            </c1chart3d:C1Chart3D.Legend>
            <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
</c1chart3d:C1Chart3D>
```

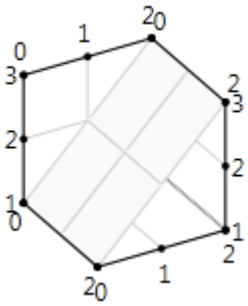
Your chart will look similar to the following:



Note that a legend is visible for contour or zone charts only.

Rotating and Elevating a Chart

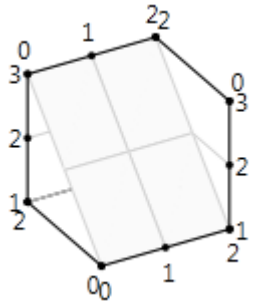
It's easy to rotate a chart using the Azimuth property. The default angle for C1Chart3D is 30 degrees.



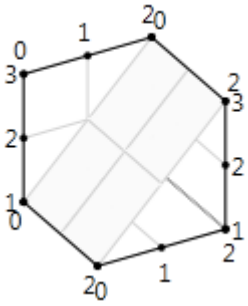
The following XAML sets the **Azimuth** property to 120 degrees.

XAML

```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="96,51,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" Azimuth="120">
```



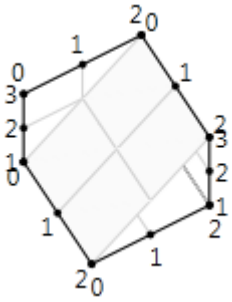
Each chart has an elevation angle, or an incline above the X-axis. When you first add a chart to the window, the elevation angle is 150 degrees, by default.



To change the incline angle, use the Elevation property. The following XAML sets the **Elevation** property to 120.

XAML

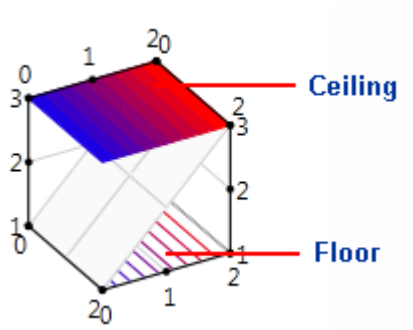
```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="96,51,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" Azimuth="30" Elevation="120">
```



You can also create a two-dimensional chart using the **Elevation** property. See Creating a Two-Dimensional Chart for more information.

Adding Floors and Ceilings

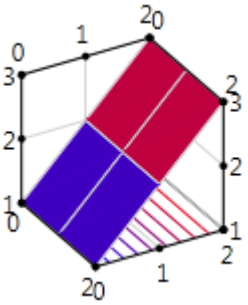
Chart3D for WPF and Silverlight allows you to give your charts a 2D representation by displaying contours and zones on the top, or ceiling, and bottom, or floor, of the plot cube of the chart.



To determine the appearance of the floor, set the FloorAppearance property. In the following example, the XAML sets the **FloorAppearance** of a **SurfaceZone** chart to **Contour**.

XAML

```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="234,132,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" ChartType="SurfaceZone"
FloorAppearance="Contour">
        <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
</c1chart3d:C1Chart3D>
```

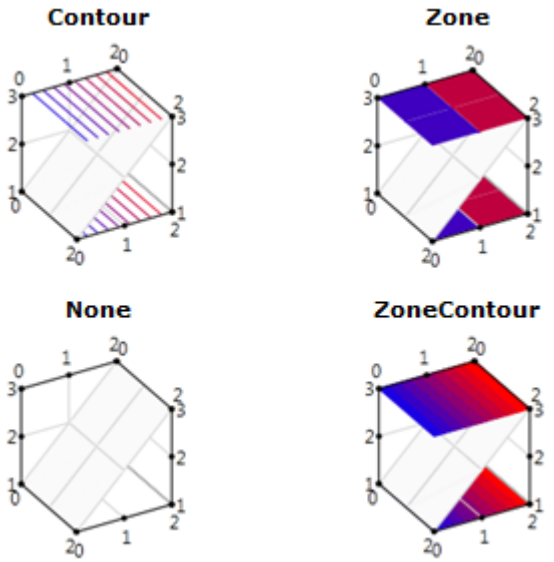


To determine the appearance of the ceiling, set the CeilAppearance property. In the following example, the XAML sets the **CeilAppearance** of a **SurfaceContour** chart to **Zone**.

XAML

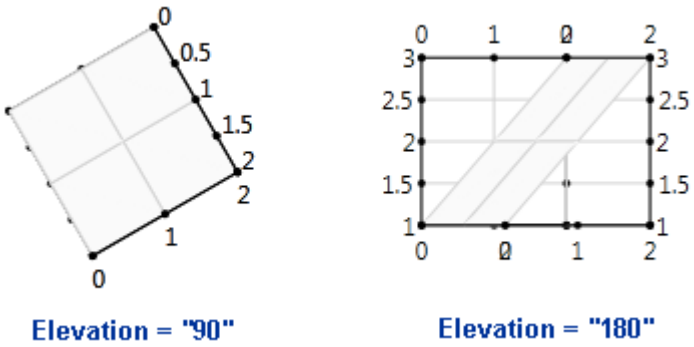
```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="234,132,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" ChartType="SurfaceContour"
CeilAppearance="Zone">
    <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
</c1chart3d:C1Chart3D>
```

The **FloorAppearance** and **CeilAppearance** properties offer four appearance options:



Creating a Two-Dimensional Chart

You can easily create a flat 3D chart, such as a heat map, by changing the Elevation property of a chart. Each chart has an elevation angle, or an incline above the X-axis. When you first add a chart to the window, the elevation angle is 150 degrees, by default. You can flatten the chart by setting the Elevation property to 90 or 180, depending on how you want to flatten it with respect to the X-axis.



The following XAML sets the Elevation property for a chart to 90.

XAML

