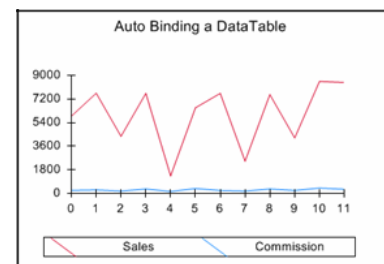
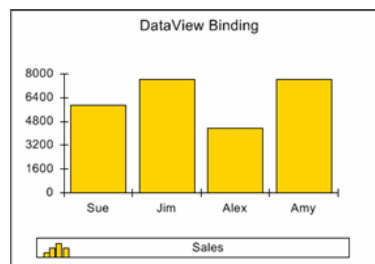


How to Generate Charts and Graphs using Visual Studio Express

It's no secret that displaying lots of raw numbers to people can quickly lead to confusion and frustration. If you are familiar with Microsoft Office or any other office productivity suite, you have probably seen charts and graphs used to convey data in a clear and concise manner. There are many different types of chart including bar, line, pie, scatter, and radar. Each chart type has strengths and weaknesses that need to be considered when determining the best way to represent a set of data. In this article, we will take a look at how you can easily add simple charts and graphs to your own applications using the Graphics Server .NET charting controls.



Contents

[Graphics Server .NET setup and configuration](#)


[Designing Charts](#)

[Programmatically using the Chart control in a Windows Form application](#)

[References and additional resources](#)

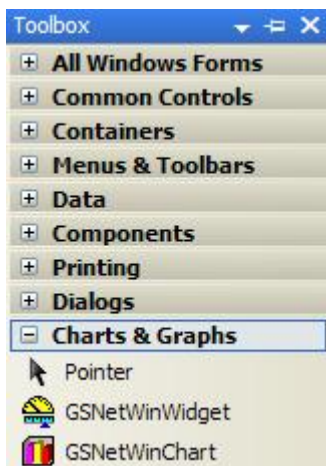
Graphics Server .NET setup and configuration

Your first task is to download and install the Graphics Server .NET software. The Express edition of GS .NET is made available to you when you register your Visual Studio Express product. The registration process can be initiated by selecting **Help | Register Product** from the main menu of your Express product. After you register, you will receive an email from Microsoft with a link to the registration benefits portal site where you will find the GS .NET download. Downloading and installing the GS .NET software is quick and easy.

Graphics Server .NET	Utility	 <p>Graphics Server .NET Experience Interactive Chart Design with Graphics Server .NET. Learn how fast and easy it is to design a chart and integrate it with your Visual Studio 2005 application with our new Interactive Designer. Professional looking results, fast development and reasonable prices for both Win and Web Forms.</p>
--------------------------------------	---------	---

Screenshot - The Registration Benefits Portal site lists your benefits and provides download links

After installation, follow the instructions in the “Add Controls to Toolbox” text file to add the chart and graph controls to your VS Express design toolbox. You can find these instructions by selecting **Start | All Programs | Graphics Server Technologies | Graphics Server .NET 3.0 for Express**. If you use Visual Web Developer you should see the **GSNetWebChart** and **GSNetWebWidget** controls in the Toolbox. For Visual Basic, C#, and the other Express products you should now have the **GSNetWinChart** and **GSNetWinWidget** controls in your Toolbox.



