Getting Started with Measurement Studio™

Support for Visual Studio 2005 and Visual Studio 2008

November 2010
373392E-01

NATIONAL INSTRUMENTS™

Worldwide Technical Support and Product Information

ni.com

National Instruments Corporate Headquarters

11500 North Mopac Expressway Austin, Texas 78759-3504 USA Tel: 512 683 0100

Worldwide Offices

Australia 1800 300 800, Austria 43 662 457990-0, Belgium 32 (0) 2 757 0020, Brazil 55 11 3262 3599, Canada 800 433 3488, China 86 21 5050 9800, Czech Republic 420 224 235 774, Denmark 45 45 76 26 00, Finland 358 (0) 9 725 72511, France 01 57 66 24 24, Germany 49 89 7413130, India 91 80 41190000, Israel 972 3 6393737, Italy 39 02 41309277, Japan 0120-527196, Korea 82 02 3451 3400, Lebanon 961 (0) 1 33 28 28, Malaysia 1800 887710, Mexico 01 800 010 0793, Netherlands 31 (0) 348 433 466, New Zealand 0800 553 322, Norway 47 (0) 66 90 76 60, Poland 48 22 328 90 10, Portugal 351 210 311 210, Russia 7 495 783 6851, Singapore 1800 226 5886, Slovenia 386 3 425 42 00, South Africa 27 0 11 805 8197, Spain 34 91 640 0085, Sweden 46 (0) 8 587 895 00, Switzerland 41 56 2005151, Taiwan 886 02 2377 2222, Thailand 662 278 6777, Turkey 90 212 279 3031, United Kingdom 44 (0) 1635 523545

For further support information, refer to the *Technical Support and Professional Services* appendix. To comment on National Instruments documentation, refer to the National Instruments Web site at ni.com/info and enter the Info Code feedback.

Important Information

Warranty

The media on which you receive National Instruments software are warranted not to fail to execute programming instructions, due to defects in materials and workmanship, for a period of 90 days from date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace software media that do not execute programming instructions if National Instruments receives notice of such defects during the warranty period. National Instruments does not warrant that the operation of the software shall be uninterrupted or error free.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. National Instruments will pay the shipping costs of returning to the owner parts which are covered by warranty.

National Instruments believes that the information in this document is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, National Instruments reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult National Instruments if errors are suspected. In no event shall National Instruments be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, NATIONAL INSTRUMENTS MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF NATIONAL INSTRUMENTS SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. NATIONAL INSTRUMENTS WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of National Instruments will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against National Instruments must be brought within one year after the cause of action accrues. National Instruments shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow the National Instruments installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

Copyright

Trademarks

CVI, LabVIEW, National Instruments, NI, ni.com, the National Instruments corporate logo, and the Eagle logo are trademarks of National Instruments Corporation. Refer to the *Trademark Information* at `ni.com/trademarks` for other National Instruments trademarks.

The mark LabWindows is used under a license from Microsoft Corporation. Windows is a registered trademark of Microsoft Corporation in the United States and other countries. Other product and company names mentioned herein are trademarks or trade names of their respective companies. Other product and company names mentioned herein are trademarks or trade names of their respective companies.

Members of the National Instruments Alliance Partner Program are business entities independent from National Instruments and have no agency, partnership, or joint-venture relationship with National Instruments.

Patents

For patents covering National Instruments products/technology, refer to the appropriate location: **Help»Patents** in your software, the `patents.txt` file on your media, or the *National Instruments Patent Notice* at `ni.com/patents`.

WARNING REGARDING USE OF NATIONAL INSTRUMENTS PRODUCTS

(1) NATIONAL INSTRUMENTS PRODUCTS ARE NOT DESIGNED WITH COMPONENTS AND TESTING FOR A LEVEL OF RELIABILITY SUITABLE FOR USE IN OR IN CONNECTION WITH SURGICAL IMPLANTS OR AS CRITICAL COMPONENTS IN ANY LIFE SUPPORT SYSTEMS WHOSE FAILURE TO PERFORM CAN REASONABLY BE EXPECTED TO CAUSE SIGNIFICANT INJURY TO A HUMAN.

(2) IN ANY APPLICATION, INCLUDING THE ABOVE, RELIABILITY OF OPERATION OF THE SOFTWARE PRODUCTS CAN BE IMPAIRED BY ADVERSE FACTORS, INCLUDING BUT NOT LIMITED TO FLUCTUATIONS IN ELECTRICAL POWER SUPPLY, COMPUTER HARDWARE MALFUNCTIONS, COMPUTER OPERATING SYSTEM SOFTWARE FITNESS, FITNESS OF COMPILERS AND DEVELOPMENT SOFTWARE USED TO DEVELOP AN APPLICATION, INSTALLATION ERRORS, SOFTWARE AND HARDWARE COMPATIBILITY PROBLEMS, MALFUNCTIONS OR FAILURES OF ELECTRONIC MONITORING OR CONTROL DEVICES, TRANSIENT FAILURES OF ELECTRONIC SYSTEMS (HARDWARE AND/OR SOFTWARE), UNANTICIPATED USES OR MISUSES, OR ERRORS ON THE PART OF THE USER OR APPLICATIONS DESIGNER (ADVERSE FACTORS SUCH AS THESE ARE HEREAFTER COLLECTIVELY TERMED "SYSTEM FAILURES"). ANY APPLICATION WHERE A SYSTEM FAILURE WOULD CREATE A RISK OF HARM TO PROPERTY OR PERSONS (INCLUDING THE RISK OF BODILY INJURY AND DEATH) SHOULD NOT BE RELIANT SOLELY UPON ONE FORM OF ELECTRONIC SYSTEM DUE TO THE RISK OF SYSTEM FAILURE. TO AVOID DAMAGE, INJURY, OR DEATH, THE USER OR APPLICATION DESIGNER MUST TAKE REASONABLY PRUDENT STEPS TO PROTECT AGAINST SYSTEM FAILURES, INCLUDING BUT NOT LIMITED TO BACK-UP OR SHUT DOWN MECHANISMS. BECAUSE EACH END-USER SYSTEM IS CUSTOMIZED AND DIFFERS FROM NATIONAL INSTRUMENTS' TESTING PLATFORMS AND BECAUSE A USER OR APPLICATION DESIGNER MAY USE NATIONAL INSTRUMENTS PRODUCTS IN COMBINATION WITH OTHER PRODUCTS IN A MANNER NOT EVALUATED OR CONTEMPLATED BY NATIONAL INSTRUMENTS, THE USER OR APPLICATION DESIGNER IS ULTIMATELY RESPONSIBLE FOR VERIFYING AND VALIDATING THE SUITABILITY OF NATIONAL INSTRUMENTS PRODUCTS WHENEVER NATIONAL INSTRUMENTS PRODUCTS ARE INCORPORATED IN A SYSTEM OR APPLICATION, INCLUDING, WITHOUT LIMITATION, THE APPROPRIATE DESIGN, PROCESS AND SAFETY LEVEL OF SUCH SYSTEM OR APPLICATION.

Contents

Chapter 4
Measurement Studio Integrated Tools and Features

Appendix A
Technical Support and Professional Services

Index

About This Manual

Getting Started with Measurement Studio introduces the concepts associated with the Measurement Studio class libraries and development tools. This manual assumes that you have a general working knowledge of Microsoft Visual Studio and the .NET Framework for .NET application development.

How to Use This Manual

Measurement Studio 2010 includes one DVD with support for Visual Studio 2005, Visual Studio 2008, and Visual Studio 2010. This manual documents support for Visual Studio 2005 and Visual Studio 2008.

Note Measurement Studio 2009 was the last version of Measurement Studio to provide support for Visual Studio 6.0, Visual Studio 2003, and Visual C++ MFC. If you want to continue building or developing applications using these legacy environments/languages, you can use the NI Measurement Studio for Legacy Environments/Languages product. For additional information, refer to the Measurement Studio Release Notes by selecting **Start»All Programs»National Instruments»<Measurement Studio>»Release Notes**. To request NI Measurement Studio for Legacy Environments/Languages, please contact a Technical Sales Representative at `ni.com/contact`.

Getting Started with Measurement Studio is organized into four chapters. Chapter 1, *Introduction to Measurement Studio*, is an overview of Measurement Studio. Chapter 1 contains installation and deployment requirements, installation instructions, and a list of Measurement Studio resources. Chapter 2, *Creating Applications with Measurement Studio*, provides walkthroughs that guide you through step-by-step instructions on how to develop with Measurement Studio features. Chapter 3, *Measurement Studio .NET Class Libraries*, and Chapter 4, *Measurement Studio Integrated Tools and Features*, describe the features and functionality of the Measurement Studio class libraries and the tools integrated into the Visual Studio environment.

Use this manual as a starting point to learn about Measurement Studio. Refer to the *NI Measurement Studio Help* within the Visual Studio environment for function reference and detailed information about the Measurement Studio class libraries, wizards, assistants, and other features.

Conventions

The following conventions appear in this manual:

<> — Text enclosed in angle brackets represents directory names and parts of paths that may vary on different computers, such as <Windows\System>.

» — The » symbol leads you through nested menu items and dialog box options to a final action. The sequence **Options»Settings»General** directs you to pull down the **Options** menu, select the **Settings** item, and select **General** from the last dialog box.

 — This icon denotes a tip, which alerts you to advisory information.

 — This icon denotes a note, which alerts you to important information.

bold — Bold text denotes items that you must select or click on in the software, such as menu items and dialog box options. Bold text also denotes class library member names or emphasis.

italic — Italic text denotes parameters, variables, cross-references, or an introduction to a key concept. Italic text also denotes text that is a placeholder for a word or value that you must supply.

`monospace` — Text in this font denotes text or characters that you enter from the keyboard, sections of code, programming examples, and syntax examples. This font also is used for the proper names of disk drives, paths, directories, programs, device names, filenames and extensions, and code excerpts.

1

Introduction to Measurement Studio

Measurement Studio is an integrated suite of tools and class libraries that is designed for developers using Microsoft Visual Basic .NET and Visual C# to develop measurement and automation applications.

Note This getting started guide documents Measurement Studio support for Visual Studio 2005 and Visual Studio 2008. If you have Measurement Studio 2010 support for Visual Studio 2010 installed, you can view *Getting Started with Measurement Studio for Visual Studio 2010* support by selecting **Start»All Programs»National Instruments»<Measurement Studio>»Getting Started with Measurement Studio**. If you do not have Measurement Studio 2010 support for Visual Studio 2010 installed, you can view the *Getting Started with Measurement Studio: Support for Visual Studio 2010* online at `ni.com/manuals`.

What Is Measurement Studio?

Measurement Studio is an integrated suite of tools and class libraries that are designed for developers using Microsoft .NET technologies to develop measurement and automation Windows and Web applications.

Note For additional Measurement Studio evaluation information, such as videos, an ASP.NET demo gallery, and a user interface control feature tour, refer to `ni.com/mstudio/try`. To learn about what's new in the current version of Measurement Studio, refer to the *What's New* section in the *Measurement Studio Release Notes*. Select **Start»All Programs»National Instruments»<Measurement Studio>»Release Notes** to view the release notes.

Why Should I Use Measurement Studio?

Measurement Studio dramatically reduces application development time through object-oriented measurement hardware interfaces, advanced analysis libraries, scientific user interface controls for Windows and Web applications, measurement data networking, wizards, interactive code designers, and highly extensible .NET classes. You can use Measurement Studio to develop a complete measurement and automation application that includes data acquisition, analysis, and presentation functionalities.

Measurement Studio Edition Comparison Chart

The following table lists the features included in the Standard, Professional, and Enterprise editions of Measurement Studio. Refer to `ni.com/mstudio` for more information about the functionality and features included with each Measurement Studio edition.

Table 1-1. Measurement Studio Edition Comparison Chart for Visual C# and Visual Basic .NET

Feature	Standard Edition	Professional Edition	Enterprise Edition
Project Wizards	✓	✓	✓
Windows Forms User Interface Controls	✓	✓	✓
Standard Analysis Libraries[1]	✓	✓	✓
NI-488.2 Class Libraries[2]	✓	✓	✓
NI-VISA Class Libraries[2]	✓	✓	✓
NI-DAQmx Class Libraries[2]	✓	✓	✓
.NET Instrument Driver Wizard	✓	✓	✓
Web Forms User Interface Controls		✓	✓
Professional Analysis Libraries[3]		✓	✓
3D Graph ActiveX Control		✓	✓
Network Variable Class Library		✓	✓
Network Variable Data Source		✓	✓
DataSocket Server		✓	✓
DataSocket Library		✓	✓

Table 1-1. Measurement Studio Edition Comparison Chart for Visual C# and Visual Basic .NET (Continued)

Feature	Standard Edition	Professional Edition	Enterprise Edition
Parameter Assistant		✓	✓
Instrument I/O Assistant[2]		✓	✓
DAQ Assistant[2]		✓	✓
Enterprise Analysis Libraries[4]			✓
LabWindows™/CVI™ Full Development System (FDS)			✓

[1] Refer to the *Standard Analysis* section of Chapter 3, *Measurement Studio .NET Class Libraries*, for a list of the functionality included in the Standard Analysis class library.

[2] Included with the NI Device Drivers DVD.

[3] Refer to the *Professional Analysis* section of Chapter 3, *Measurement Studio .NET Class Libraries*, for a list of the functionality included in the Professional Analysis class library.

[4] Refer to the *Enterprise Analysis* section of Chapter 3, *Measurement Studio .NET Class Libraries*, for a list of the functionality included in the Enterprise Analysis class library.

Activating Measurement Studio

After you install Measurement Studio, you must use the NI Activation Wizard to activate the software. To activate Measurement Studio, you need the serial number printed on the Certificate of Ownership included in your software kit. Refer to the *Licensing, Evaluation, and Activation* topic in the *NI Measurement Studio Help* for more information about how to activate Measurement Studio. Measurement Studio 2010 is the first version of Measurement Studio to use activation.

National Instruments offers a variety of Measurement Studio licenses, with certain licensed features available for each license type: Standard Edition, Professional Edition, and Enterprise Edition. Refer to the *Measurement Studio Editions* topic in the *NI Measurement Studio Help* for more information on Measurement Studio editions, and the *Licensing Measurement Studio* topic in the *NI Measurement Studio Help* for more information about licensing Measurement Studio.

Note To move to a different Measurement Studio edition, you can activate the new edition by using NI License Manager. Select **Start»All Programs»National Instruments» NI License Manager** to access NI License Manager

For general license activation information, refer to `ni.com/activate`. Refer to `ni.com/mstudio` to purchase a Measurement Studio license. Contact a local National Instruments representative at `ni.com/contact` for more information or for questions about specific licensing needs.

Driver Support

To use .NET class libraries that interface to National Instruments device drivers, NI-DAQmx, NI-VISA, NI-488.2, NI-SCOPE, NI-IMAQ, NI-IMAQdx, and the MAX (Measurement & Automation Explorer) configuration utility, you must install the underlying device drivers and the .NET class libraries. You can install the device drivers and the .NET class libraries from the NI Device Drivers DVD included with Measurement Studio.

Refer to *NI Drivers and Updates* on `ni.com` and enter `Device Drivers` into the search field to download the latest version of the NI Device Drivers.

Driver Support Notes

Refer to the following sections for information about .NET support for NI device drivers.

- The DAQ Assistant and the Instrument I/O Assistant are installed from the NI Device Drivers DVD. You must install the NI Device Drivers DVD to use the assistants.
- 64-bit support is available in class libraries for Visual Studio 2008 for the following drivers: NI-DAQmx 8.9.5 and later, NI-VISA 4.5.1 and later, NI-488.2 2.7.1 and later, and MAX 4.6 and later.
- 64-bit support is not available in class libraries for Visual Studio 2005.
- Visual C++ MFC 2005 and Visual C++ MFC 2008 support:
  - The following drivers versions are the last that include full feature support for Visual C++ MFC 2005 and Visual C++ MFC 2008: NI-DAQmx 9.2, NI-VISA 5.0.2, and NI-488.2 2.8.
  - The following drivers do not provide support for Visual C++ MFC 2005 and Visual C++ MFC 2008: NI-IMAQ, NI-IMAQdx, MAX (Measurement & Automation Explorer), and NI-SCOPE.
- To download NI-SCOPE .NET class libraries, refer to NI-SCOPE .NET Driver Support on `ni.com`.

Deploying Measurement Studio Applications

To deploy an application built with Measurement Studio .NET class libraries, the target computer must have a Windows 7/Vista/XP/Server 2003 and 2008 (R2 editions) operating system and the .NET Framework version 3.5 for Visual Studio 2008 or the .NET Framework version 2.0 for Visual Studio 2005.

Deploying 64-bit Applications

To facilitate use in Visual Studio Setup projects, all Measurement Studio class libraries that support 64-bit include both 32-bit and 64-bit deployment merge modules. This is true regardless of whether the class library includes platform-specific (i.e., x86 or x64) or platform-agnostic (i.e., Any CPU) assemblies. Refer to *Deploying Windows Applications* in the *NI Measurement Studio Help* for more information on using 64-bit merge modules.

Learning Measurement Studio

As you work with Measurement Studio, you might need to consult additional resources. For detailed Measurement Studio help, including function reference and in-depth documentation on developing with Measurement Studio, refer to the *NI Measurement Studio Help* within the Visual Studio environment. The *NI Measurement Studio Help* is fully integrated with the Visual Studio help. You must have Visual Studio installed to view the online help, and you must have the Microsoft .NET Framework SDK 2.0 for Visual Studio 2005 or the Microsoft .NET Framework SDK 3.5 for Visual Studio 2008 installed in order for links from Measurement Studio help topics to .NET Framework help topics to work.

You can launch the *NI Measurement Studio Help* in the following ways:

- From the Windows Start menu, select **Start»All Programs»National Instruments»<Measurement Studio>»Measurement Studio Documentation**.
- In Visual Studio 2005 and 2008, select **Help»Contents** to view the Visual Studio table of contents. The *NI Measurement Studio Help* is integrated into the Visual Studio help.
- In Visual Studio, select **Measurement Studio»NI Measurement Studio Help**.

Tip As you work through this manual, you will see italicized references to relevant help topics. To find these topics, use the table of contents in the *NI Measurement Studio Help* installed on your machine.

The following resources also are available to provide you with information about Measurement Studio.

- Getting Started information—Refer to the *Measurement Studio Core Overview* topic for an introduction to Measurement Studio. For a list of Measurement Studio resources, refer to the *Using the Measurement Studio Help* topic in the *NI Measurement Studio Help*.
- Examples—Measurement Studio installs examples organized by class library, depending on the component, the version of Visual Studio or the .NET Framework that the example supports, the version of Measurement Studio installed on the system, and the operating system. For more information on example locations, refer to the *Where to Find Examples* topic in the *NI Measurement Studio Help*.
- NI Technical Support—Refer to Appendix A, *Technical Support and Professional Services*, for more information.
- Measurement Studio Web site, `ni.com/mstudio`—Contains Measurement Studio news, support, downloads, white papers, and product tutorials.
- NI Developer Zone, `zone.ni.com`—Provides access to online example programs, tutorials, technical news, and Measurement Studio discussion forums.
- *Measurement Studio .NET Class Hierarchy Chart*—Provides overviews of class relationships within class libraries. Chart is included with all Measurement Studio packages and is posted online at `ni.com/manuals`.
- Review the information from the Microsoft Web site on using Visual Studio.

2

Creating Applications with Measurement Studio

The following sections include overview information and step-by-step instructions on developing applications with Measurement Studio tools and features. Refer to the *Developing Projects with Measurement Studio* section of the *NI Measurement Studio Help* for more information about the functionality of these tools and features.

Use the following walkthroughs to help you develop Measurement Studio applications:

- *Walkthrough: Creating a Measurement Studio Application with Windows Forms Controls and Analysis*
- *Walkthrough: Creating a Measurement Studio Application with Web Forms Controls and Analysis*
- *Walkthrough: Creating a Measurement Studio Application with Windows Forms Controls and Network Variable*
- *Walkthrough: Creating a Measurement Studio Application with Windows Forms Controls and Network Variable*
- *Walkthrough: Creating a Measurement Studio NI-DAQmx Application*
- *Walkthrough: Creating a Measurement Studio Instrument I/O Application*

Walkthrough: Creating a Measurement Studio Application with Windows Forms Controls and Analysis

Note To complete this walkthrough, you must have either the Measurement Studio Professional or Measurement Studio Enterprise edition installed. This walkthrough will not work with the Measurement Studio Standard edition.