```
<c1chart3d:C1Chart3D Height="150" HorizontalAlignment="Left" Margin="234,139,0,0"
Name="c1Chart3D1" VerticalAlignment="Top" Width="200" Elevation="90">
     <c1chart3d:GridDataSeries ZDataString="1 1 1,2 2 2,3 3 3" />
</c1chart3d:C1Chart3D>
```

Creating a Custom Axis Annotation

The 3D chart has an X, Y, and Z axis that can be controlled by the C1Chart3D object. You can customize the look and feel of the axes through the C1Chart3D object and then set the X, Y, or Z axis object through the AxisX, AxisY, and

AxisZ properties for the new Axes settings to appear on the X, Y, and Z axes.

This section describes the most common axis configuration scenarios used to make the chart more readable such as labeling, scaling, and formatting.

Custom Axis Annotation Template

The following sample shows how to use the AnnoTemplate property to create axis labels with color depending on the corresponding axis value.

Add the following XAML to your MainWindow.xaml or your MainPage.xaml:

> **Note**: In a Silverlight application, you'll add <Page.Resources> </Page.Resources> tags instead of the <Window.Resources> tags.

XAML

```
<Window.Resources>
    <!-- instance of converter -->
    <local:LabelToColorConvereter x:Key="cnv" />
  </Window.Resources>
    <Grid>
    <c1chart3d:C1Chart3D Name="c1Chart3D1" >
      <c1chart3d:GridDataSeries ZDataString="-1 -1 -1,0 0 0, 1 1 1" />
      <c1chart3d:C1Chart3D.AxisZ>
        <c1chart3d:Axis3D>
          <!-- set axis annotation template  -->
          <c1chart3d:Axis3D.AnnoTemplate>
            <DataTemplate >
              <!-- DataContext is axis label(string), use converter to set color  -->
                <TextBlock Width="20" Height="12" Text="{Binding}"
Foreground="{Binding Converter={StaticResource cnv}}" />
            </DataTemplate>
          </c1chart3d:Axis3D.AnnoTemplate>
        </c1chart3d:Axis3D>
      </c1chart3d:C1Chart3D.AxisZ>
    </c1chart3d:C1Chart3D>
  </Grid>
```

Select **View | Code** and add the following code. The binding converter selects the brush of the axis label depending on the value.

C#

```
  public class LabelToColorConvereter : IValueConverter
  {
    static Brush belowZero = new SolidColorBrush(Colors.Blue);
    static Brush aboveZero = new SolidColorBrush(Colors.Red);
    static Brush zero = new SolidColorBrush(Colors.DarkGray);
    public object Convert(object value, Type targetType, object parameter,
      System.Globalization.CultureInfo culture)
    {
```

```
    // string representing axis label
    string s = value as string;
    if (!string.IsNullOrEmpty(s))
    {
      var dv = double.Parse(s);
      // return brush depending on the value
      if (dv < 0)
        return belowZero;
      else if (dv > 0)
        return aboveZero;
      else
        return zero;
    }
    return value;
  }

  public object ConvertBack(object value, Type targetType, object parameter,
    System.Globalization.CultureInfo culture)
  {
    throw new NotImplementedException();
  }
}
```

Axis Title

Adding a title to an axis clarifies what is charted along that axis. A title with a specified font can be added to any axis.

To Add an Axis Title:

Use the axis Title property to add a title to an axis.

1. In the Visual Studio **Properties** window, expand the **AxisX**, **AxisY**, or **AxisZ** property.
2. Enter a title next to the **Title** property.
3. To remove the title, delete the text.

DirectX rendering

Note: This topic applies only to WPF applications.

The **DirectX Rendering Mode** allows you to create a high performance rendering of surface charts for larger data sets with **Direct3D** rendering. To implement DirectX Rendering, mode, just set the RenderMode property to **Direct3D**.

To use the **Direct3D** render mode in your WPF applications, you should distribute the following assemblies along with your application. During development these assemblies should be found in the same directory as the **C1.WPF.C1Chart3D** assembly but do not need to be directly referenced by the project.

- SharpDX.D3DCompiler.dll

- SharpDX.Direct3D9.dll
- SharpDX.Direct3D10.dll
- SharpDX.dll
- SharpDX.DXGI.dll

System Requirements for **Direct3D Rendering** include:

- DirectX Version 10.0 or higher
- D3D9 Overlay Support

The **Direct3D** mode does not work on some virtual machines (Hyper-V) and through remote desktop connections since the required hardware layer is not supported in such cases.