Screenshot - Windows Form Controls

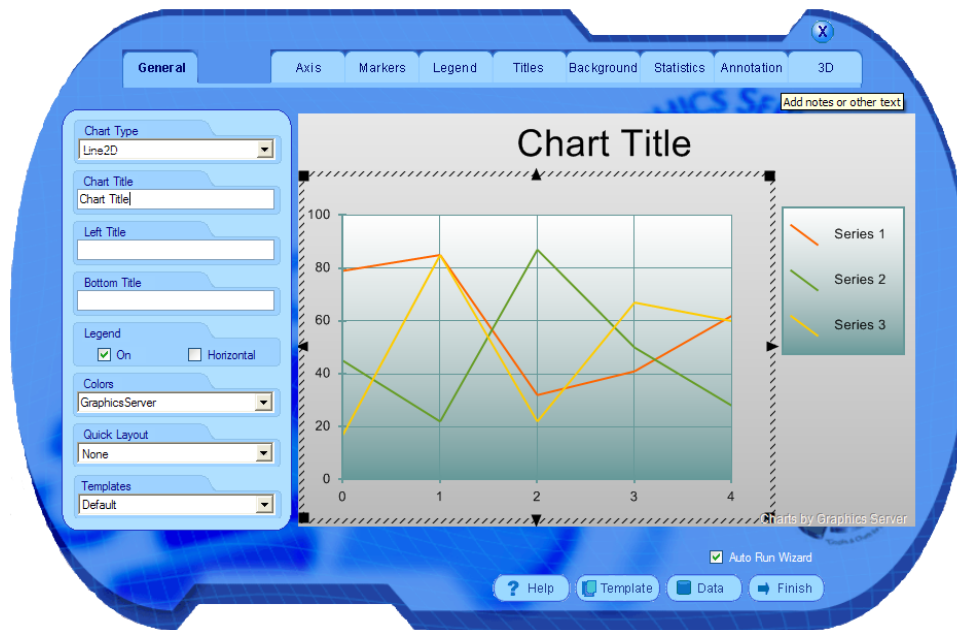


Screenshot - Web Form Controls

Note: The Toolbox will only list controls when a designer view is open.

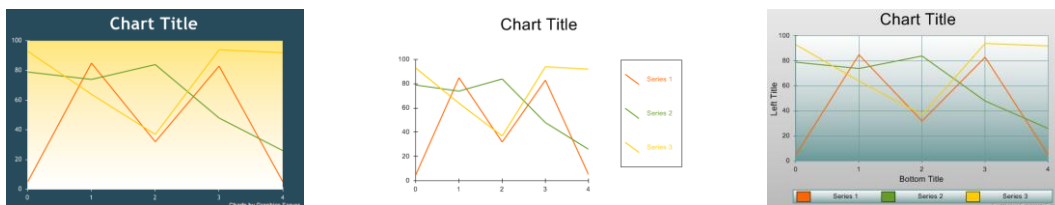
Designing Charts

Graphics Server .NET includes a wizard to help with the rapid creation of charts and graphs. It can be used in a standalone fashion or within the Visual Studio development environment. The wizard allows you to modify chart properties such as type, axis, legend, titles, backgrounds, and so on.



Screenshot - Chart Wizard designer in Visual Studio Express

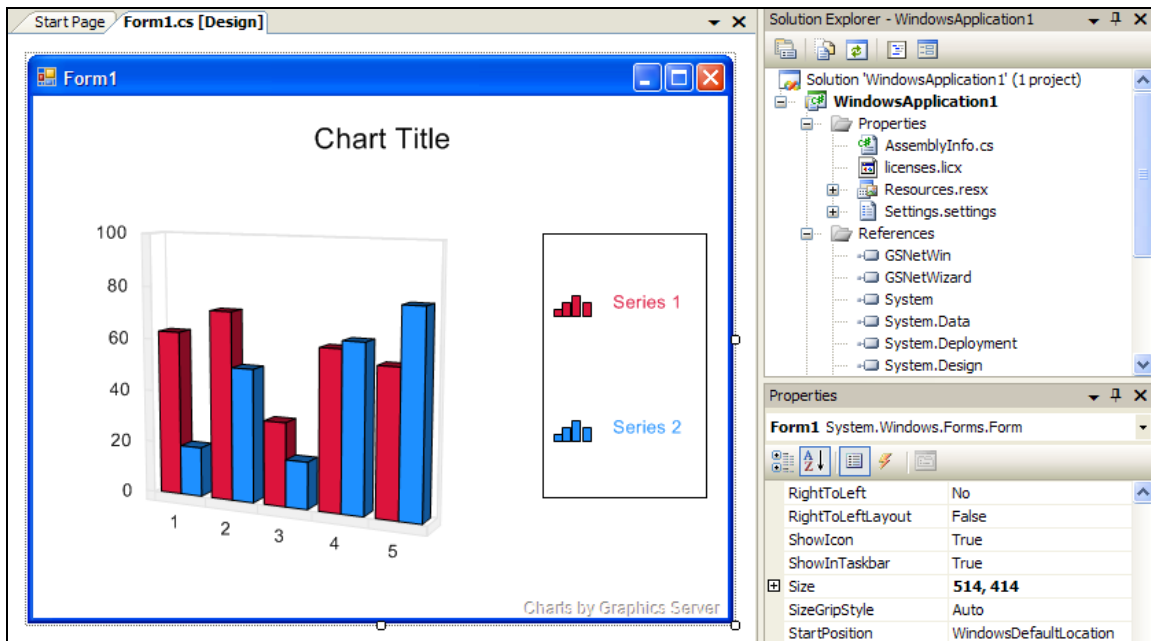
There are some slight differences between the wizard that runs stand alone and the one that runs as part of Express. The stand alone version is used to design chart templates and experiment with available properties. If you click the **Template** button you can load a number of pre-defined templates that change the properties of the currently displayed chart.