Measurement Studio includes user interface controls, such as a waveform graph control and a gauge control, and analysis functionality, such as signal

generation and mathematical functions. This walkthrough is designed to help you learn how to add analysis and presentation functionality to a Windows Forms application by taking you through the following steps:

- **Setting up the project**—Using the Measurement Studio Application Wizard, you will create a new project that references the Measurement Studio Analysis class library and Windows Forms controls.
- **Adding user interface controls to the project**—Using the Toolbox, smart tags, and the Properties window, you will add and configure a button, waveform graph, legend, gauge, and numeric edit user interface control.
- **Generating, plotting, and analyzing the data**—Using `NationalInstruments.Analysis.SignalGeneration.WhiteNoiseSignal` and `NationalInstruments.Analysis.Math.Statistics.Mean`, you will generate data, plot the generated data on a waveform graph, and calculate the mean of the data.
- **Customizing the user interface**—Using smart tags and the Collection Editor and Auto Format dialog boxes, you will display the mean value on the gauge and the numeric edit, as well as customize your user interface.

Before you begin

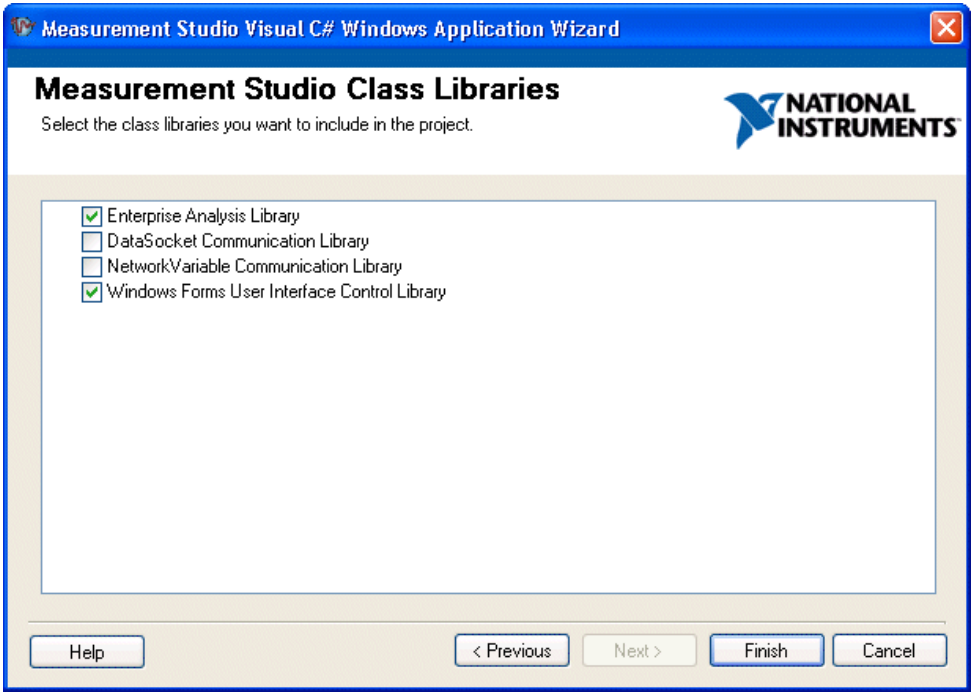
The following components are required to complete this walkthrough:

- Microsoft Visual Studio 2005 or Visual Studio 2008
- Measurement Studio 8.0.1 or later (Professional or Enterprise edition) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise edition) for Visual Studio 2008

Setting up the project

1. Select **File»New»Project**. The New Project dialog box launches.
2. In the Project Types pane, select **Measurement Studio** under Visual C# or Visual Basic, depending on which language you want to create the project in.
3. In the Templates pane, select **NI Windows Application**. Specify `MyMeasurementStudioProject` for **Name** and specify a **Location** of your choice.
4. Click **OK**. The Measurement Studio Application Wizard launches.

5. Select **Analysis Library** and **Windows Forms User Interface Control Library**.



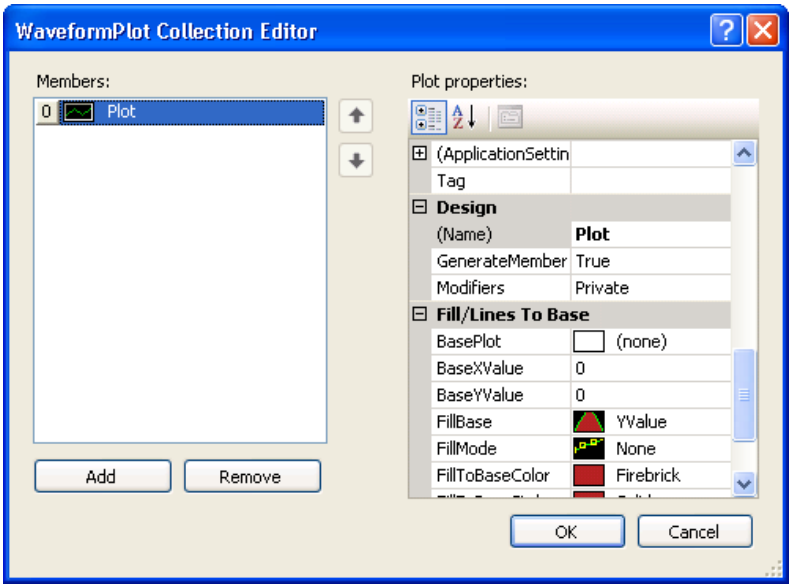
Tip If you are working with an existing project, you can access the Add/Remove Class Libraries wizard by selecting **Measurement Studio»Add/Remove .NET Class Libraries**.

6. Click **Finish** to display `Form1` in the Windows Forms Designer.

Adding user interface controls to the project

1. Select **View»Toolbox** to display the Toolbox. The Toolbox contains components and controls that you can add to your project.
2. Expand the **All Windows Forms** group. The All Windows Forms group contains controls and components included in the `System.Windows.Forms` namespace.
3. Select the **Button** control and drag and drop it onto the form.
4. Right-click the button and select **Properties** to display the Properties window. You configure the properties of the control in the Properties window.

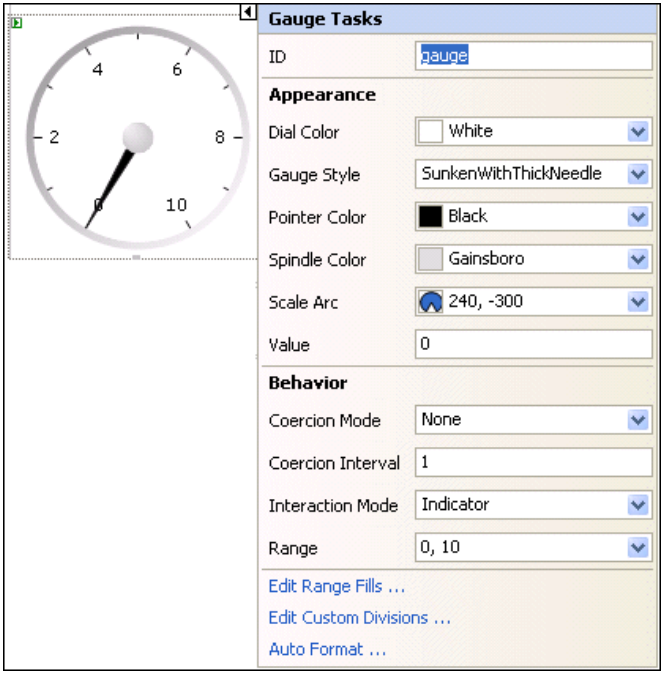
5. The Text property will be highlighted. Type `Start` for the button text.
6. Expand the **Measurement Studio** group in the Toolbox.
7. Select the **WaveformGraph** control and drag and drop it onto the form.
8. Right-click the waveform graph and select **Edit Plots** to display the WaveformPlot Collection Editor dialog box. You use the WaveformPlot Collection Editor dialog box to add or remove plots and to configure plot properties.



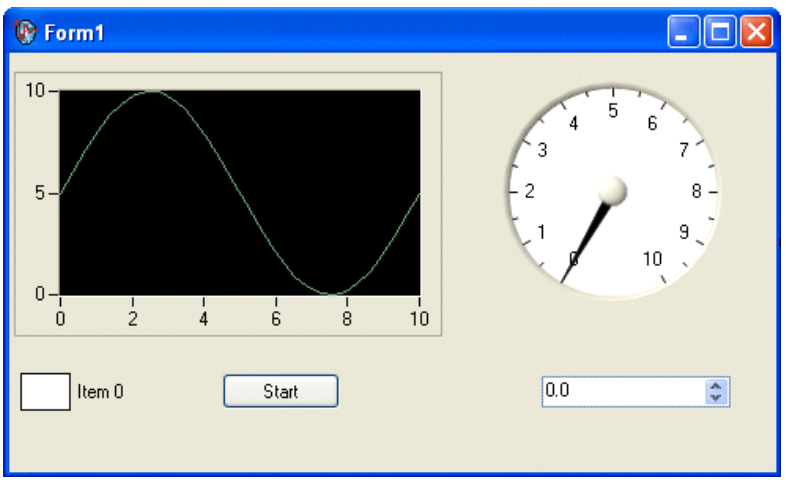
Note You can also access the WaveformPlot Collection Editor dialog box by clicking the waveform graph smart tag. To access the smart tag, left click on the control to select it and then left click on the arrow button in the upper right corner of the control.

9. Type `Plot` for the Name. Click **OK**.
10. Before you add the Measurement Studio legend, numeric edit, and gauge controls, you need to resize the form to accommodate them. Select the form and use the double-sided arrow to resize it.
11. Select the **Legend** control and drag and drop it onto the form.
12. Select the **NumericEdit** control and drag and drop it onto the form.
13. Select the **Gauge** control and drag and drop it onto the form.

14. Click the gauge smart tag to display the Gauge Tasks.



15. Type `gauge` for the name of the gauge.



Generating, plotting, and analyzing the data

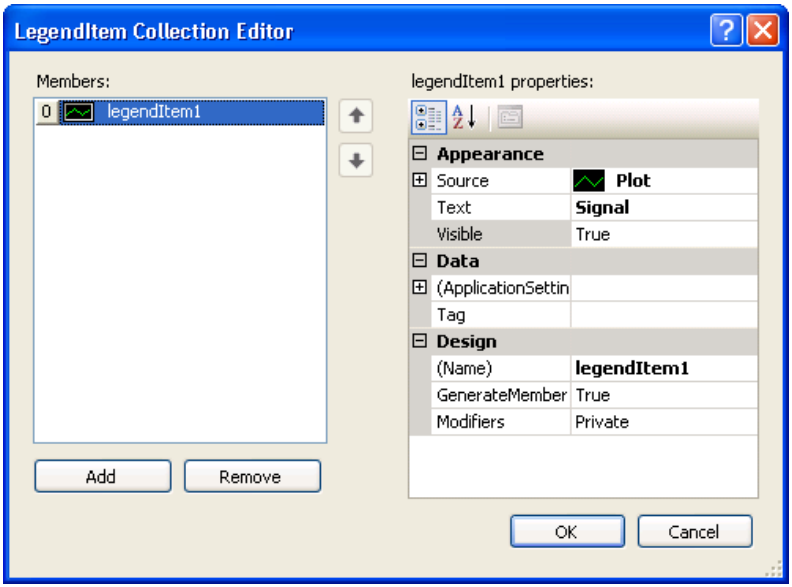
1. Double-click the button control to display the `Form1` code, with the cursor inside the click event handler of the button control.
2. Add the following code to generate random data, plot the data, calculate the mean of the data, and display the mean on the gauge.

```
[Visual Basic]
' Declare and initialize an instance of WhiteNoiseSignal.
Dim whiteNoise As New WhiteNoiseSignal()
' Store the generated data in a double array named data.
Dim data As Double() = whiteNoise.Generate(1000.0, 256)
' Use the PlotY method to plot the data.
Plot.PlotY(data)
' Use the Mean method to calculate the mean of the data.
Dim mean As Double = Statistics.Mean(data)
' Display the mean on the gauge.
gauge.Value = mean
```

```
[C#]
// Declare and initialize an instance of WhiteNoiseSignal.
WhiteNoiseSignal whiteNoise = new WhiteNoiseSignal();

// Store the generated data in a double array named data.
double[] data = whiteNoise.Generate(1000.0, 256);

// Use the PlotY method to plot the data.
Plot.PlotY(data);

// Use the Mean method to calculate the mean of the data.
double mean = Statistics.Mean(data);

// Display the mean on the gauge.
gauge.Value = mean;
```

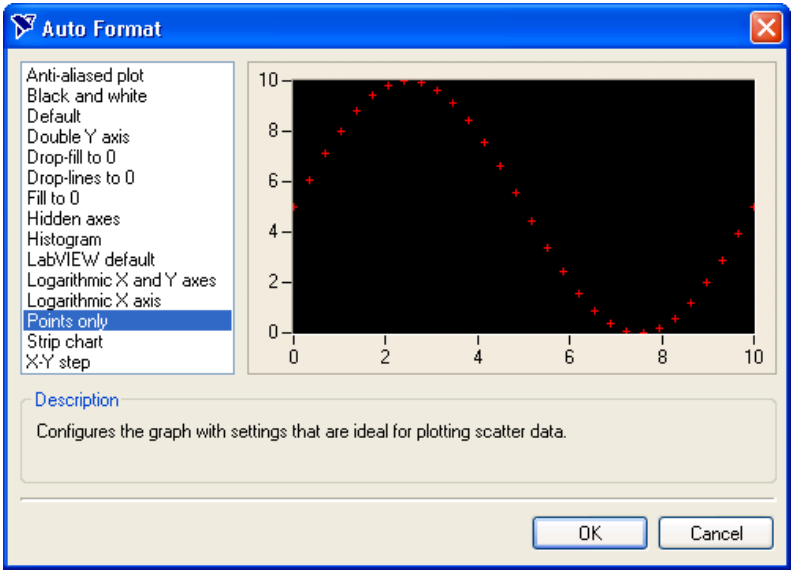
Customizing your user interface

1. Right-click the legend and select **Edit Items** to display the LegendItem Collection Editor dialog box. You use the LegendItem Collection Editor dialog box to add or remove legend items and to configure legend item properties.



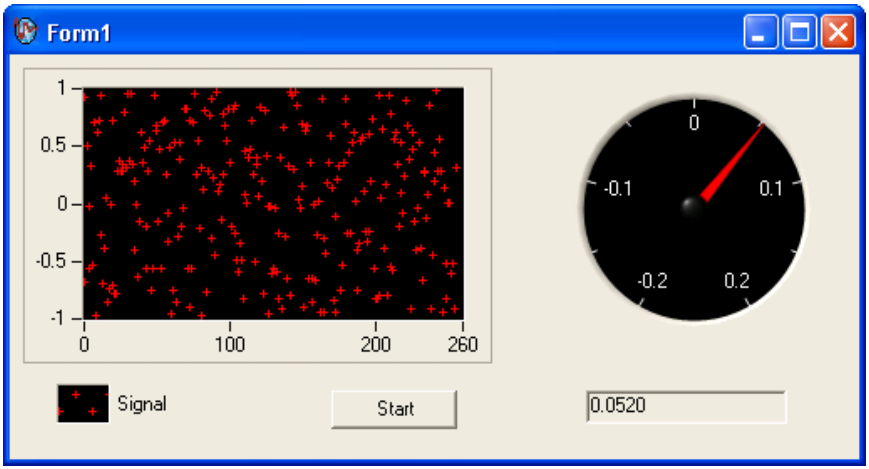
2. Select **Plot** in the **Source** drop-down list and enter `Signal` in the **Text** box. Click **OK**. Now that you have specified a legend item for the plot, changes you make to the plot will be reflected on the legend.
3. Right-click the graph and select **Auto Format** to display the Auto Format dialog box. The Auto Format dialog box provides a set of pre-configured control styles. When you select a style and click **OK**, the Auto Format feature configures the appropriate control properties to reflect the style you chose.

4. Select **Points Only**. Click **OK**. Notice that the legend changed automatically to match the formatting of the graph.



5. Click the gauge smart tag to display the Gauge Tasks.
6. Select **Auto Format** to display the Auto Format dialog box.
7. Select **Dark** and click **OK**.
8. Right-click the gauge and select **Properties** to display the Properties window.
9. Set the Range property for the gauge with the drop-down Range type editor. Type `-0.2` for the minimum value and type `0.2` for the maximum value.
10. Click the numeric edit smart tag to display the NumericEdit Tasks.
11. Select **Gauge** in the **Source** drop-down list. Setting the Source property to the gauge allows two-way binding between the controls.
12. Deselect **ArrowKeys**, **Buttons**, and **Text** for the **InteractionMode** property of the numeric edit control. Deselecting these interaction modes makes the numeric edit an indicator. The numeric edit control only displays the calculated mean.
13. Select the Format Mode property and in the Numeric Edit Format Mode Editor dialog box, change the Precision to `4` to show four decimal places of precision. Click **OK**.
14. Select **File»Save Form1.cs** to save your application.

15. Select **Debug»Start Without Debugging** to run the application.
16. After your program builds, click **Start**. Notice the graph shows the data plot, and the gauge and the numeric edit display the mean of the data.



Walkthrough: Creating a Measurement Studio Application with Web Forms Controls and Analysis

Note To complete this walkthrough, you must have either the Measurement Studio Professional or Measurement Studio Enterprise edition installed. This walkthrough will not work with the Measurement Studio Standard edition.

Measurement Studio includes user interface controls, such as a waveform graph control and a gauge control, and Analysis functionality, such as signal generation and mathematical functions. This walkthrough is designed to help you learn how to add analysis and presentation functionality to a Web Forms application by taking you through the following steps:

- **Setting up the project**—Using the Measurement Studio Application Wizard, you will create a new project that references the Measurement Studio Analysis class library and Web Forms controls.
- **Adding user interface controls to the project**—Using the Toolbox and the Properties window, you will add and configure a button, waveform graph, legend, gauge, and numeric edit user interface control.

- **Generating, plotting, and analyzing the data**—Using `NationalInstruments.Analysis.SignalGeneration.WhiteNoiseSignal` and `NationalInstruments.Analysis.Math.Statistics.Mean`, you will generate data, plot the generated data on a waveform graph, and calculate the mean of the data.
- **Customizing the user interface**—Using the Collection Editor and Auto Format dialog boxes, you will display the mean value on the gauge and the numeric edit, as well as customize your user interface.

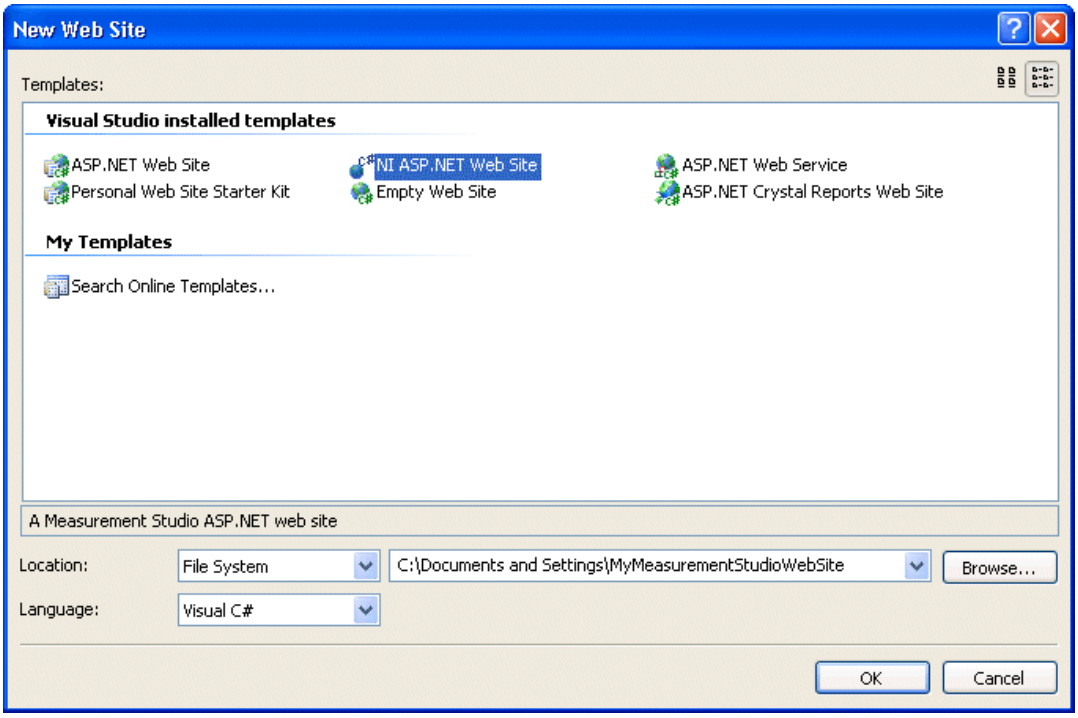
Before you begin

The following components are required to complete this walkthrough:

- Microsoft Visual Studio 2005 or Visual Studio 2008
- Measurement Studio 8.0.1 or later (Professional or Enterprise edition) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise edition) for Visual Studio 2008

Setting up the project

1. Select **Start»All Programs»Microsoft Visual Studio 2008»Microsoft Visual Studio 2008**.
2. Select **File»New»Web Site**. The New Web Site dialog box launches.