Screenshot - Charts can be customized using templates

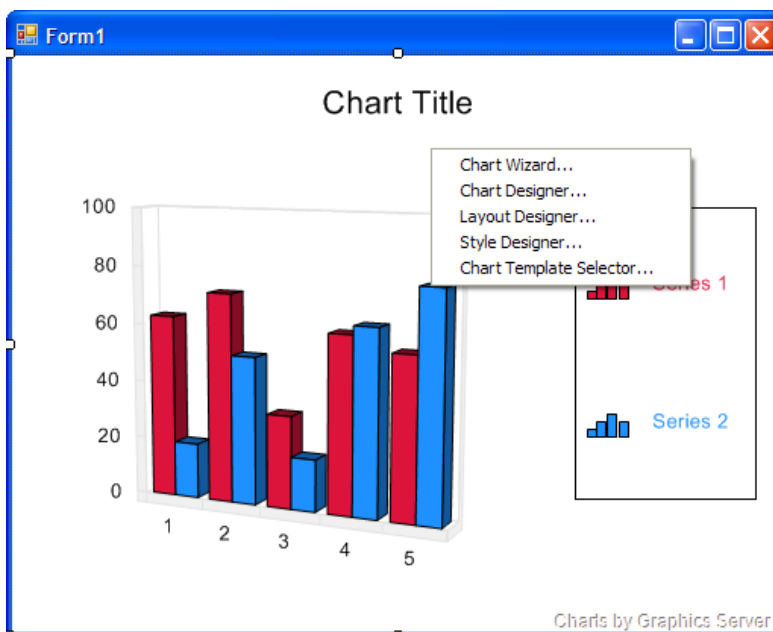
The stand alone Chart Wizard does not have the ability to bind to different data sources other than the simple test data that can be modified by clicking on the **Data** button. Keep in mind that the Graphics Server .NET software package is meant to be used by software developers, not end users.

Working with the Chart Wizard and other tools in the Express development environment provides developers with fine-grained control over **Chart** and **Widget** controls. In the following screenshot, an instance of the **GSNetWinChart** control has been added to a new Windows Form by dragging it from the toolbox. Some default test data is also applied to the chart to make the design process easier.



Screenshot - Chart control added to a Form

To access the **Chart Designer** or one of the other designers within Visual Studio, right-click on the chart control instance within the Windows Form. This will show a context menu as shown below.



Screenshot - Right-click on chart control to access context menu

If you select the Chart Wizard and then click on the Data button within the Express development environment, you can setup a database connection and select statement in order to get data for the chart.

Provide Data Source

☒ Connect to database ☐ Manually enter data

OLEDB Connection String:

Select Statement:

Chart Series

Series 1
Series 2

Data

X Value:

Y Value:

Z Value:

High Value:

Low Value:

Close Value:

Open Value:

Angle Value:

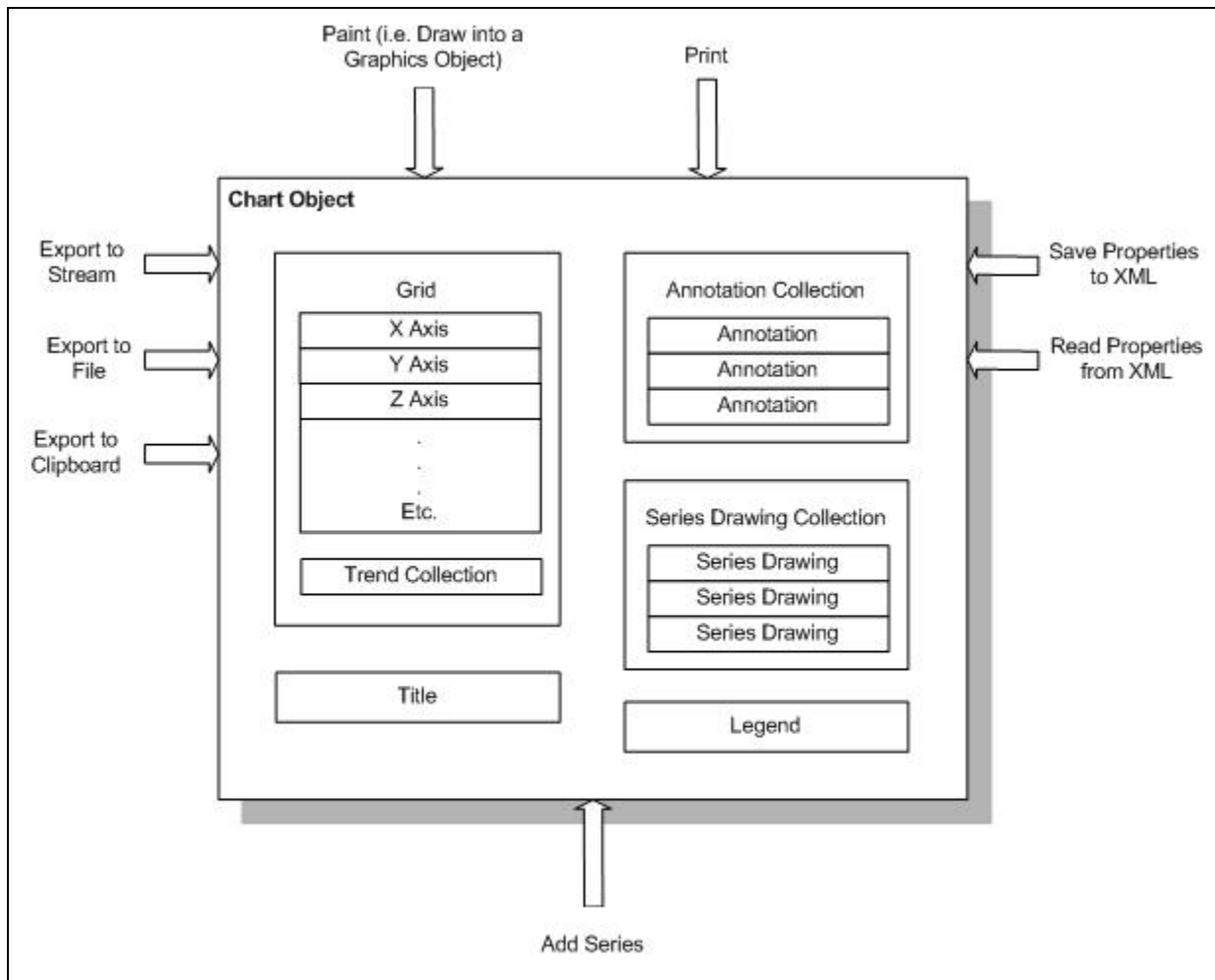
Radius Value:

X-Axis Labels:

Screenshot - Configuring the data source for the chart

Programmatically using the Chart control in a Windows Form Application

So far we have introduced Graphics Server .NET and looked at the design-time support provided to developers. Now let's take a moment to examine the capabilities of the charting class library. The following diagram gives you a good overview of what the object model looks like as well as some of the capabilities of the charting engine.