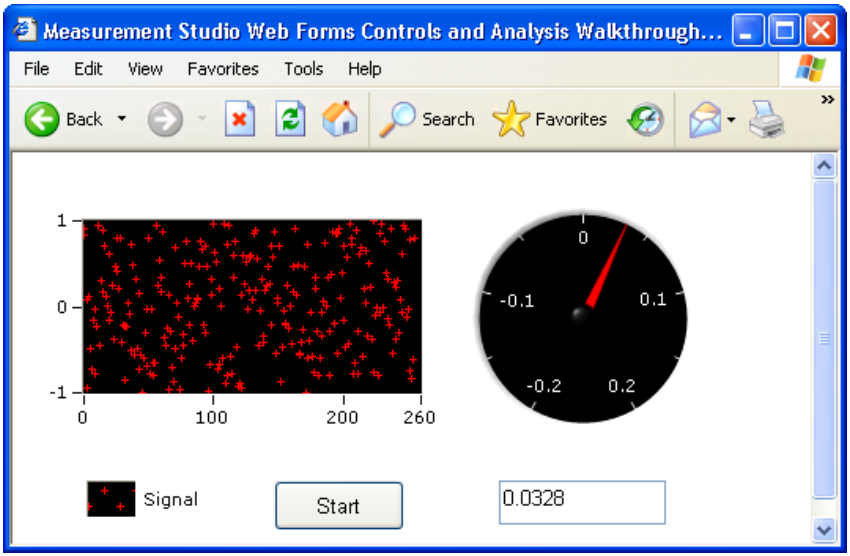


3. In the Templates pane, select **NI ASP.NET Web Site**. Select `File System` and specify a Web location of your choice.
4. Use the drop-down box to select **Visual C#** or **Visual Basic**, depending on which language you want to create the project in.
5. Click **OK**. The Measurement Studio ASP.NET Web Site Wizard launches.
6. Select **Analysis Library** and **Web Forms User Interface Control Library**.

   If you are working with an existing project, you can access the Add/Remove Class Libraries dialog box by selecting **Measurement Studio»Add/Remove .NET Class Libraries Wizard**.
7. Click **Finish** to display `Default.aspx` in the Web Forms Designer.
8. You can change the title of your Web page. Click inside the `<title>` tag and rename the title to **Measurement Studio Web Forms Controls and Analysis Walkthrough**.

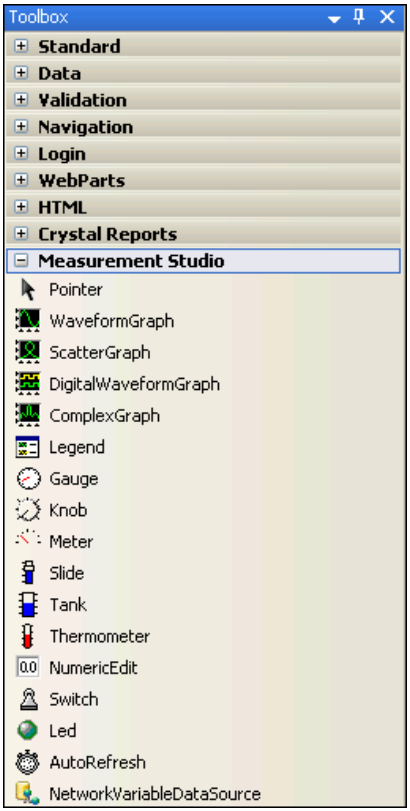
Adding user interface controls to the project

1. In this section, you will build a Web page that looks like the following screenshot.



2. Click **Design** in the lower left corner to switch from Source View to Design View.
3. Select **View»Toolbox** to display the Toolbox. The Toolbox contains components and controls that you can add to your project.
4. Expand the **HTML** group on the Toolbox. Select the Table control in the Toolbox and drag and drop it on the form. You use the table cells to arrange the user interface controls on your Web page, as shown in the previous screenshot.
5. The default table that appears is 3x3. This table provides a customizable form for arranging the user interface controls for your Web page. Expand the table to approximately 300 px (pixels) tall by 550 px wide by clicking and dragging the table borders.
6. Merge the top two cells of all three columns by selecting the cells, right-clicking, and selecting **Modify»Merge Cells**.
7. Expand the **Standard** group on the Toolbox. The Standard group contains ASP.NET server controls included in the `System.Web.UI` namespace.
8. Select the **Button** control and drag and drop it into the lower right table cell.

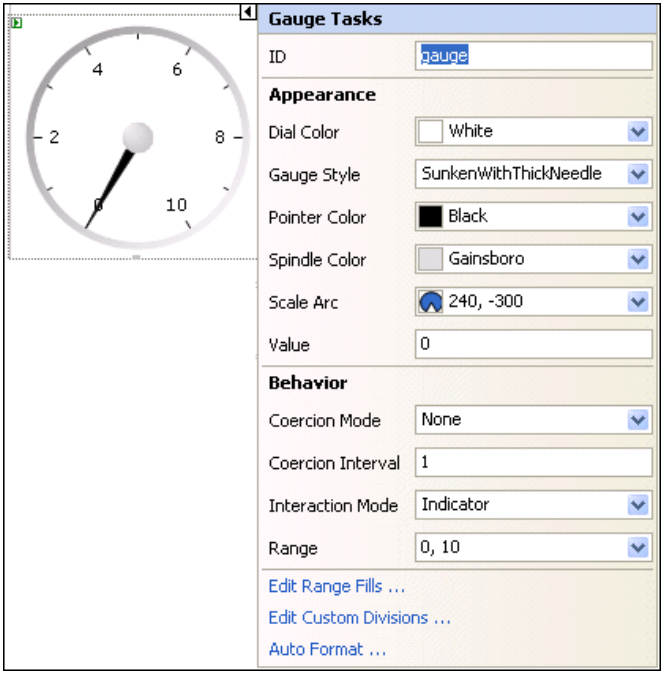
9. Right-click the button and select **Properties** to display the Properties window. You configure the properties of the control in the Properties window.
10. Scroll to the Text property in the Properties window. Type `Start` for the button text.
11. Expand the **Measurement Studio** group on the Toolbox.



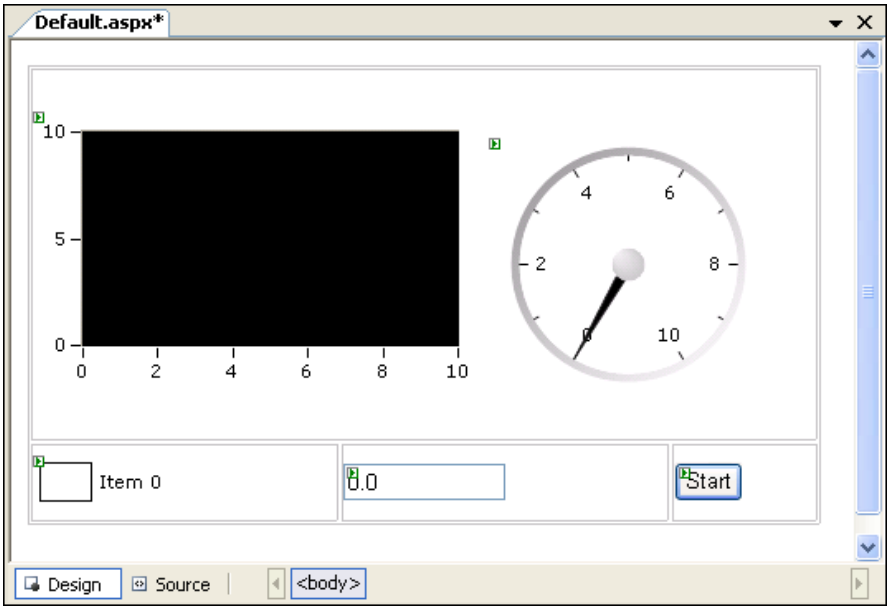
12. Select the **WaveformGraph** control and drag and drop it into the top table cell.
13. On the waveform graph smart tag, type `graph` for the name of the waveform graph ID.

    To access the smart tag, left click on a control to select it and then left click on the arrow button in the upper right corner of the control.
14. Select the **Legend** control and drag and drop it into the bottom left table cell.

15. Select the **NumericEdit** control and drag and drop it into the bottom center table cell.
16. On the numeric edit smart tag, type `numericedit` for the name of the numeric edit ID.
17. Select the **Gauge** control and drag and drop it into the top table cell, to the right of the waveform graph.
18. On the gauge smart tag, type `gauge` for the name of the gauge ID. Resize controls and table cells as necessary.



The following screenshot shows `Default.aspx` with the user controls.



Generating, plotting, and analyzing the data

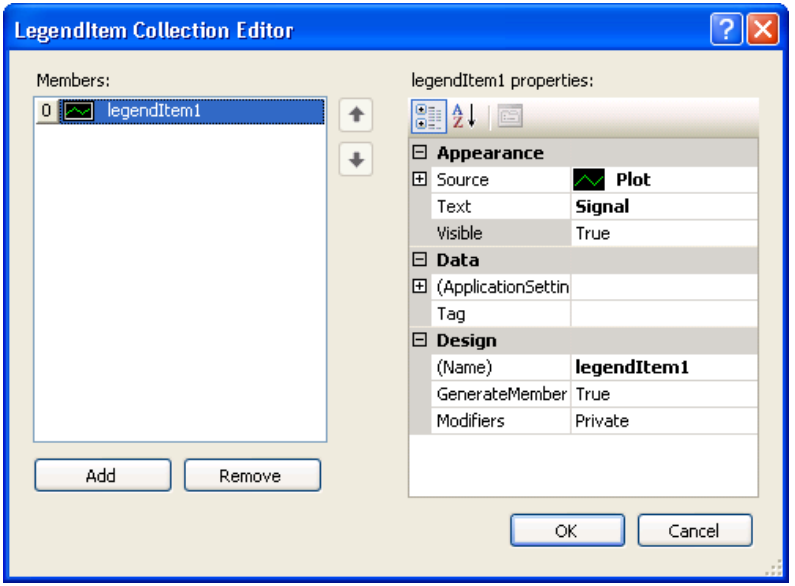
1. Double-click the button control to display the `Default.aspx.cs` code, with the cursor inside the click event handler of the button control.
2. Add the following code to generate random data, plot the data, calculate the mean of the data, and display the mean on the gauge.

```
[Visual Basic]
' Declare and initialize an instance of WhiteNoiseSignal.
Dim whiteNoise As New WhiteNoiseSignal()
' Store the generated data in a double array named data.
Dim data As Double() = whiteNoise.Generate(1000.0, 256)
' Use the PlotY method to plot the data.
graph.PlotY(data)
' Use the Mean method to calculate the mean of the data.
Dim mean As Double = Statistics.Mean(data)
' Display the mean on the numeric edit.
numericedit.Value = mean
' Display the mean on the gauge.
gauge.Value = mean
```

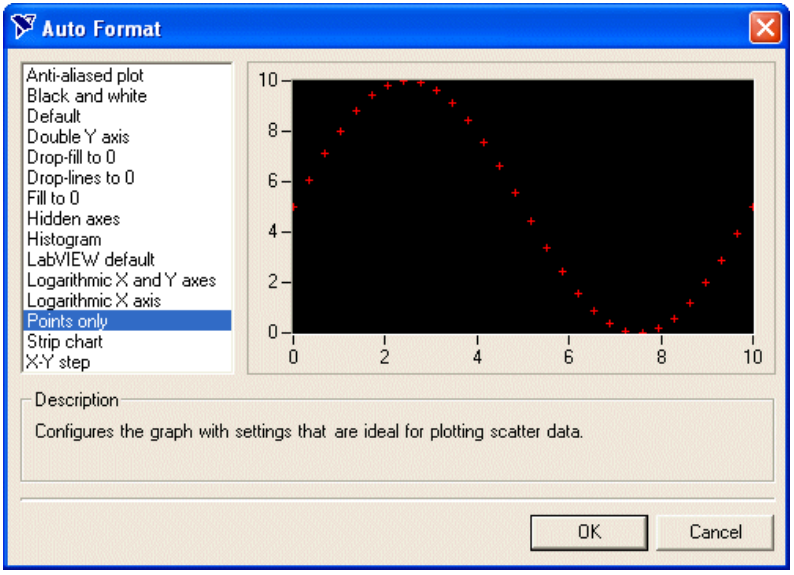
```
[C#]
// Declare and initialize an instance of WhiteNoiseSignal.
WhiteNoiseSignal whiteNoise = new WhiteNoiseSignal();

// Store the generated data in a double array named data.
double[] data = whiteNoise.Generate(1000.0, 256);

// Use the PlotY method to plot the data.
graph.PlotY(data);

// Use the Mean method to calculate the mean of the data.
double mean = Statistics.Mean(data);

// Display the mean on the numeric edit.
numericedit.Value = mean;
// Display the mean on the gauge.
gauge.Value = mean;
```

Customizing your user interface

1. Select the **Default.aspx** tab to return to the Web Forms Designer.
2. Right-click the legend and select **Edit Items** to display the LegendItem Collection Editor dialog box. You use the LegendItem Collection Editor dialog box to add or remove legend items and to configure legend item properties.

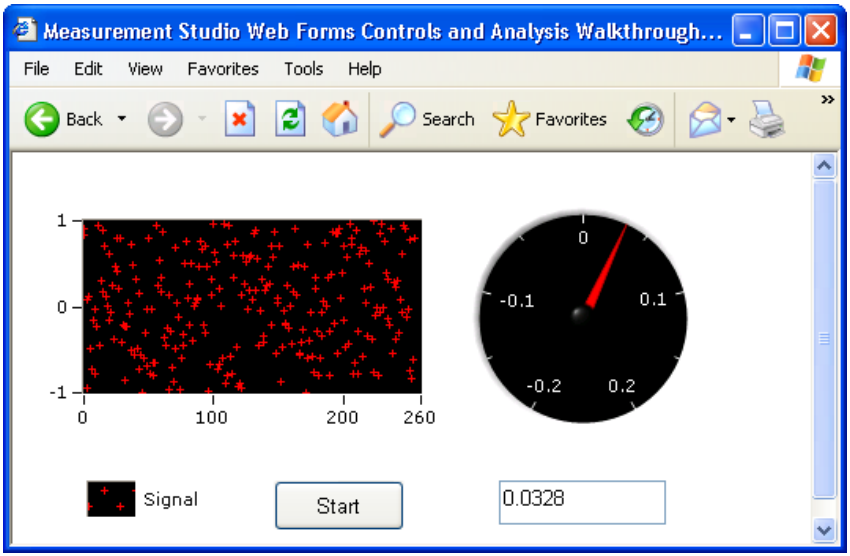


3. Select **Plots[0]** in the **Source** drop-down list and enter `Signal` in the **Text** box. Click **OK**. Now that you have specified a legend item for the plot, changes you make to the plot will be reflected on the legend.
4. Right-click the graph and select **Auto Format** to display the Auto Format dialog box. The Auto Format dialog box provides a set of pre-configured control styles. When you select a style and click **OK**, the Auto Format feature configures the appropriate control properties to reflect the style you chose.
5. Select **Points Only**. Click **OK**. Notice that the legend changed automatically to match the formatting of the graph.



6. Right-click the gauge and select **Auto Format** to display the Auto Format dialog box.
7. Select **Dark** and click **OK**.
8. On the gauge smart tag, set the Range property for the gauge with the drop-down Range type editor. Type `-0.2` for the minimum value and type `0.2` for the maximum value.
9. On the numeric edit smart tag, select **Indicator** for the **InteractionMode** property of the numeric edit control.
10. On the numeric edit smart tag, select Format Mode and in the Numeric Format Mode Editor dialog box, change the Precision to `4` to show four decimal places of precision. Click **OK**.

11. Select **File»Save Default.aspx** to save your application.
12. Select **Debug»Start Without Debugging** to run the application.
13. After your program builds, click **Start**. Notice the graph shows the data plot, and the gauge and the numeric edit display the mean of the data. The following screenshot shows `Default.aspx` in its final form.



Walkthrough: Creating a Measurement Studio Application with Windows Forms Controls and Network Variable

Note To complete this walkthrough, you must have either the Measurement Studio Professional or Measurement Studio Enterprise edition installed. This walkthrough will not work with the Measurement Studio Standard edition.

Measurement Studio includes user interface controls, such as a waveform graph control, and network variable functionality to transfer live measurement data between applications over the network. This walkthrough is designed to help you learn how to add network variable functionality to a Windows Forms application by taking you through the following steps:

- **Writing an array of data to the server**—Using `NationalInstruments.NetworkVariable.NetworkVariableBufferedWriter<TValue>`, you will create and run a console application that writes an array of values to the server.
- **Setting up a Windows Forms project**—Using the Measurement Studio Application Wizard, you will create a new project that references the Measurement Studio Network Variable class library and Windows Forms controls.
- **Configuring the network variable data source control**—Using the Toolbox and the `NationalInstruments.NetworkVariable.WindowsForms.NetworkVariableDataSource` smart tag, you will add and configure a data source control to your application.
- **Displaying the array of data on a Windows Forms page**—Using the Toolbox, you will add and configure a `NationalInstruments.WaveformGraph` control to display the data.

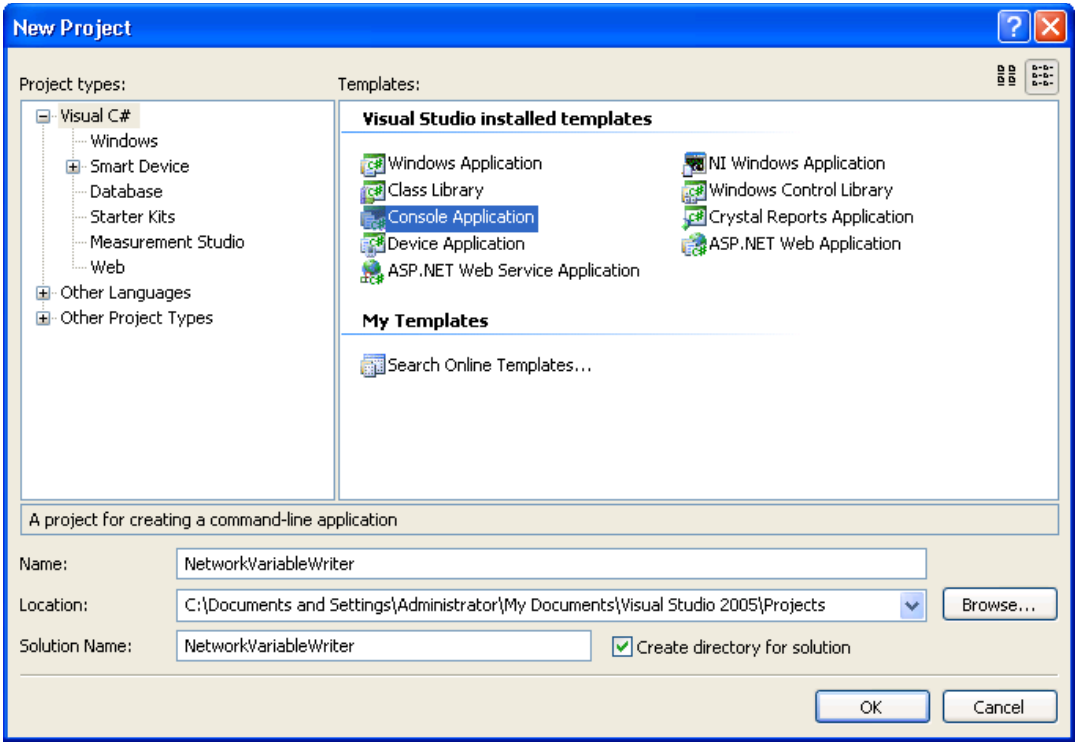
Before you begin

The following components are required to complete this walkthrough:

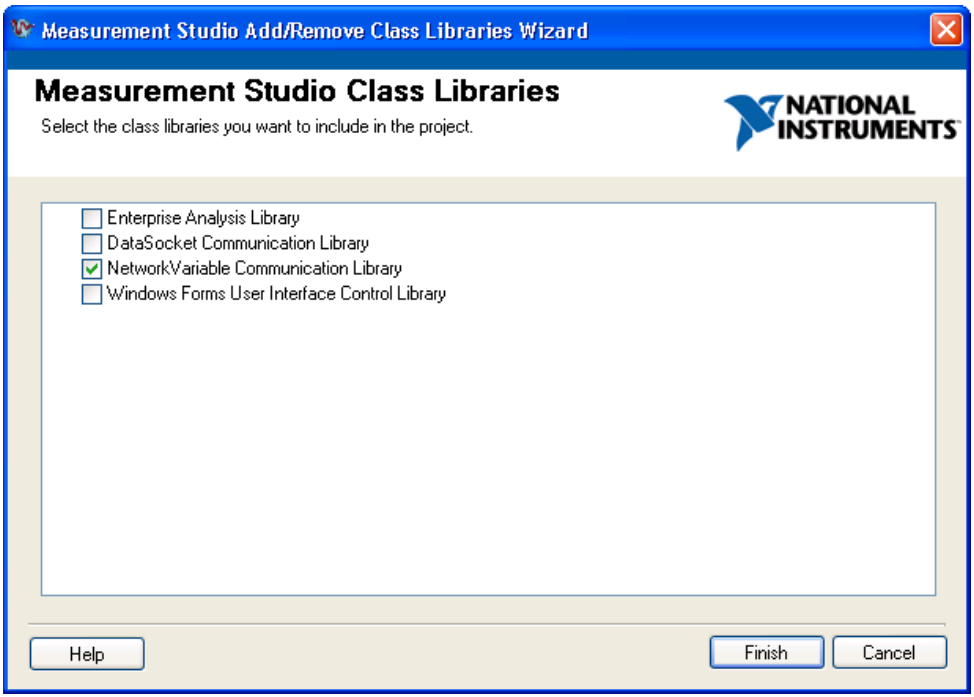
- Microsoft Visual Studio 2005 or Visual Studio 2008
- Measurement Studio 8.0.1 or later (Professional or Enterprise edition) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise edition) for Visual Studio 2008

Writing an array of data to the server

1. Select **Start»All Programs»Microsoft Visual Studio 2008»Microsoft Visual Studio 2008**.
2. Select **File»New»Project**. The New Project dialog box launches.



3. In the Project Types pane, select **Visual C#** or **Visual Basic**, depending on which language you want to create the project in.
4. In the Templates pane, select **Console Application**. Specify `NetworkVariableWriter` for **Name** and specify a **Location** of your choice.
5. Click **OK**.
6. Select **Measurement Studio»Add/Remove .NET Class Libraries**. The Measurement Studio Add/Remove Class Libraries Wizard launches. You use this wizard to add Measurement Studio components to your project.
7. Select **NetworkVariable Communication Library**. Click **Finish**.



8. In `Program.cs`, add the following code to write an array of data to the server:

```
[Visual Basic]
Imports NationalInstruments.NetworkVariable
Imports System.Threading
Module Module1
Private Function GenerateDoubleArray(ByVal phase As Double) As Double()
     Dim values(999) As Double
     Dim x As Integer
     For x = 0 To 999
     values(x) = Math.Sin(((2 * Math.PI * x) / 1000) + phase) * 2
     Next x
     Return values
End Function
Sub Main()
     Const location As String = "\\localhost\system\double"
     Dim bufferedWriter As NetworkVariableBufferedWriter(Of Double()) = New
NetworkVariableBufferedWriter(Of Double())(location)
     bufferedWriter.Connect()
     Dim phase As Integer = 0
```

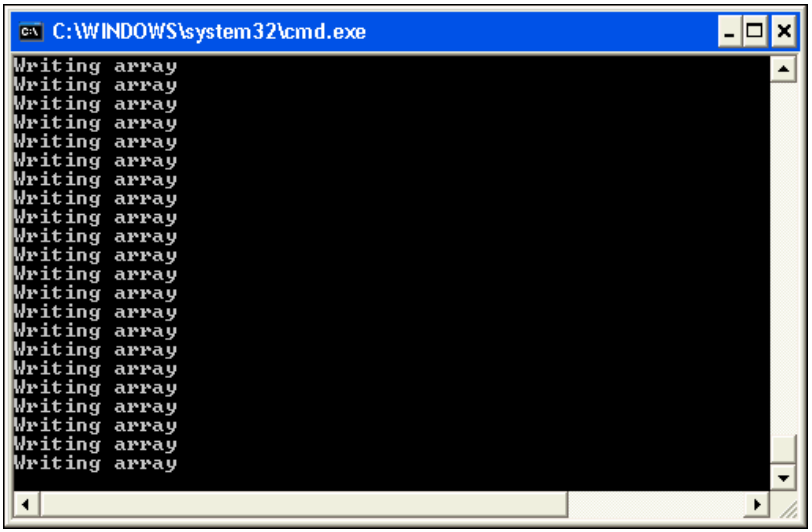
```
     While (True)
     Dim values As Double() = GenerateDoubleArray(phase)
     Console.WriteLine("Writing Array")
     bufferedWriter.WriteValue(values)
     Thread.Sleep(500)
     phase = phase + 1
     End While
End Sub
End Module
```

[C#]

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Threading;
using NationalInstruments.NetworkVariable;
namespace NetworkVariableWriter
{
class Program
{
    private static double[] GenerateDoubleArray(double phase)
    {
        double[] values = new double[1000];
        for (int x = 0; x < 1000; x++)
            values[x] = Math.Sin(((2 * Math.PI * x) / 1000) + phase) * 2;
        return values;
    }
    static void Main(string[] args)
    {
        const string Location = @"\\localhost\system\double";
        NetworkVariableBufferedWriter<double[]> bufferedWrite = new
        NetworkVariableBufferedWriter<double[]>(Location);
        bufferedWrite.Connect();
        int phase = 0;
        while (true)
        {
            double[] value = GenerateDoubleArray(phase);
            Console.WriteLine("Writing array");
            bufferedWrite.WriteValue(value);
            Thread.Sleep(500);
            phase++;
        }
    }
}
}
```

Note You should choose the appropriate code depending on whether you created a VB or C# project.

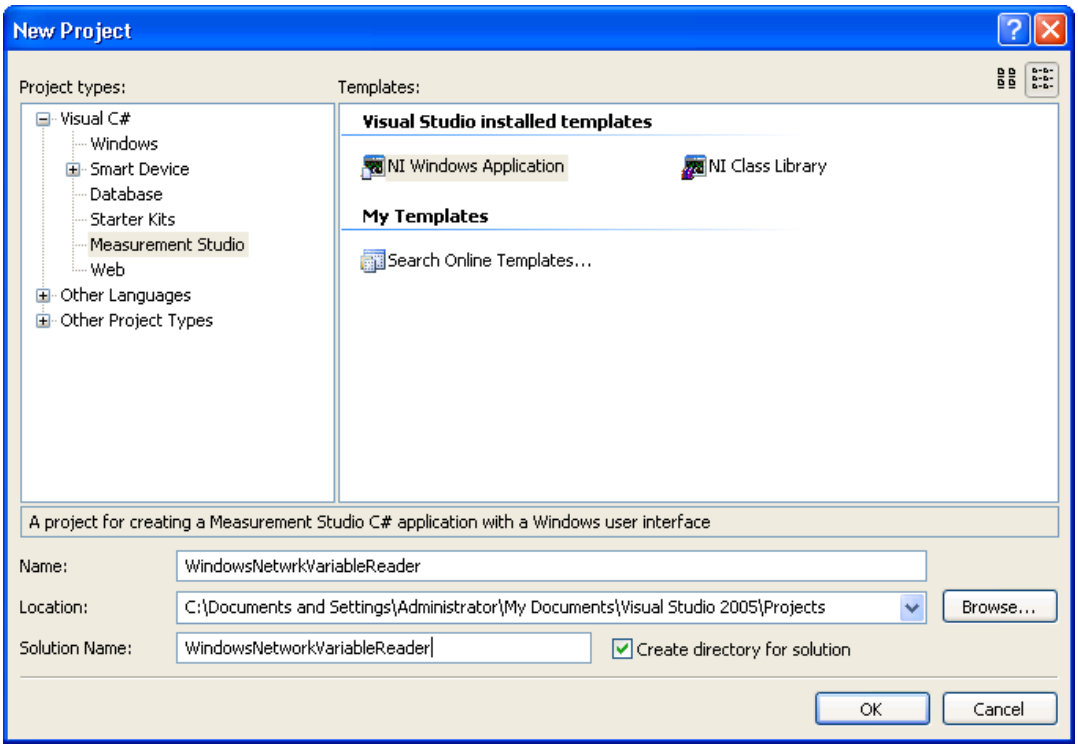
9. Select **Debug»Start Without Debugging** to run the application.



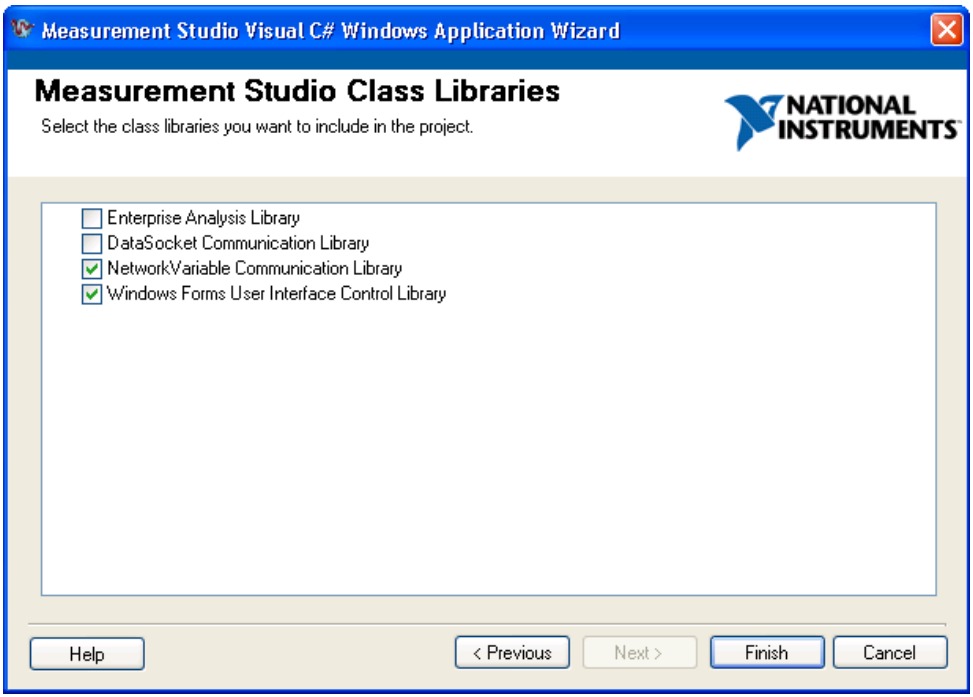
10. Minimize the console, but keep the application running.

Setting up a Windows Forms project

1. Select **Start»All Programs»Microsoft Visual Studio 2008» Microsoft Visual Studio 2008**.
2. Select **File»New»Project**. The New Project dialog box launches.



3. In the Project types pane, select **Measurement Studio** under **Visual C#** or **Visual Basic**, depending on which language you want to create the project in.
4. In the Templates pane, select **NI Windows Application**. Name the project `WindowsNetworkVariableReader` and specify a Location you wish to save to project by clicking **Browse** and navigating to a directory of your choice.
5. Click **OK**. The Measurement Studio Application Wizard launches.
6. Select **Network Variable Communication Library** and **Windows Forms User Interface Control Library**.



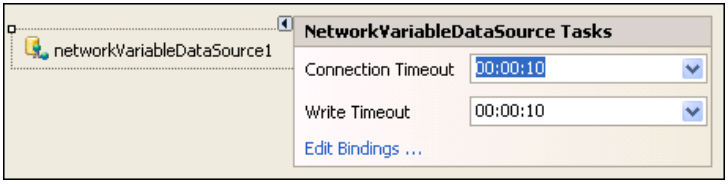
Tip If you are working with an existing project, you can access the Add/Remove Class Libraries wizard by selecting **Measurement Studio»Add/Remove Class Libraries Wizard**.

7. Click **Finish** to display `Form1` in the Windows Forms Designer.

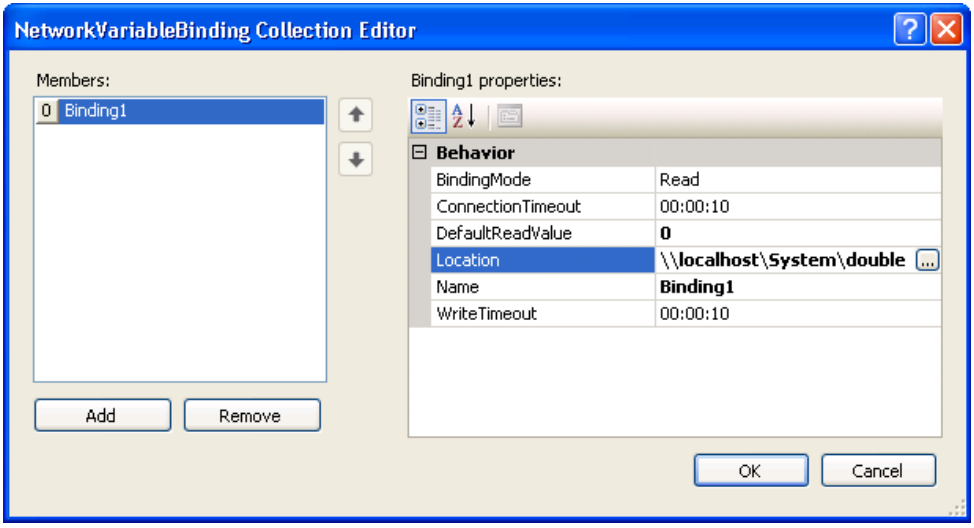
Configuring the network variable data source control

1. Select **View»Toolbox** to display the Toolbox. The toolbox contains components and controls that you can add to your project.
2. Expand the **Measurement Studio** group on the Toolbox.
3. Select the NetworkVariableDataSource control in the toolbox and drag and drop it on the form. The `NationalInstruments.NetworkVariable.WindowsForms.NetworkVariableDataSource` control is a data source control with functionality similar to `System.Web.UI.WebControls.ObjectDataSource` and `System.Web.UI.WebControls.SqlDataSource` in the .NET Framework. The `NationalInstruments.NetworkVariable.WindowsForms.NetworkVariableDataSource` control encapsulates `NationalInstruments.NetworkVariable` functionality.

4. In the NetworkVariableDataSource smart tag, select **Edit Bindings** to launch the NetworkVariableBinding Collection Editor dialog box.



5. Select **Add** to create a connection with the underlying network variable, You can use the NetworkVariableBinding Collection Editor to configure the binding properties. Enter **0** as the **DefaultReadValue**.

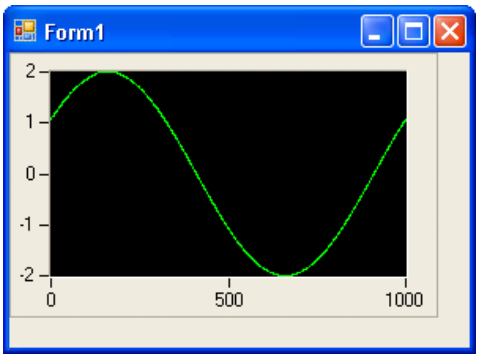


6. For the **Location**, browse to the `\\localhost\System\double` location in the Select Network Item dialog box.

7. Click **OK** to return to the NetworkVariableBinding Collection Editor dialog box.
8. Click **OK** to return to the Windows Forms Designer.

Displaying the array of data on a Windows form

1. Select **WaveformGraph** in the Toolbox and drag and drop it on the form.
2. Right-click the waveform graph and select **Properties** to display the Properties window for the graph. You can configure the properties of the control in the Properties window.
3. Expand the **Data Bindings** group in the Properties window. Select **Other Data Sources»Form 1 List Instances»networkVariableDataSource1»Binding1** from the **Binding Data** drop-down list. This will bind the waveform graph to the network variable that you are writing to in the console application. The waveform graph will then read and display the data being written to the network variable.
4. Select **File»Save Form1** to save your application.
5. Select **Debug»Start Without Debugging** to run the application. The waveform graph displays the array of data.



Walkthrough: Creating a Measurement Studio Application with Web Forms Controls and Network Variable

Note To complete this walkthrough, you must have either the Measurement Studio Professional or Measurement Studio Enterprise edition installed. This walkthrough does not work with the Measurement Studio Standard edition.

Measurement Studio includes user interface controls, such as a waveform graph control, and network variable functionality to transfer live measurement data between applications over the network. This walkthrough is designed to help you learn how to add network variable functionality to a Web Forms application by taking you through the following steps:

- **Writing an array of data to the server**—Using `NationalInstruments.NetworkVariable.NetworkVariableBufferedWriter<TValue>`, you will create and run a console application that writes an array of values to the server.
- **Setting up a Web Forms project**—Using the Measurement Studio Application Wizard, you will create a new project that references the Measurement Studio Network Variable class library and Web Forms controls.
- **Configuring the network variable data source control**—Using the Toolbox and the `NationalInstruments.NetworkVariable.WebForms.NetworkVariableDataSource` smart tag, you will add and configure a data source control to your application.

- **Displaying the array of data on a Web page**—Using the Toolbox, you will add and configure an `NationalInstruments.UI.WebForms.AutoRefresh` control and a `NationalInstruments.UI.WebForms.WaveformGraph` control to display the data.

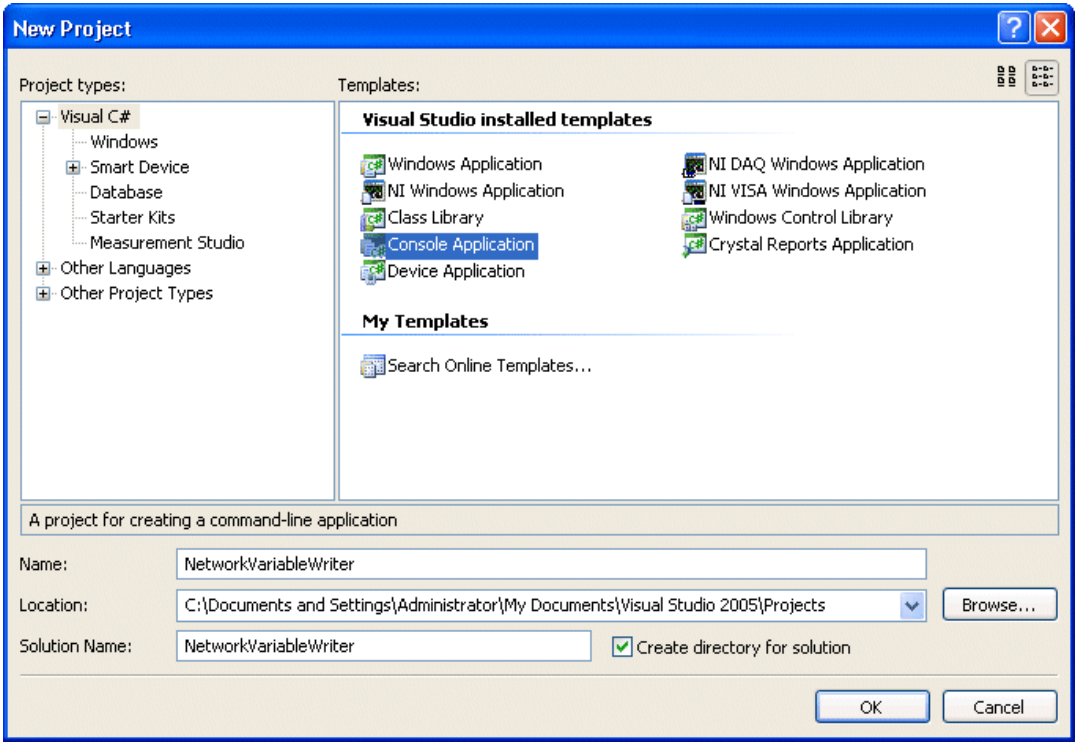
Before you begin

The following components are required to complete this walkthrough:

- Microsoft Visual Studio 2005 or Visual Studio 2008
- Measurement Studio 8.0.1 or later (Professional or Enterprise edition) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise edition) for Visual Studio 2008

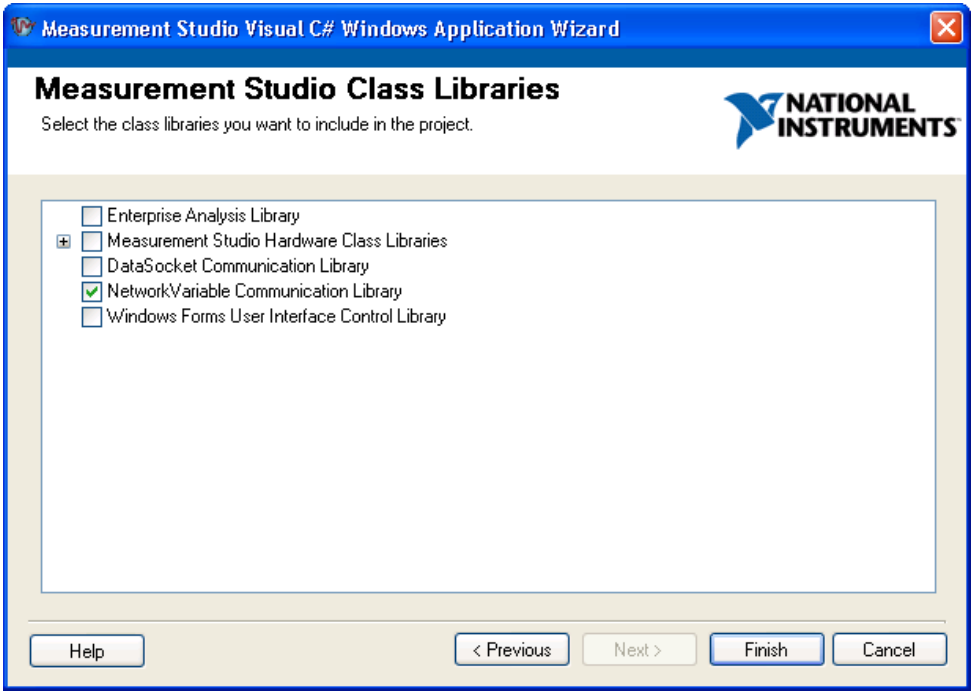
Writing an array of data to the server

1. Select **Start»All Programs»Microsoft Visual Studio 2008» Microsoft Visual Studio 2008**.
2. Select **File»New»Project**. The New Project dialog launches.



3. In the Project Types pane, select **Visual C#** or **Visual Basic**, depending on which language you want to create the project in.

4. In the Templates pane, select **Console Application**. Specify `NetworkVariableWriter` for **Name** and specify a **Location** of your choice.
5. Click **OK**.
6. Select **Measurement Studio»Add/Remove .NET Class Libraries**. The Measurement Studio Add/Remove Class Libraries Wizard launches. You use this wizard to add Measurement Studio components to your project.
7. Select **NetworkVariable Communication Library**. Click **Finish**.



8. In `Program.cs`, add the following code to write an array of data to the server:

```
[Visual Basic]
Imports NationalInstruments.NetworkVariable
Imports System
Imports System.Collections.Generic
Imports System.Text
Imports System.Threading
Imports System.Linq
```

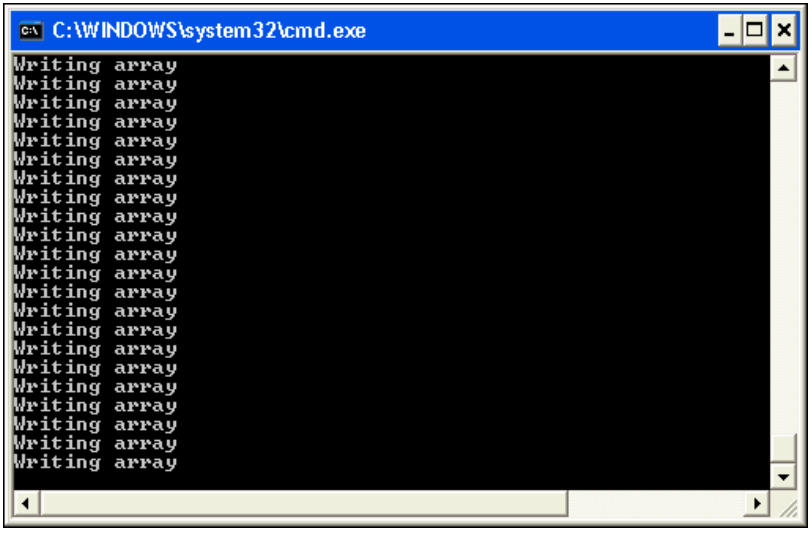
```
Module Module1
Private Function GenerateDoubleArray(ByVal phase As Double) As Double()
    Dim values(999) As Double
    Dim x As Integer
    For x = 0 To 999
        values(x) = Math.Sin(((2 * Math.PI * x) / 1000) + phase) * 2
    Next x
    Return values
End Function
Sub Main()
    Const location As String = "\\localhost\system\double"
    Dim bufferedWriter As NetworkVariableBufferedWriter(Of Double()) = New
NetworkVariableBufferedWriter(Of Double())(location)
    bufferedWriter.Connect()
    Dim phase As Integer = 0
    While (True)
        Dim values As Double() = GenerateDoubleArray(phase)
        Console.WriteLine("Writing Array")
        bufferedWriter.WriteValue(values)
        Thread.Sleep(500)
        phase = phase + 1
    End While
End Sub
End Module
```

[C#]

```
using NationalInstruments.NetworkVariable;
using System;
using System.Collections.Generic;
using System.Text;
using System.Threading;
using System.Linq;
namespace NetworkVariableWriter
{
class Program
{
    private static double[] GenerateDoubleArray(double phase)
    {
        double[] values = new double[1000];
        for (int x = 0; x < 1000; x++)
            values[x] = Math.Sin(((2 * Math.PI * x) / 1000) + phase) * 2;
        return values;
    }
```

```
    static void Main(string[] args)
    {
        const string Location = @"\\localhost\system\double";
        NetworkVariableBufferedWriter<double[]> bufferedWriter = new
NetworkVariableBufferedWriter<double[]>(Location);
        bufferedWriter.Connect();
        int phase = 0;
        while (true)
        {
            double[] value = GenerateDoubleArray(phase);
            Console.WriteLine("Writing array");
            bufferedWriter.WriteValue(value);
            Thread.Sleep(500);
            phase++;
        }
    }
}
}
```

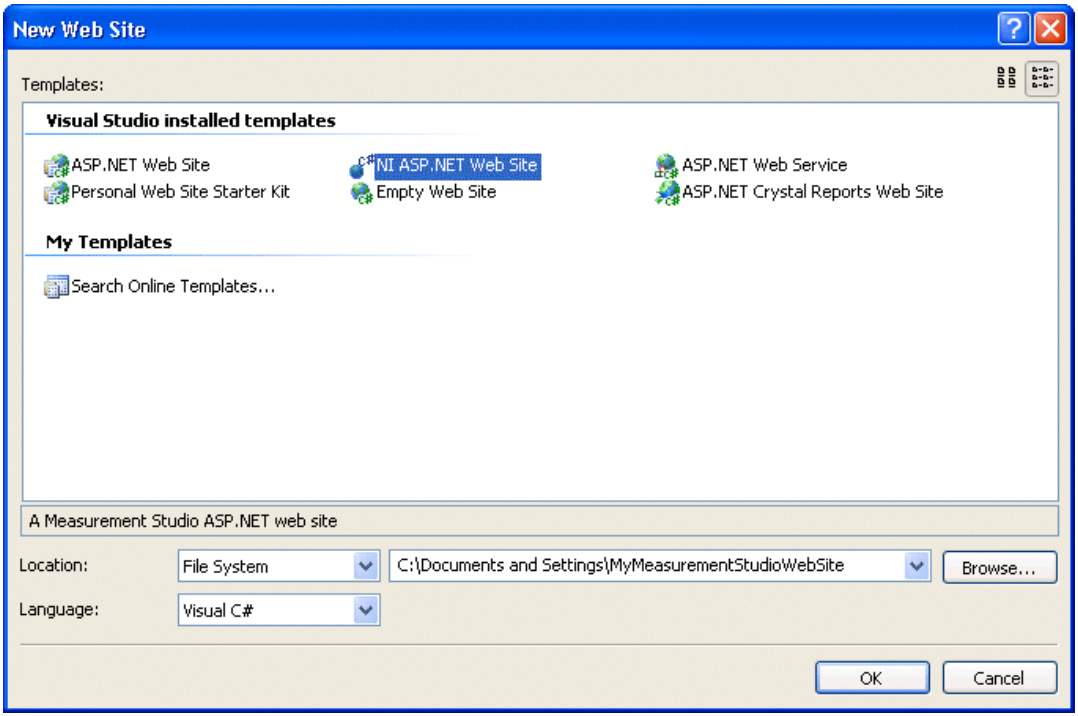
9. Select **Debug»Start Without Debugging** to run the application.



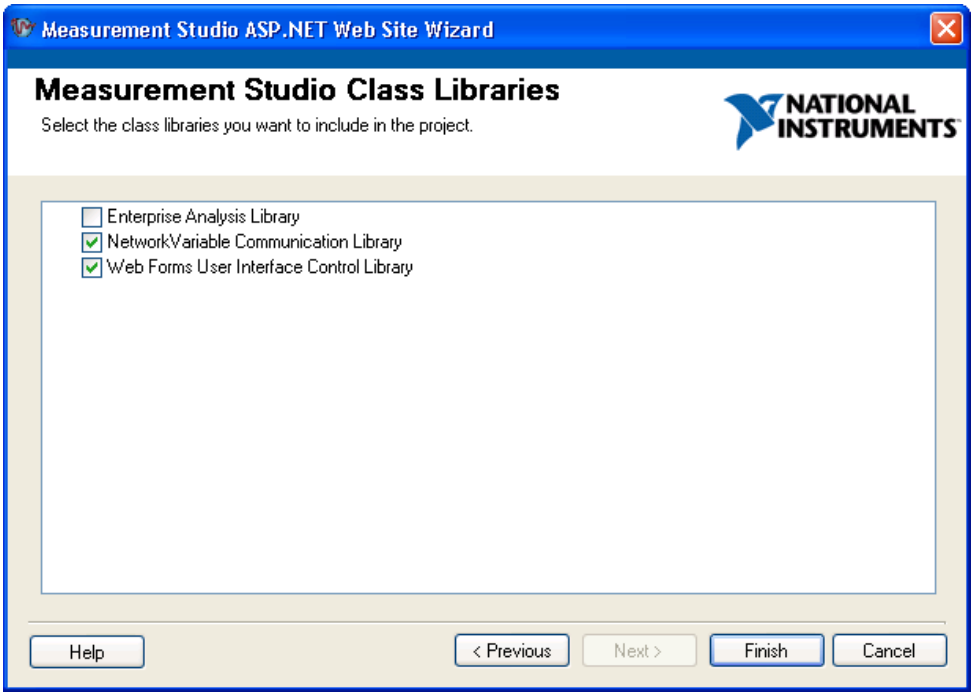
10. Minimize the console application, but keep the application running.

Setting up a Web Forms project

1. Select **Start»All Programs»Microsoft Visual Studio 2008» Microsoft Visual Studio 2008**.
2. Select **File»New»Web Site**. The New Web Site dialog box launches.



3. In the Templates pane, select **NI ASP.NET Web Site**. Select `File System` for Location and specify a file path of your choice.
4. Use the drop-down box to select **Visual C#** or **Visual Basic**, depending on which language you want to create the project in.
5. Click **OK**. The Measurement Studio ASP.NET Web Site Wizard launches.
6. Select **Network Variable Communication Library** and **Web Forms User Interface Control Library**.



Tip If you are working with an existing project, you can access the Add/Remove Class Libraries dialog box by selecting **Measurement Studio»Add/Remove Class Libraries Wizard**.

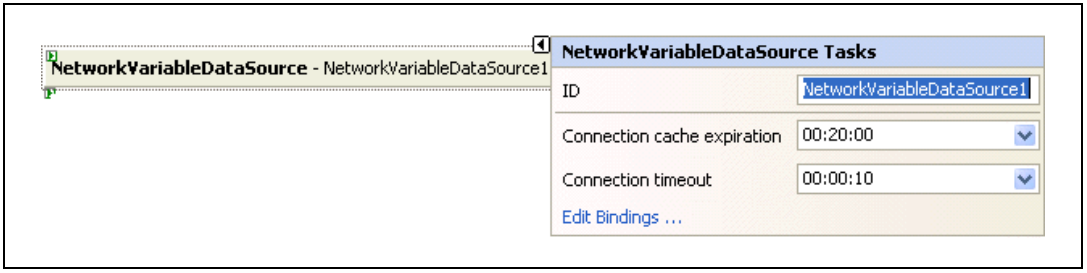
7. Click **Finish** to display `Default.aspx` in the Web Forms Designer.
8. You can change the title of your Web page. Click inside the `<title>` tag and rename the title to **Measurement Studio Network Variable and Web Forms Controls Walkthrough**.

Configuring the network variable data source control

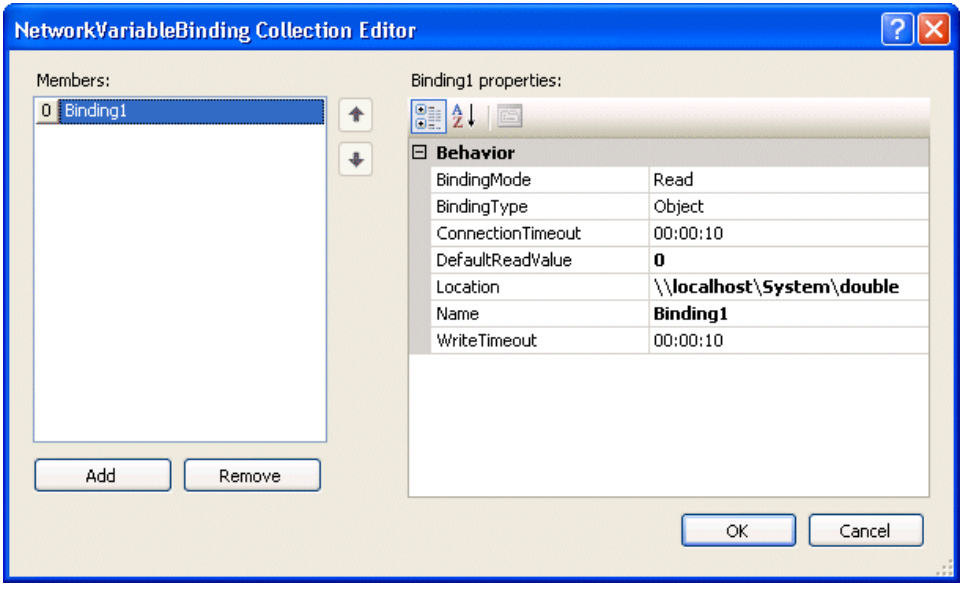
1. Click **Design** in the lower left corner to switch from Source View to Design View.
2. Select **View»Toolbox** to display the Toolbox. The toolbox contains components and controls that you can add to your project.
3. Expand the **Measurement Studio** group on the Toolbox.
4. Select the NetworkVariableDataSource control in the toolbox and drag and drop it on the form. The `NationalInstruments.NetworkVariable.WebForms.NetworkVariableDataSource` control is a data source control with functionality similar to

`System.Web.UI.WebControls.ObjectDataSource` and `System.Web.UI.WebControls.SqlDataSource`in the .NET Framework. The `NationalInstruments.NetworkVariable.WebForms.NetworkVariableDataSource` control encapsulates `NationalInstruments.NetworkVariable` functionality.

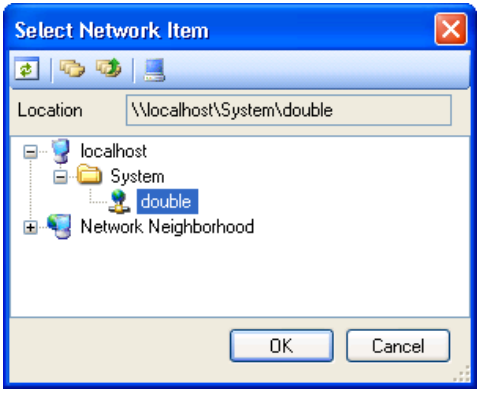
5. In the NetworkVariableDataSource smart tag, select **Edit Bindings** to launch the NetworkVariableBinding Collection Editor dialog box.



6. Select **Add**. You add a binding to create a connection with the underlying network variable, and you use the NetworkVariableBinding Collection Editor to configure the binding properties. Select **Object** for the BindingType. You select **Object** because this walkthrough binds to `NationalInstruments.UI.WebForms.WaveformGraph.BindingData`. Enter **0** as the **DefaultReadValue**.



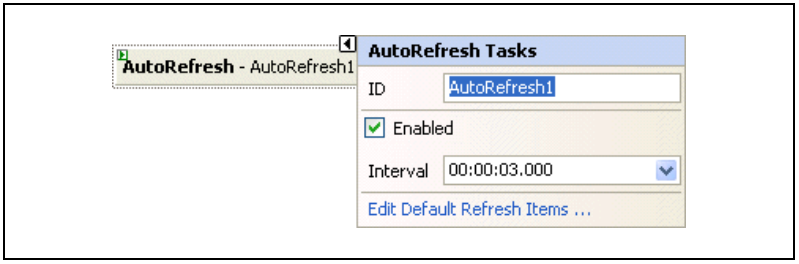
7. Browse to the `\\localhost\System\double` location in the Select Network Item dialog box.



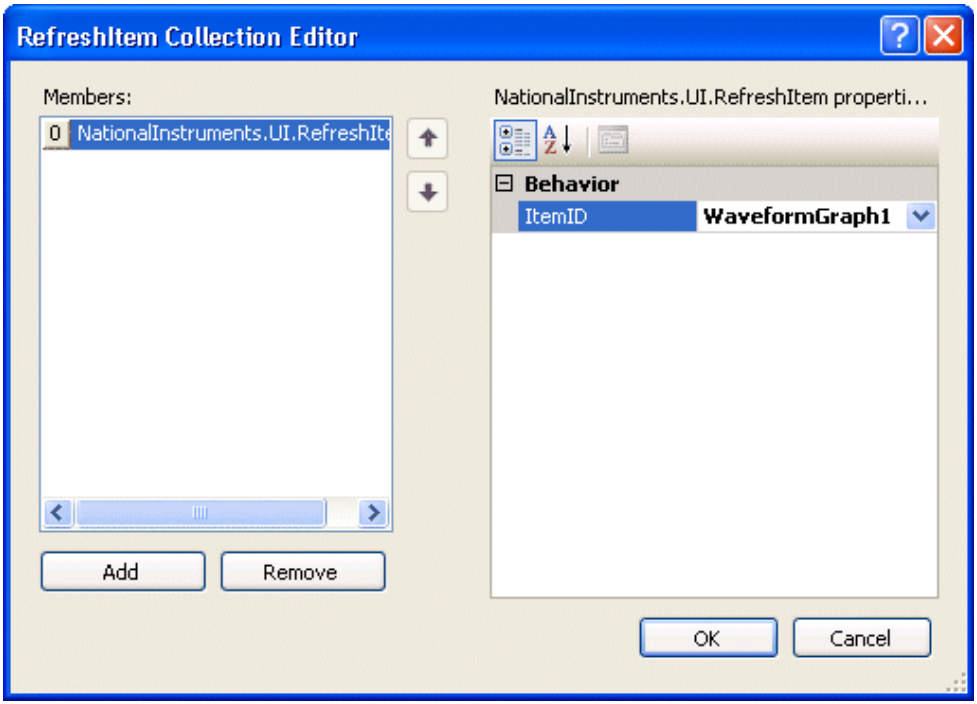
8. Click **OK** to return to the NetworkVariableBinding Collection Editor dialog box.
9. After you configure the binding properties, click **OK** to return to the ASP.NET Designer.

Displaying the array of data on a Web page

1. Select **WaveformGraph** in the Toolbox and drag and drop it on the form.
2. Select **AutoRefresh** in the Toolbox and drag and drop it on the form.
3. In the AutoRefresh smart tag, check **Enabled**. Select **Edit Default Refresh Items** to launch the RefreshItem Collection Editor dialog box.



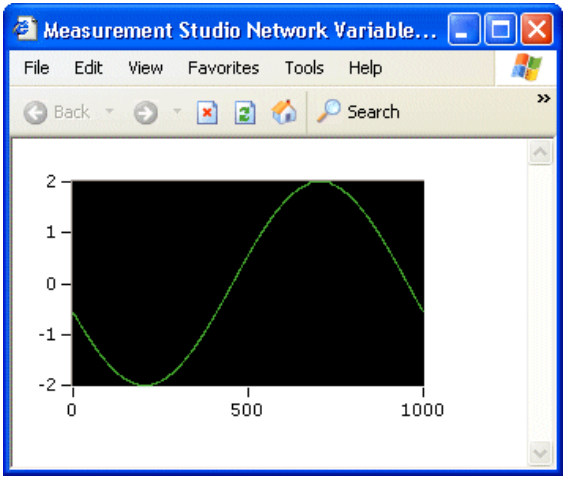
4. Select **Add**. Select **WaveformGraph1** for the ItemID and click **OK**.



5. Double-click the AutoRefresh control. Add the following code to the AutoRefresh event handler to bind the waveform graph control to the network variable data source control:

```
[Visual Basic]
WaveformGraph1.BindingData =
NetworkVariableDataSource1.Bindings(0).GetValue()

[C#]
WaveformGraph1.BindingData =
NetworkVariableDataSource1.Bindings[0].GetValue();
```

6. Select **File»Save Default.aspx** to save your application.
7. Select **Debug»Start Without Debugging** to run the application. The waveform graph displays the array of data.



Note You can also use the `System.Web.UI.WebControls.FormView` control to bind to `NationalInstruments.NetworkVariable.WebForms.NetworkVariableDataSource`. Refer to *Using the Measurement Studio Network Variable Data Source in Web Forms* for more information.

Walkthrough: Creating a Measurement Studio NI-DAQmx Application

Note To complete this walkthrough, you must have either the Measurement Studio Professional or Measurement Studio Enterprise package installed. This walkthrough requires the DAQ Assistant, which is not included in the Measurement Studio Standard package.

This walkthrough is designed to help you learn how to create an NI-DAQmx application by taking you through the following steps:

- **Setting up the project**—Using the Measurement Studio DAQ Application Wizard, you will create a new project that references the NI-DAQmx assembly and launches the DAQ Assistant to create an NI-DAQmx task.
- **Configuring your task**—Using the DAQ Assistant, you will interactively configure and save your task. The wizard then generates code to reflect your configuration settings. The wizard also generates a component that provides common operations for your task and integration with the Windows Forms designer.

- **Creating a custom user interface for the task**—Using the DAQ Component UI generation wizard, you will create a custom user interface that uses the DAQ component you created to automatically plot the DAQ signal.

Before you begin

The following components are required to complete this walkthrough:

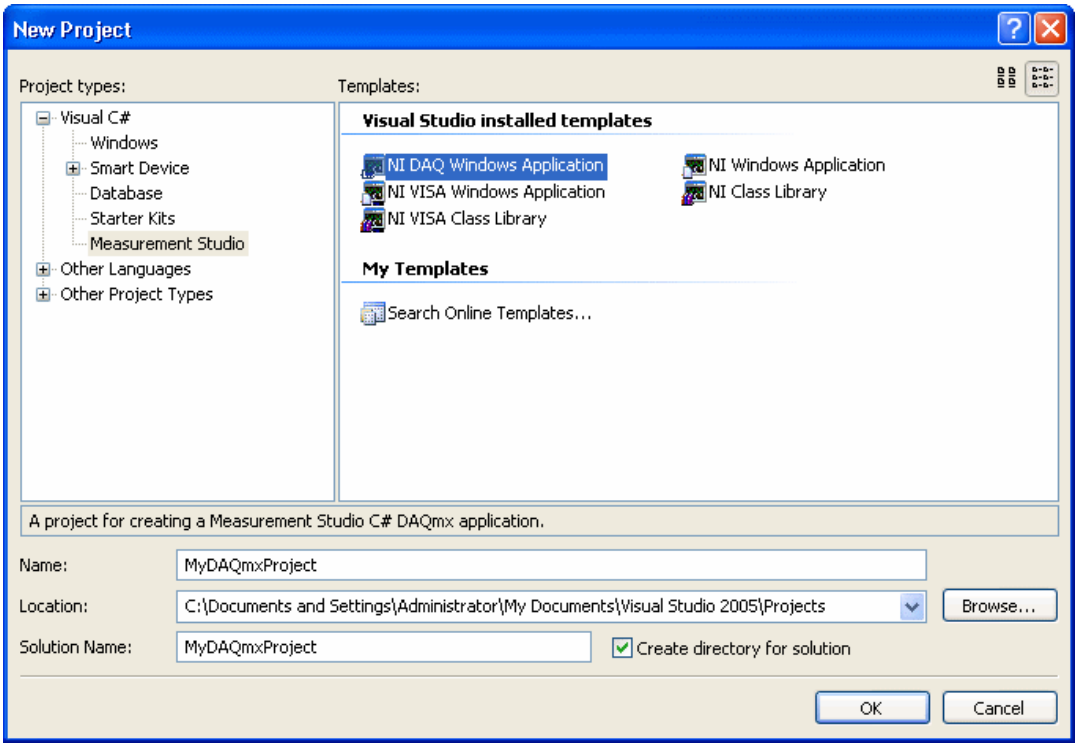
- Microsoft Visual Studio 2005 or Visual Studio 2008
- Measurement Studio 8.0.1 or later (Professional or Enterprise edition) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise edition) for Visual Studio 2008
- NI-DAQmx 8.1 or later for Visual Studio 2005 or NI-DAQmx 8.7.1 or later for Visual Studio 2008
- NI-DAQmx-supported DAQ device or simulated device

For information about installing and configuring your DAQ device, refer to the *DAQ Getting Started Guide*. You can also use simulation to complete this walkthrough. For information on how to create an NI-DAQmx simulated device, refer to *Creating NI-DAQmx Simulated Devices* in the *Measurement & Automation Explorer Help for NI-DAQmx*. To open this help, select **Start»All Programs»National Instruments»Measurement & Automation**. In Measurement & Automation Explorer (MAX), select **Help»Help Topics»NI-DAQmx»MAX Help for NI-DAQmx**. For the purposes of this walkthrough, the NI PCI-6280 device of the M Series DAQ family is recommended.

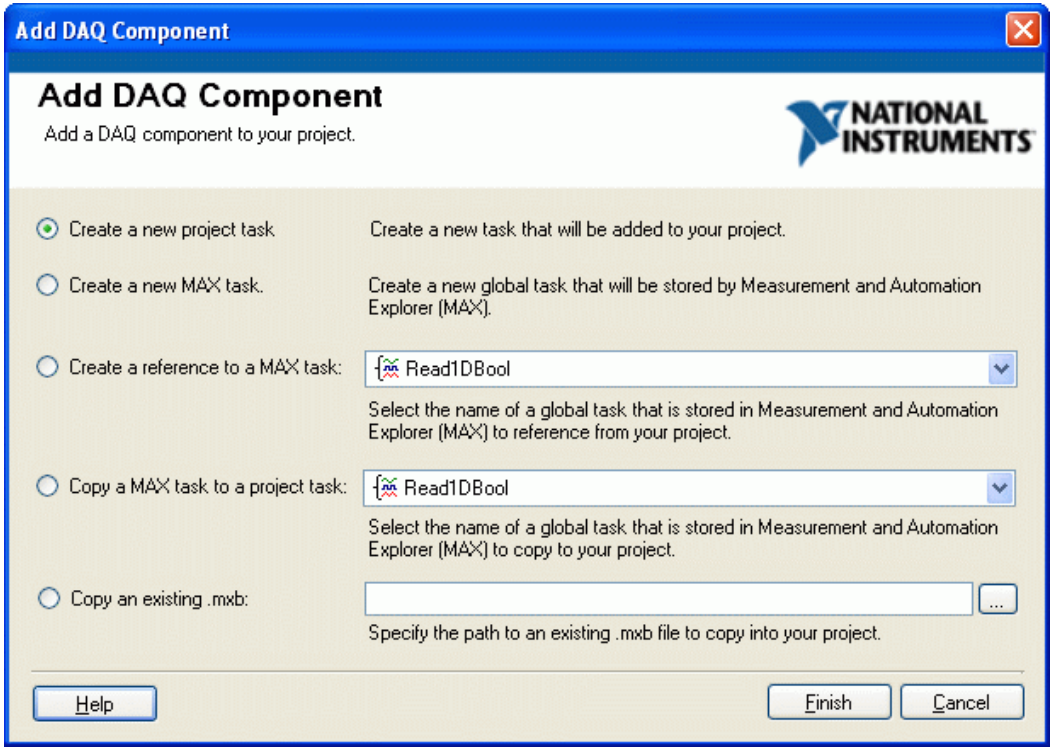
To set up the project

1. Open Visual Studio from **Start»All Programs»Microsoft Visual Studio 2005»Microsoft Visual Studio 2005**.
2. Select **File»New»Project**.The New Project dialog box launches.
3. In the Project types pane, expand the **Visual C#** or **Visual Basic** node, depending on which language you want to create the project in, and select **Measurement Studio**. Code generation works in both languages.

4. In the Templates pane, select **NI DAQ Windows Application**. Specify `MyDAQmxProject` for **Name** and specify a **Location** of your choice. Click **OK**. The Measurement Studio DAQ Application Wizard launches.



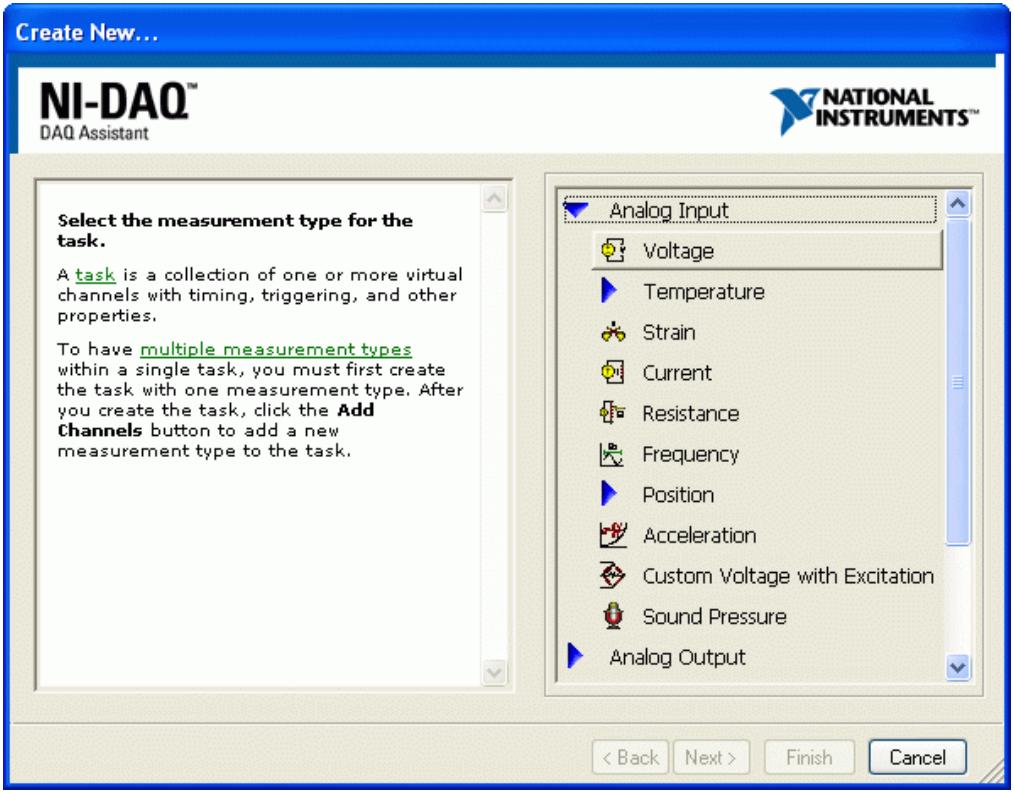
5. In the Add DAQ Component dialog box, you can choose to create a new project task, create a new MAX task, create a reference to a MAX task, copy a MAX task to a project task, or copy an existing `.mxb`. For this walkthrough, select **Create a new project task** and click **Finish**.



The Measurement Studio DAQ Application Wizard automatically sets up your data acquisition project and launches the DAQ Assistant.

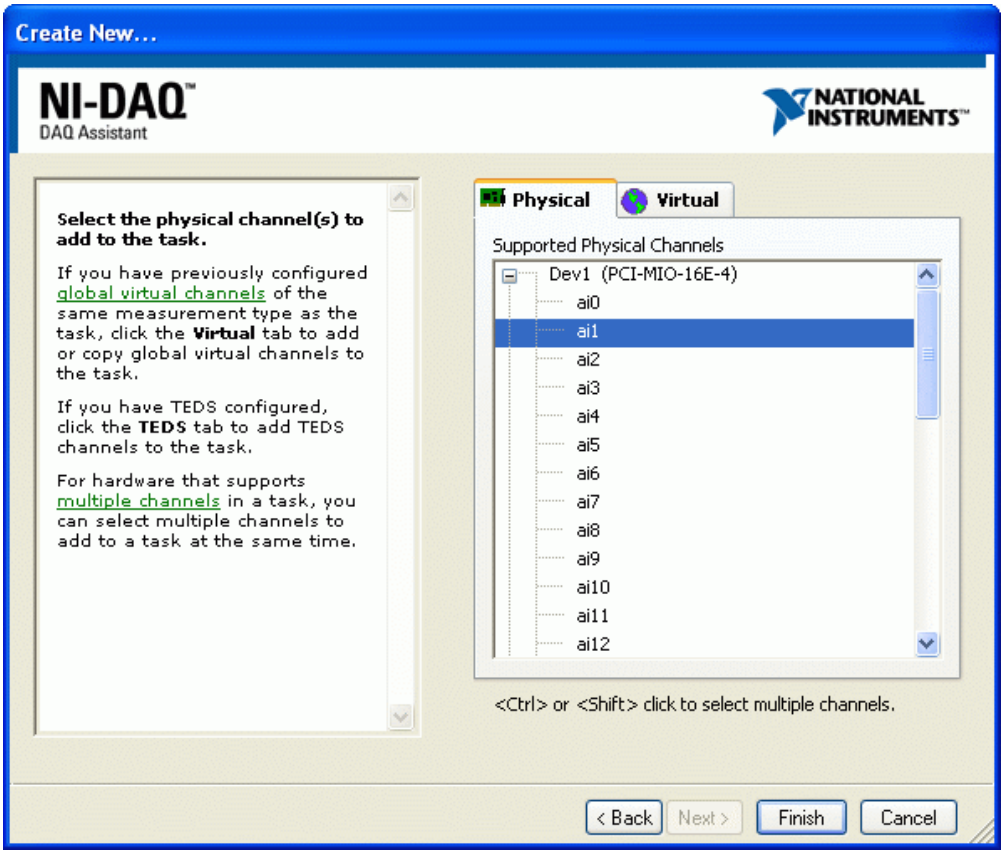
To configure your task

1. In the Create New dialog box of the DAQ Assistant, you can begin to interactively define your DAQ task. Select **Analog Input** as the measurement type for your task.
2. Next, select **Voltage**.

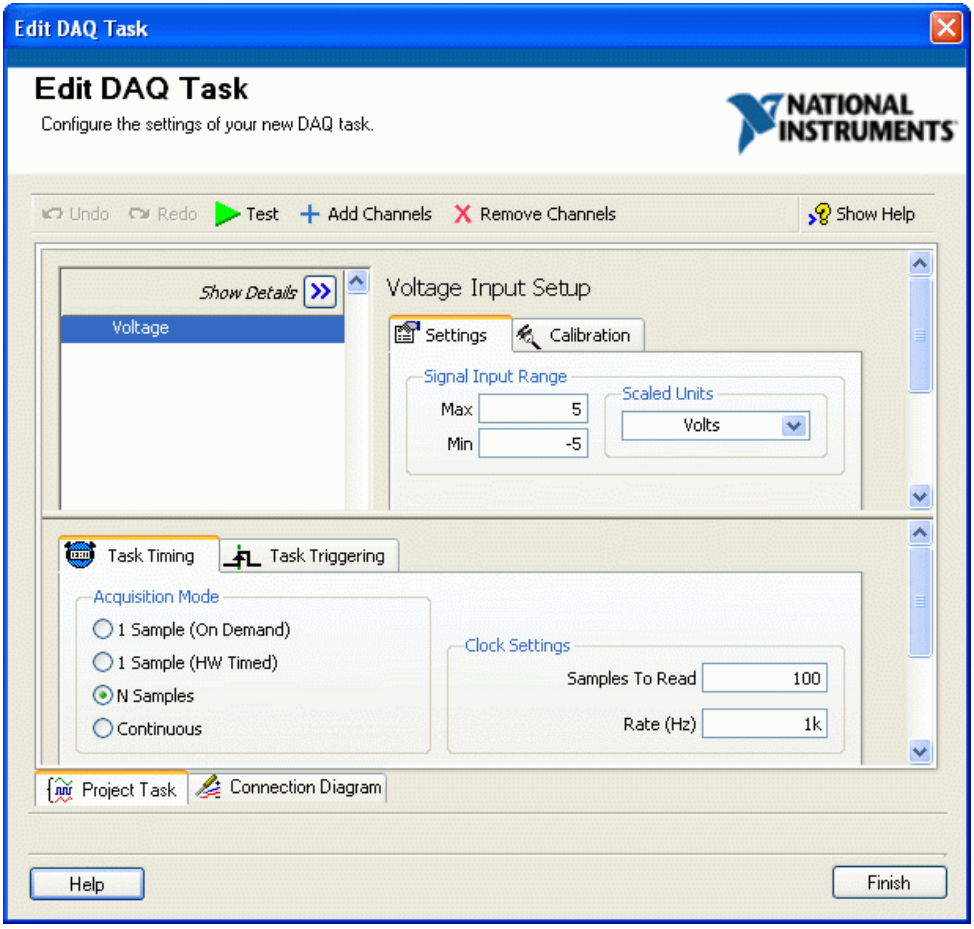


3. From the **Supported Physical Channels** tree in the **Physical** tab, select the physical channel, such as **ai1**, on the DAQ device to which you connected the voltage signal. Click **Finish**.

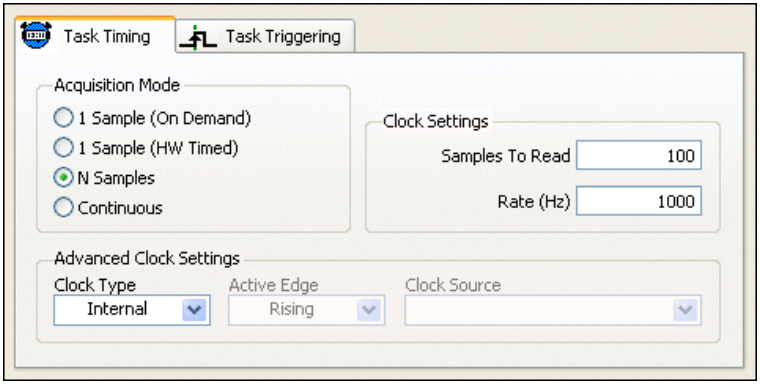
Note You can also use simulation in this walkthrough. For more information, refer to *Creating NI-DAQmx Simulated Devices* in the *Measurement & Automation Explorer Help for NI-DAQmx*.



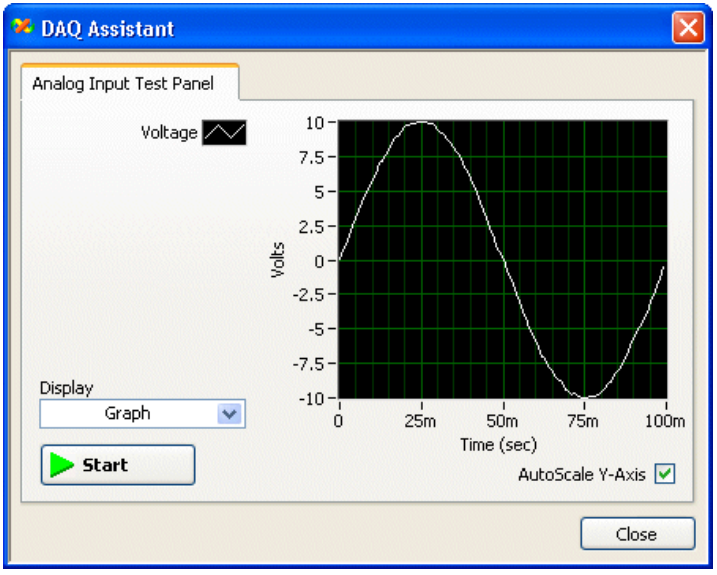
4. In the Edit DAQ Task dialog box, you can edit the configuration of your DAQ task. If the embedded DAQ Assistant help is not open by default, click the **Show Help** button in the upper-right corner of the window to display the help.



5. To complete the DAQ configuration, select the **N Samples** Acquisition Mode in the **Task Timing** tab. For more information on timing, refer to Timing in the *NI-DAQmx Help*.



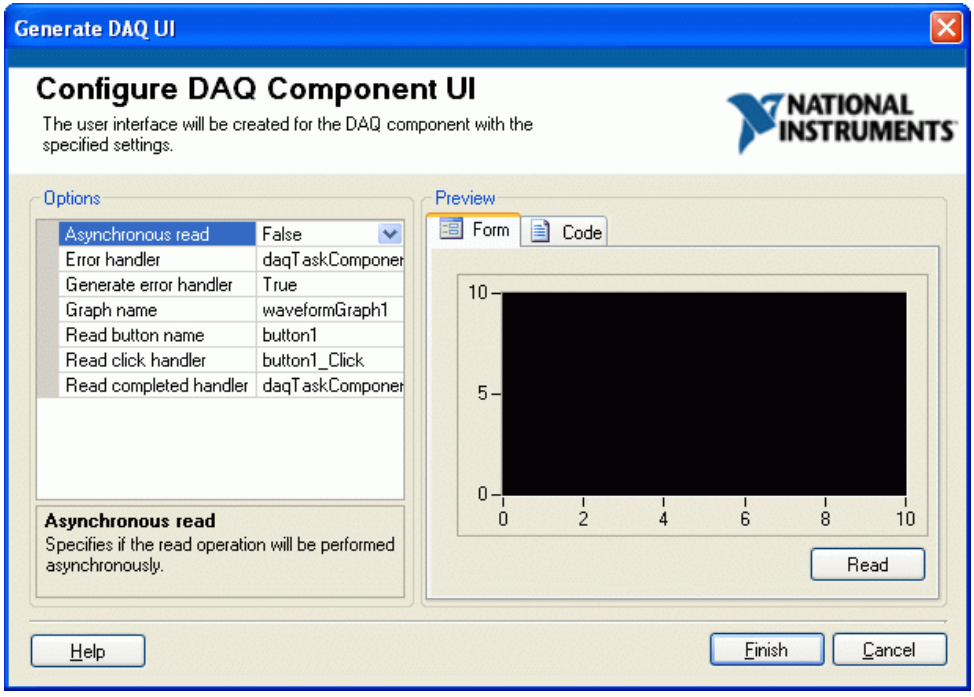
6. Next, click the **Test** button in the toolbar near the top of the Edit DAQ Task dialog box to launch the test panel for your task. The test runs automatically. You can use test panels in the DAQ Assistant to test the task and make sure you connected the signal properly. If necessary, you can modify the settings before any code is generated.



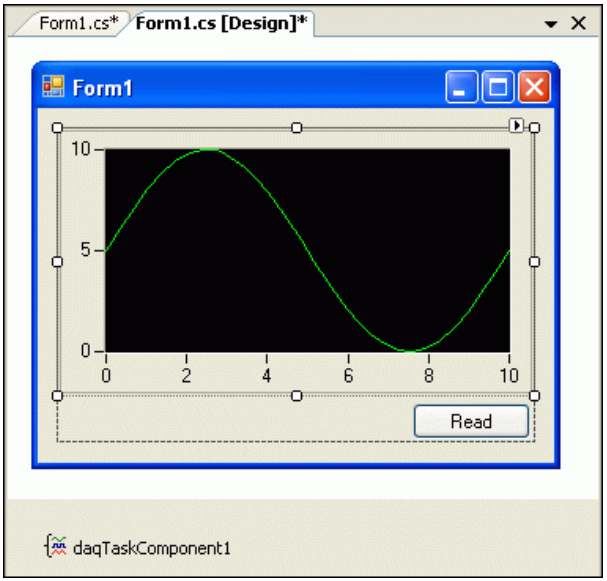
7. Click the **Close** button when you are done.
8. Click the **Finish** button in the Edit DAQ Task dialog box to complete the configuration of your DAQ task and to launch the Configure DAQ Component UI wizard.

To create a custom user interface for the task

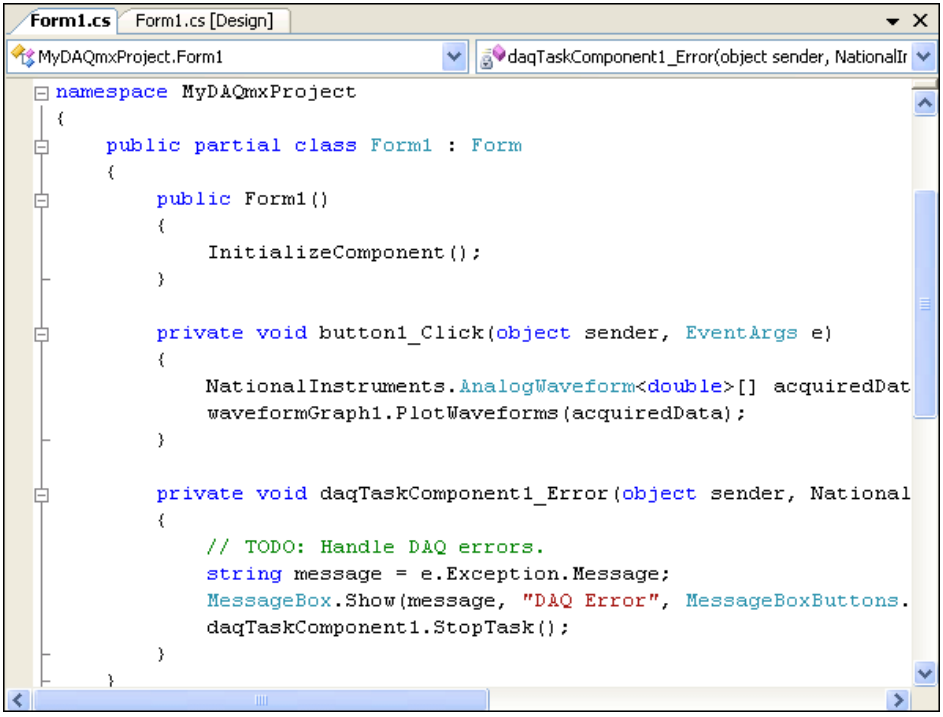
1. In the Configure DAQ Component UI wizard, you can customize and preview a user interface and code for your task.



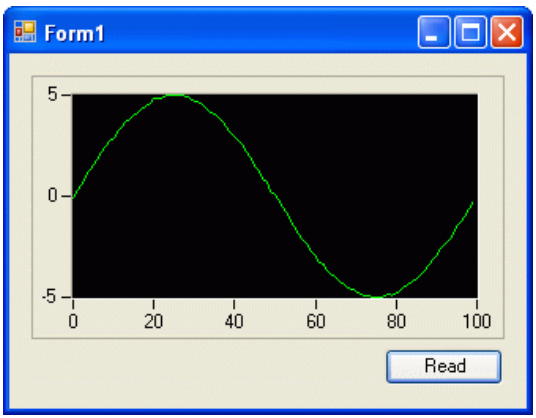
2. Click **Finish** to generate the task user interface in your project form.



The wizard also generates event handlers and code to acquire data and present it on your generated user interface.



3. Press <F5> to run the application.
4. After you have started the application, click the **Read** button to begin acquiring data from your DAQ device.



What's next
To learn more about tasks, channels, and other NI-DAQmx concepts, refer to the NI-DAQmx Help located at **Start»All Programs»National Instruments»NI-DAQ»NI-DAQmx Help**.

For more information about creating and using tasks in Measurement Studio, refer to *Using the Measurement Studio NI-DAQmx .NET Library* in the *NI Measurement Studio Help*.