The GS .NET charting engine is entirely contained within the **Chart** class, which can be accessed through the `GSNetWinChart.Chart` property in code. For example, to change the chart type to a two-dimensional bar we can use the following C# code:

```
this.gsNetWinChart1.Chart.ChartType = ChartType.Bar2D;
```

As we saw earlier, it is possible to save and load template files containing chart properties. This provides a built-in and convenient method by which you can adjust the charts that are produced. The chart properties can be saved to an XML file using the **Chart.WriteChartProperties** method. Other capabilities include data binding, creating arbitrarily high resolution charts, printing, scrolling, and more.

Here is some example code that shows how to work with a `GSNetWinChart` control in C#:

```
this.gsNetWinChart1.Chart.ChartType = ChartType.Bar2D;
this.gsNetWinChart1.Chart.ChartTitle.Text = "Hello Charting World!";

int[] myArrayData = {1, 16, 5, 9, 12};
ArrayDataProvider dataProvider = new ArrayDataProvider(myArrayData);
```

```

Series mySeries = new Series("My Data");
mySeries.BindComponent(SeriesComponent.Y, dataProvider, "My Field");

this.gsNetWinChart1.Chart.RemoveAllSeries();
this.gsNetWinChart1.Chart.AddSeries(mySeries);

```

And for VB.NET:

```

Me.GsNetWinChart1.Chart.ChartType = ChartType.Bar2D
Me.GsNetWinChart1.Chart.ChartTitle.Text = "Hello Charting World!"

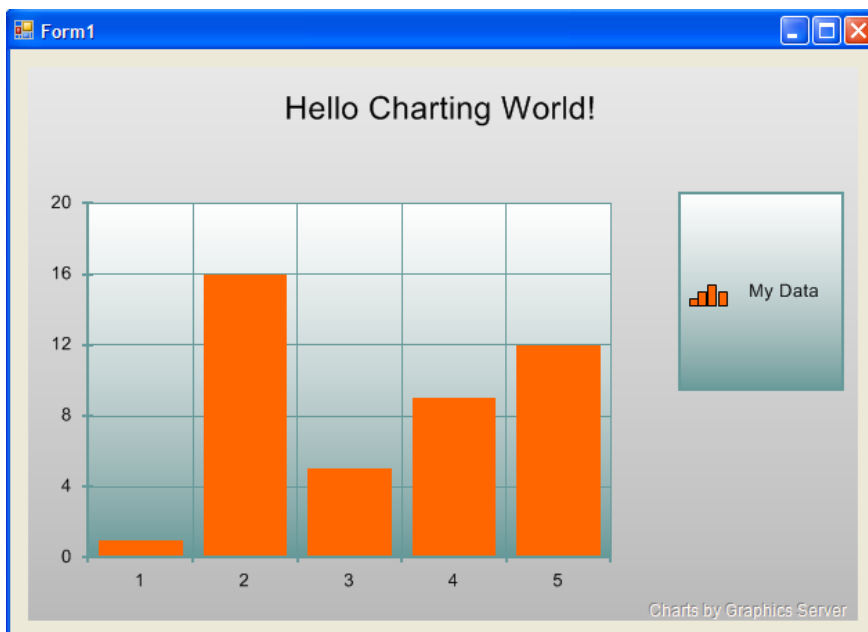
Dim myArrayData As Integer() = {1, 16, 5, 9, 12}
Dim dataProvider As ArrayDataProvider = New
ArrayDataProvider(myArrayData)

Dim mySeries As Series = New Series("My Data")
mySeries.BindComponent(SeriesComponent.Y, dataProvider, "My Field")

Me.GsNetWinChart1.Chart.RemoveAllSeries()
Me.GsNetWinChart1.Chart.AddSeries(mySeries)

```

The example code produces the following chart show below:



Screenshot - 2D bar graph of data provided by an array of integer values

Note: charts controls can be used with Visual Web Developer Express as well.

References and additional resources

For further documentation, sample code, and product information visit:

<http://www.graphicsserver.com>