You can also look at examples that ship with NI-DAQmx. Refer to *Measurement Studio NI-DAQmx .NET Examples* for the locations of these examples.

Walkthrough: Creating a Measurement Studio Instrument I/O Application

Note To complete this walkthrough, you must have either the Measurement Studio Professional or Measurement Studio Enterprise package installed. This walkthrough requires the Instrument I/O Assistant, which is not included in the Measurement Studio Standard package.

The National Instruments Instrument I/O Assistant organizes instrument communication for a serial, Ethernet, or GPIB instrument into ordered steps. This walkthrough is designed to help you learn how to build an instrument I/O application by taking you through the following steps:

- **Setting up the project**—Using the Measurement Studio VISA Windows Application project, you will create a new project that references the VisaNS assembly and launches the Instrument I/O Assistant to create a VisaNS task.
- **Performing a query on the instrument**—Using the Instrument I/O Assistant, you will write a command to an instrument and read the instrument response.
- **Displaying Instrument I/O Assistant data on your UI**—Using text box and button controls, you will create a Windows Forms application to display the Instrument I/O Assistant data.

Before you begin
The following components are required to complete this walkthrough:

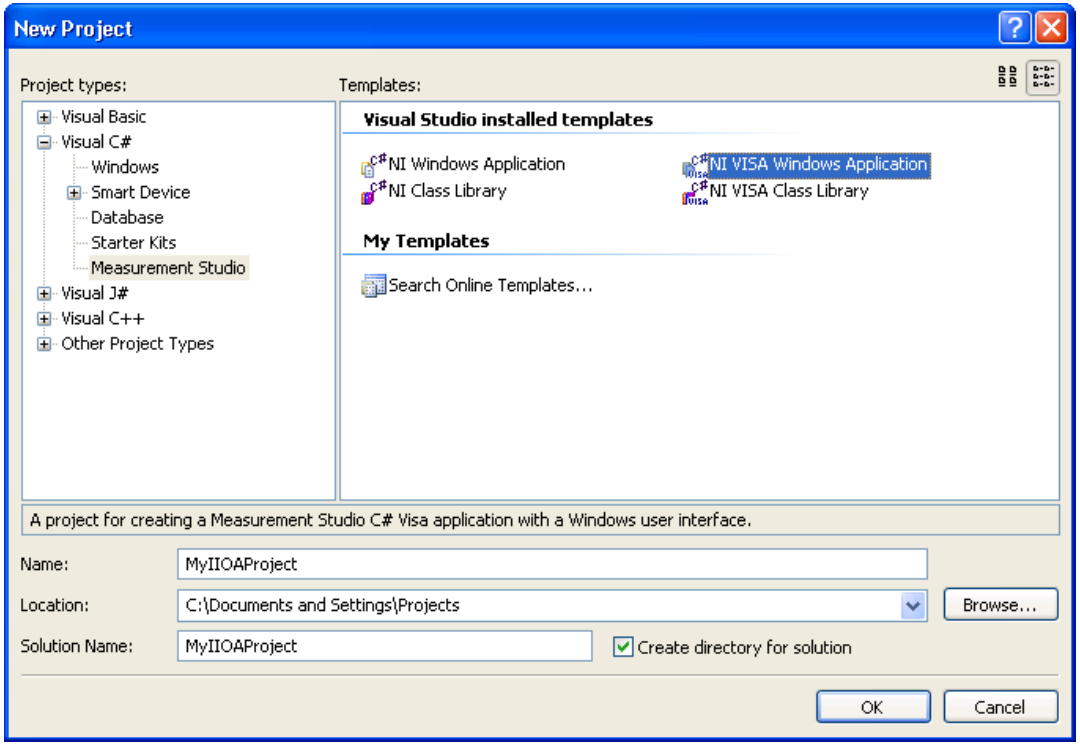
- Microsoft Visual Studio 2005 or Visual Studio 2008

- Measurement Studio 8.0.1 or later (Professional or Enterprise edition) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise edition) for Visual Studio 2008
- National Instruments Device Driver DVD
- Message-based instrument on a supported VISA bus, such as GPIB or Serial

Note For more information about the Instrument I/O Assistant, refer to the *Instrument I/O Assistant Help* by selecting the **Show Help** button inside the assistant.

Setting up the project

1. Open Visual Studio from **Start»All Programs»Microsoft Visual Studio 2008»Microsoft Visual Studio 2008**.
2. Select **File»New»Project**. The New Project dialog box launches.



3. In the Project Types pane, select **Measurement Studio** under Visual C# or Visual Basic, depending on which language you want to create the project in. This walkthrough refers to Visual C#, but you can follow the same process if you use Visual Basic .NET.
4. In the Templates pane, select **NI VISA Windows Application**. Specify `MyIIOAProject` for **Name** and select a **Location** of your choice.
5. Click **OK**. Your project opens in Visual Studio with a `VisaTask.mxb` file and references to `NationalInstruments.VisaNS`, `NationalInstruments.UI.WindowsForms`, and `NationalInstruments` created for you.
6. Select **View»Solution Explorer** to display the Solution Explorer. Double-click **VisaTask.mxb** in the Solution Explorer to launch the Instrument I/O Assistant.

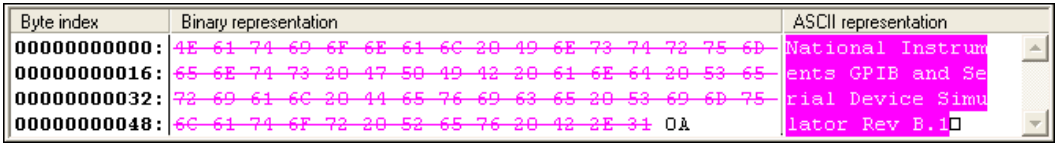
Performing a query on the instrument

Note This walkthrough was created using the NI Instrument Simulator. Any identification information or sample code generated for this device will be different depending on the instrument actually used.

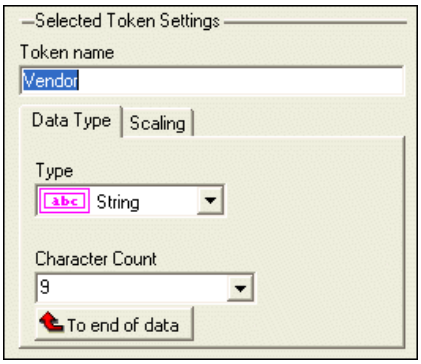
1. The **Select Instrument** step automatically appears in the **Step Sequence** window when you launch the Instrument I/O Assistant. Select the instrument you want to communicate with or the port to which your instrument is connected from the **Select an instrument** drop-down listbox.
2. Select **Add Step** and then select **Query and Parse** from the expanded list. You use a Query and Parse step to both write a command to an instrument and read the instrument's response.

3. Enter the command `*idn?` and click **Run this step**. The `*idn?` command is a standard instrument command for querying an instrument's identification information. If your instrument does not support the `*idn?` command, refer to the documentation for the instrument for more information about the instrument's command set.

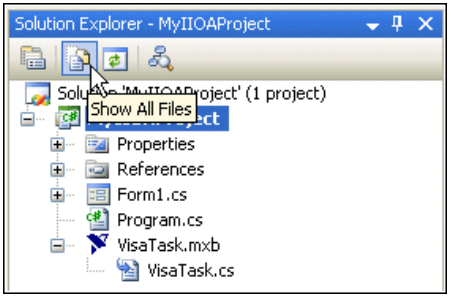
4. Click **Auto parse** to parse the instrument's response. The **Auto parse** button automatically parses binary block data and ASCII text. Refer to the *Parsing an Instrument Response* topic in the *Instrument I/O Assistant Help* for information about how the Assistant parses different data formats. Select the **Parsing Help** button to view the help.
5. If there are more than two tokens in the token list, remove them for this example. To remove a token, right-click on it in the response window and select **Remove**. The response window displays data in binary form, ASCII form, or binary form and ASCII form together. If there is only one token in the token list, split the token into two tokens for this example. Refer to *Parsing an Instrument Response* in the *Instrument I/O Assistant Help* for more information about how to manually parse the data into two tokens.



6. In the **Token name** text box, enter `Vendor` to rename the first token. You use this name to reference the token in your application. Ensure that the data type of the token is **String**. You specify the data type of the token using the **Type** drop-down list on the **Data Type** tab.



7. Select **Token2**, rename it to `Device`, and ensure that the data type for Token2 is **String**. To select Token2, left click on it within the Query and Parse step in the Step Sequence window on the left side of the Instrument I/O Assistant. Follow the instructions from step 6 to change the token name and to set the token data type.
8. Select **File»Save** to save your task.
9. Select **View»Solution Explorer** to display the Solution Explorer.
10. Click the **Show All Files** icon and expand the `VisaTask.mxb` node.

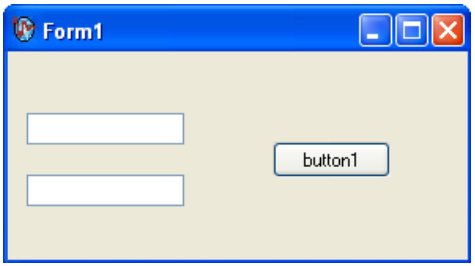


11. Double-click the `VisaNSTask1.cs` file to view the code that the Instrument I/O Assistant generated for you.

Displaying Instrument I/O Assistant data on your user interface

1. Double-click `Form1.cs` in the Solution Explorer to open your main application form.
2. Select **View»Toolbox** to display the Toolbox.
3. Expand the **All Windows Forms** group on the Toolbox.
4. Select the **Button** control and drag and drop it onto the form.

5. Select the **TextBox** control and drag and drop it onto the form. Repeat this step to add a second text box to the form. The following screenshot shows the controls on the form:

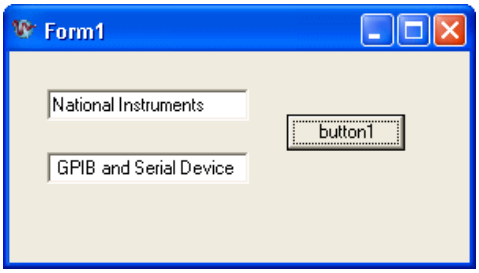


6. Double-click the **Button** control to display the `Form1` code, with the cursor inside the click event handler of the button control.
7. Add the following code to display the vendor and model name of your instrument in the text boxes.

```
[Visual Basic]
' Declare an instance of VisaTask
Dim myTask As New VisaTask()
Dim results As VisaTaskResults
'Display the data in the text boxes
results = myTask.Run()
textBox1.Text = results.Vendor
textBox2.Text = results.Device

[C#]
//Declare an instance of VisaTask
VisaTask myTask = new VisaTask();
//Display the data in the text boxes
VisaTaskResults results = myTask.Run();
textBox1.Text = results.Vendor;
textBox2.Text = results.Device;
```

Note Your sample code will be different depending on the instrument actually used.

8. Select **Debug»Start Without Debugging** to run the application.
9. Click the **Button** on the form to run the task. The following screenshot shows the controls on the form, with sample returned data.



Note Although this walkthrough only covers the use of a simple **Query and Parse** step, the Instrument I/O Assistant offers additional capabilities, such as independent **Write** and **Read and Parse** steps and advanced parsing capabilities. The following screenshot shows the IIOA's ability to scale and parse IEEE long definite block data.

What's next

For more information about creating and using VisaNS tasks in Measurement Studio, refer to *Using the Measurement Studio VisaNS .NET Library* and *Creating a Measurement Studio Instrument Control Application* in the *NI Measurement Studio Help*.

3

Measurement Studio .NET Class Libraries

This chapter provides overview information about the .NET class libraries included with Measurement Studio support for Visual Studio 2005 and Visual Studio 2008. Refer to the *Using the Measurement Studio .NET Class Libraries* section of the *NI Measurement Studio Help* for detailed information about these libraries. Refer to Chapter 2, *Creating Applications with Measurement Studio*, for step-by-step instructions on developing applications with these libraries.

Measurement Studio Support for Visual Studio .NET Class Library Overview

Measurement Studio provides .NET class libraries that you can use to develop complete measurement and automation applications in Visual Basic .NET and Visual C#.

Measurement Studio includes the following .NET class libraries:

- Analysis
- Common
- DataSocket
- Network Variable
- NI-488.2
- NI-DAQmx
- NI-IMAQ
- NI-IMAQdx
- NI-SCOPE
- NI-VISA
- NI Vision Development Module
- Technical Data Management Streaming (TDMS)
- User Interface

Refer to the following sections for information about each Measurement Studio .NET class library.

64-bit Support in Measurement Studio .NET Assemblies

Measurement Studio support for Visual Studio 2008 includes 64-bit support for Measurement Studio .NET assemblies. You can use Measurement Studio .NET assemblies in 64-bit applications to take advantage of the increased processing power and memory capabilities that are available to 64-bit applications.

Note Projects created from project templates included with Measurement Studio 8.6.1 or earlier will run as 32-bit unless you manually change the project settings to AnyCPU.

Measurement Studio support for Visual Studio 2008 includes 64-bit support for the following class libraries:

- Analysis
- Common
- Network Variable
- TDMS
- User Interface
- NI-DAQmx (8.9.5 and later versions)

Note Not all NI-DAQ hardware supported by NI-DAQmx is supported in 64-bit user mode. Refer to the *NI-DAQ Readme* for more information, installed at **Start»All Programs»National Instruments»NI-DAQ»NI-DAQ Readme**.

- NI-VISA (4.5.1 and later versions)
- NI-488.2 (2.7.1 and later versions)
- MAX (4.6 and later versions)

Measurement Studio support for Visual Studio 2008 does **not** include 64-bit support for the following class libraries:

- DataSocket .NET class libraries
- NI-SCOPE .NET class libraries
- NI-VISION, NI-IMAQ, and NI-IMAQdx .NET class libraries
- Modular instruments .NET wrappers for other NI drivers

Analysis

The Measurement Studio Analysis .NET class library is in the `NationalInstruments.Analysis` namespace. The Analysis class library includes a set of classes that provides various digital signal processing, signal filtering, signal generation, peak detection, and other general mathematical functionality. Use this library to analyze acquired data or to generate data. Additionally, the documentation for the Analysis class library includes analysis code snippets that you can copy and paste into an application and use immediately.

The functionality included in the Analysis library varies based on the Measurement Studio edition you purchase. Refer to the following sections for information about the Standard, Professional, and Enterprise Analysis class libraries.

Standard Analysis

The Standard Analysis class library, which ships with Measurement Studio Standard Edition, includes the sawtooth, sine, square, triangle, and basic function wave generators.

Professional Analysis

The Professional Analysis class library, which ships with Measurement Studio Professional Edition, includes the Standard Analysis functionality as well as the following functionality:

- Bessel, Chebyshev, Inverse Chebyshev, Windowed, Kaiser, and Elliptic Low, High, Bandpass, and Bandstop filters
- Signal processing functions such as convolution, deconvolution, correlation, decimation, integration, and differentiation
- FFT, Inverse FFT, Real FFT, Fast Hartley, Inverse Fast Hartley, Fast Hilbert, Inverse Fast Hilbert, DST, Inverse DST, DCT, and Inverse DCT transformations
- Linear algebra functions such as determinant, check positive definiteness, calculate dot product, and other various matrix functions
- Scaled and unscaled windowing classes
- Common statistical functions such as mean, median, mode, and variance
- Exponential, linear, and polynomial curve fitting functions
- Signal generation functions

Enterprise Analysis

The Enterprise Analysis class library, which ships with Measurement Studio Enterprise Edition, includes the Standard and Professional Analysis functionality as well as the following advanced functionality:

- EquiRipple filters
- Linear algebra functions such as forward and back substitution, LU factorization, Cholesky factorization, Schur deomposition, and Hessenberg decomposition
- Probability and analysis of variance
- Sinc, impulse, pulse, ramp, and chirp patterns
- General least square curve fit, power fit, log fit, Gauss fit, cubic spline fit, and interpolation functions
- Measurement functions such as transition measurements, pulse measurements, cycle RMS average, and single tone and multiple tone information
- Special functions

Refer to Table 3-1 to determine the type of measurements available in the Professional and Enterprise Analysis .NET libraries.

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages

Analysis .NET Library	Professional Package	Enterprise Package
Measurements		
AC and DC Estimator		✓
Amplitude and Phase Spectrum		✓
Auto Power Spectrum		✓
Cross Power Spectrum		✓
Cycle RMS Average		✓
Harmonic Analyzer		✓
Harmonic Analyzer Using Signal		✓
Impulse Response Function	✓	✓
Network Functions (avg)	✓	✓
Power and Frequency Estimate		✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Pulse Measurements		✓
Scaled Time Domain Window		✓
Single Tone Information		✓
Multiple Tone Information		✓
Spectrum Unit Conversion		✓
State Levels		✓
Transfer Function		✓
Transition Measurements		✓
Signal Generation		
Arbitrary Wave	✓	✓
Chirp Pattern		✓
Gaussian White Noise	✓	✓
Impulse Pattern		✓
Pulse Pattern		✓
Ramp Pattern		✓
Sawtooth Wave		✓
Sinc Pattern		✓
Sine Pattern	✓	✓
Sine Wave	✓	✓
Square Wave	✓	✓
Triangle Wave	✓	✓
Uniform White Noise	✓	✓
Windowing		
Blackman Window	✓	✓
Blackman-Harris Window	✓	✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Blackman-Nuttall Window	✓	✓
Cosine Tapered Window	✓	✓
Dolph-Chebyshev Window	✓	✓
Exact Blackman Window	✓	✓
Exponential Window	✓	✓
Flat Top Window	✓	✓
Force Window	✓	✓
Gauss Window	✓	✓
General Cosine Window	✓	✓
Hamming Window	✓	✓
Hanning Window	✓	✓
Kaiser-Bessel Window	✓	✓
Scaled Time Domain Windows	✓	✓
Symmetric Time Domain Windows	✓	✓
Triangle Window	✓	✓
Filters		
Bessel	✓	✓
Butterworth	✓	✓
Cascade	✓	✓
Chebyshev	✓	✓
Elliptic	✓	✓
Equiripple		✓
FIR	✓	✓
FIR Windowed	✓	✓
IIR Cascade	✓	✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
IIR	✓	✓
Inverse Chebyshev	✓	✓
Kaiser	✓	✓
Signal Processing		
Autocorrelation	✓	✓
Convolution	✓	✓
Cross Power	✓	✓
Cross Correlation	✓	✓
Decimate	✓	✓
Deconvolution	✓	✓
Derivative x(t)	✓	✓
Discrete Cosine Transform	✓	✓
Discrete Sine Transform	✓	✓
Fast Hilbert Transform	✓	✓
Fast Hartley Transform	✓	✓
Integral x(t)	✓	✓
Inverse Real and Complex Fast Fourier Transform (FFT)	✓	✓
Inverse Fast Hilbert Transform	✓	✓
Inverse Fast Hartley Transform	✓	✓
Peak Detection	✓	✓
Power Spectrum	✓	✓
Pulse Parameters	✓	✓
Real and Complex FFT	✓	✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Threshold Peak Detector	✓	✓
Unwrap Phase	✓	✓
Linear Algebra		
Back Transform Eigen Vectors		✓
Backward Substitution		✓
Cholesky Factorization		✓
Complex Back Transform Eigen Vectors		✓
Complex Cholesky Factorization		✓
Complex Determinant	✓	✓
Complex Dot Product	✓	✓
Complex Eigen Vectors and Eigen Values		✓
Complex General Eigen AB		✓
Complex Hessenberg Decomposition		✓
Complex Inverse Matrix		✓
Complex Linear Equations		✓
Complex LU Factorization		✓
Complex Matrix Balance		✓
Complex Matrix Condition Number	✓	✓
Complex Matrix Norm	✓	✓
Complex Matrix Rank	✓	✓
Complex Outer Product	✓	✓
Complex Pseudo Inverse Matrix	✓	✓
Complex QR Factorization		✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Complex QR Factorization with Pivot Matrix		✓
Complex QR Factorization with Pivot Vector		✓
Complex QZ Decomposition		✓
Complex Schur Decomposition		✓
Complex Solve Linear Equations (Multiple Right Hand)		✓
Complex Solve Linear Equations (Single Right Hand)		✓
Complex SVD Factorization		✓
Complex Vector Norm		✓
Determinant	✓	✓
Dot Product	✓	✓
Forward Substitution		✓
General Eigen AB		✓
Hessenberg Decomposition		✓
Inverse Matrix	✓	✓
Linear Equations		✓
LU Factorization		✓
Matrix Balance		✓
Matrix By Vector Multiplication	✓	✓
Matrix Condition Number	✓	✓
Matrix Multiplication	✓	✓
Matrix Norm	✓	✓
Matrix Rank	✓	✓
Outer Product	✓	✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Pseudo Inverse Matrix	✓	✓
QR Factorization		✓
QR Factorization with Pivot Matrix		✓
QR Factorization with Pivot Vector		✓
QZ Decomposition		✓
Schur Decomposition		✓
Solve Linear Equations (Multiple Right Hand)		✓
Solve Linear Equations (Single Right Hand)		✓
Special Matrix	✓	✓
SVD Factorization		✓
Test Positive Definite Matrix	✓	✓
Trace	✓	✓
Transpose	✓	✓
Array and Numeric Operations		
1D and 2D Array Arithmetic	✓	✓
1D and 2D Linear Evaluation	✓	✓
1D and 2D Polynomial Evaluation	✓	✓
1D Polar to Rectangular	✓	✓
1D Rectangular to Polar	✓	✓
Complex Number Arithmetic	✓	✓
Find Polynomial Roots	✓	✓
Scale 1D and 2D	✓	✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Curve Fitting		
Cubic Spline Fit		✓
Exponential Fit	✓	✓
Exponential Fit Interval		✓
Gauss Fit		✓
Gauss Fit Interval		✓
General Least Squares Linear Fit		✓
General Polynomial Fit	✓	✓
Goodness of Fit		✓
Linear Fit	✓	✓
Linear Fit Interval		✓
Logarithm Fit		✓
Logarithm Fit Interval		✓
Nonlinear Fit		✓
Power Fit		✓
Power Fit Interval		✓
Remove Outliers		✓
Statistics		
1D, 2D, and 3D ANOVA		✓
Chi-Square Distribution		✓
erf(x) and erfc(x)		✓
F-Distribution		✓
Histogram	✓	✓
Inverse Chi-Square Distribution		✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Inverse F-Distribution		✓
Inverse Normal Distribution		✓
Inverse T-Distribution		✓
Mean	✓	✓
Median and Mode	✓	✓
Moment about Mean	✓	✓
Normal Distribution		✓
Polynomial Interpolation		✓
Root-Mean_Square (RMS)	✓	✓
Spline Interpolant		✓
Spline Interpolation		✓
Standard Deviation	✓	✓
T-Distribution		✓
Variance		✓
Special Functions		
Airy		✓
Bessel 1st		✓
Bessel 2nd		✓
Beta		✓
Complimentary Gamma		✓
Cosine Integral		✓
Dawson’s Integral		✓
Dilogarithm		✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Elliptic 1st		✓
Elliptic 2nd		✓
Exponential Integral		✓
Factorial		✓
Fresnel Integrals		✓
Gamma		✓
Gauss Hypergeometric		✓
Hyperbolic Cosine Integral		✓
Hyperbolic Sine Integral		✓
Incomplete Beta		✓
Incomplete Elliptic 1st		✓
Incomplete Elliptic 2nd		✓
Incomplete Gamma		✓
Jacobian Elliptic Function		✓
Kelvin 1st		✓
Kelvin 2nd		✓
Kummer		✓
Logarithm of Factorial		✓
Modified Bessel 1st		✓
Modified Bessel 2nd		✓
Parabolic Cylinder		✓
Psi		✓
Sine Integral		✓
Spherical Bessel 1st		✓
Spherical Bessel 2nd		✓

Table 3-1. Analysis .NET Library Measurement Types included in the Professional and Enterprise Packages (Continued)

Analysis .NET Library	Professional Package	Enterprise Package
Stirling		✓
Struve		✓
Tricomi		✓
Zeta		✓

Tip For more information about analyzing or generating data with the Analysis class library, refer to the *Using the Measurement Studio Analysis .NET Library* topic in the *NI Measurement Studio Help*. For more information about the functionality included in the Analysis class library, visit `ni.com/analysis`.

Common

The Measurement Studio Common .NET class library is in the `NationalInstruments` namespace. The Common class library provides a set of classes that facilitates the exchange of data between the acquisition, analysis, and user interface portions of your application. The Common class library includes the following features:

- A `ComplexDouble` data type. This data type represents a complex number of type `Double` that is composed of a real part and an imaginary part.
- A `DigitalWaveform` data type. This data type represents a set of digital states that are grouped by samples or signals.
- A `ComplexWaveform` data type. This data type represents an analog signal that varies over time and is composed of complex data values.
- An `AnalogWaveform` data type. This data type represents an analog signal that varies over time.
- A `DataConverter` class that converts data from one data type to another data type, such as converting an array of integers to an array of doubles.
- An `EngineeringFormatInfo` class that defines a custom formatter to format numeric values as strings with engineering notation and International System of Units (SI) prefixes and symbols.
- A `PrecisionDateTime` structure that you can use to represent an instance in time, typically expressed as a date and time of day, that is accurate to the nearest 2^{-64} second.

- A `PrecisionTimeSpan` structure that you can use to represent a time interval, or duration of time, that is accurate to the nearest 2^{-64} second.
- A `PrecisionWaveformTiming` class that you can use to represent the timing of an analog or digital waveform that is accurate to the nearest 2^{-64} second.
- An `AnalogWaveformCollection` class that contains a strongly typed collection of `AnalogWaveform<TData>` objects; one object for each channel and record combination. You can access these objects through the 1D indexer or the 2D indexer.

Tip For more detailed information about the Common class library, refer to the *NationalInstruments* section in the *NI Measurement Studio Help*.

Data Transfer

You can use the NetworkVariable class library or the DataSocket class library to transfer live measurement data between applications over a network. You can use NetworkVariable or DataSocket to exchange different types of data between Measurement Studio, LabVIEW, LabWindows/CVI, and other applications that support NI-Publish Subscribe Protocol (`psp:`). NetworkVariable is the preferred method for transferring data between these applications, and, in these cases, NetworkVariable supersedes DataSocket. You can also use NetworkVariable and DataSocket to exchange different types of data between OLE for Process Control (`opc:`) servers. Exchanging data between Measurement Studio applications and OPC servers with NetworkVariable requires the LabVIEW DSC Run-Time System. Use DataSocket to communicate directly with an OPC server.

Network Variable

The Measurement Studio Network Variable .NET class library includes three namespaces: `NationalInstruments.NetworkVariable`, `NationalInstruments.NetworkVariable.WindowsForms`, and `NationalInstruments.NetworkVariable.WebForms`. You use the Network Variable class library to transfer live measurement data between applications and servers over the network. You use `NationalInstruments.NetworkVariable.WindowsForms` and `NationalInstruments.NetworkVariable.WebForms` data sources to expose Network Variable data items that you can bind to properties of a Windows Forms or a Web Forms control.

Use the features in the Network Variable class library to perform the following operations:

- Exchange different types of data between Measurement Studio, LabVIEW, LabWindows/CVI, and other applications that support NI-Publish Subscribe Protocol (`psp:`) and OLE for Process Control (`opc:`) servers. Exchanging data between Measurement Studio applications and OPC servers requires LabVIEW DSC.

Note Measurement Studio and LabWindows/CVI refer to variables as network variables and LabVIEW refers to variables as shared variables. However, you can read to and write from Measurement Studio and LabWindows/CVI network variables with LabVIEW shared variables.

- Use Windows Forms and Web Forms data sources to expose Network Variable data items that you can bind to properties of a Windows Forms or a Web Forms control.
- Use the `NationalInstruments.NetworkVariable.ServerProcess`, `NationalInstruments.NetworkVariable.ServerProcessInfo`, `NationalInstruments.NetworkVariable.ServerVariable`, and `NationalInstruments.NetworkVariable.ServerVariableInfo` classes to explicitly create network variables. You can configure the description and type for explicitly created network variables.
- Use the Network Variable Browser dialog box to quickly locate and select data items on other computers and servers. The Browser Dialog is included in the **WindowsForms** class.

Tip For more detailed information about the Network Variable class library, refer to the *Using the Measurement Studio Network Variable .NET Library* section in the *NI Measurement Studio Help*.

DataSocket

The Measurement Studio DataSocket .NET class library is in the `NationalInstruments.Net` namespace. Use the DataSocket class library to transfer live measurement data over the Internet or an intranet, between applications on the same computer, and to and from files. Use the classes in the DataSocket class library to perform the following operations:

- Read and write data between different data sources and targets.
- Use a single, simple API to communicate with several types of servers, including DataSocket Servers (`dstp:`), Web servers (`http:`), file

transfer protocol servers (`ftp:`), file systems (`file:`), and OLE for Process Control (`opc:`) servers.

- Specify data sources and targets using a URL, the same way you access Web pages in a Web browser.
- Use DataSocket Transfer Protocol (DSTP) to exchange different types of data.
- Expose DataSocket data items as data sources that you can bind to properties of a Windows Forms control.
- Interactively browse to quickly locate and select data items on other computers and servers.

Note The Measurement Studio DataSocket .NET class libraries do not include 64-bit support.

Tip For more detailed information about the DataSocket class library, refer to the *Using the Measurement Studio DataSocket .NET Library* section in the *NI Measurement Studio Help*.

NI-488.2

The Measurement Studio NI-488.2 .NET class library is in the `NationalInstruments.NI4882` namespace. This class library is included when you install the NI-488.2 driver. The NI-488.2 driver is available at `ni.com/downloads` and on the NI Device Drivers DVD that is included with Measurement Studio. The NI-488.2 class library includes a set of classes for communicating with GPIB instruments, controlling GPIB devices, and acquiring GPIB status information. Use this library to design code that communicates with and controls instruments on a GPIB interface. Use the NI-488.2 class library to configure and communicate with GPIB devices using the `Device` and `Board` classes.

Tip For more information about the NI-488.2 class library, refer to the *Using the Measurement Studio NI-488.2 .NET Library* topic in the *NI Measurement Studio Help*. For more information about GPIB, visit `ni.com/gpib`.

NI-DAQmx

The Measurement Studio NI-DAQmx .NET class library is in the `NationalInstruments.DAQmx` namespace. This class library is included when you install the NI-DAQmx driver. The NI-DAQmx driver is available at `ni.com/downloads` and on the NI Device Drivers DVD that is included with Measurement Studio. Use the NI-DAQmx class library to communicate with and control NI data acquisition (DAQ) devices.

Note Some DAQ devices are not currently supported by the NI-DAQmx driver. Refer to the *NI-DAQ Readme* for a complete listing of supported hardware.

Use the NI-DAQmx class library to perform the following types of tasks:

- Analog signal measurement
- Analog signal generation
- Digital I/O
- Counting and timing
- Pulse generation
- Signal switching

Tip For more information about the NI-DAQmx class library, refer to the *Using the Measurement Studio NI-DAQmx.NET Library* topic in the *NI Measurement Studio Help*. For more information about DAQ, visit `ni.com/dataacquisition`.

NI-IMAQ

You can use the NI-IMAQ .NET class libraries to acquire images from NI frame grabbers, display images, and save images. These class libraries provide a .NET interface to the underlying driver API. NI-IMAQ 4.3 and later include support for .NET class libraries.

You can install NI-IMAQ from the NI Device Drivers DVD that is included with Measurement Studio. You can find NI-IMAQ under Vision Acquisition Software.

Tip For more information about the NI-IMAQ class library, refer to the *Measurement Studio Support for NI-IMAQ Overview* topic in the *NI Measurement Studio Help*. For more information about NI-IMAQ, visit `ni.com/vision`.

NI-IMAQdx

You can use the NI-IMAQdx .NET class libraries to acquire images from GigE, IEEE 1394, or USB cameras; display images; and save images. These class libraries provide a .NET interface to the underlying driver API. NI-IMAQdx 3.4 and later include support for .NET class libraries.

You can download NI-IMAQdx from the NI Device Drivers DVD that is included with Measurement Studio. You can find NI-IMAQdx under Vision Acquisition Software.

Tip For more information about the NI-IMAQdx class library, refer to the *Measurement Studio Support for NI-IMAQdx Overview* topic in the *NI Measurement Studio Help*. For more information about NI-IMAQdx, visit `ni.com/vision`.

NI-SCOPE

The .NET class libraries for NI-SCOPE include .NET APIs for NI-SCOPE, NI-TClk, and NI-ModInst instrument drivers. These class libraries provide a .NET interface to the underlying driver API. You can use the .NET class libraries to create and configure NI-SCOPE components programmatically and at design time.

For further information on NI-SCOPE .NET driver support and to download the NI-SCOPE .NET class libraries, refer to *NI-SCOPE .NET Driver Support* on `NI Developer Zone` at `ni.com/devzone`.

NI-VISA

The Measurement Studio NI-VISA .NET class library is in the `NationalInstruments.VisaNS` namespace. This class library is included when you install the NI-VISA driver. The NI-VISA driver is available at `ni.com/downloads`. The NI-VISA class library includes a set of classes that provides a rich, object-oriented interface to the NI-VISA driver. Use this library to quickly create bus-independent or bus-specific instrument control applications.

The NI-VISA class library supports formatted I/O operations, locking, event handling, and interface-specific extensions. With this class library you can access the functionality available in NI-VISA for communicating

with message-based and register-based instruments using the following interfaces:

- GPIB
- IEEE 1394
- PXI
- Serial (RS-232 and RS-485)
- TCP/IP
- USB
- VXI

Tip For information about creating a Measurement Studio NI-VISA application using the Instrument I/O Assistant, refer to the *Walkthrough: Creating a Measurement Studio Instrument I/O Application* section of Chapter 2, *Creating Applications with Measurement Studio*. For more information about NI-VISA, visit `ni.com/visa`.

.NET Support Included with Other Products

The following National Instruments product includes support for .NET class libraries; however, the NI Vision Development Module 2009 is not included with Measurement Studio. You must purchase the NI Vision Development Module 2009 or later to use the .NET assemblies. Refer to `ni.com/vision` for more information.

NI Vision Development Module

The NI Vision Development Module .NET assemblies contain all of the classes necessary to analyze, process, and display images. NI Vision Development Module 2009 and later includes support for .NET class libraries.

Technical Data Management Streaming (TDMS)

Technical Data Management Streaming (TDMS) is a file format based on the National Instruments TDM data model used to stream data to disk. You can use the TDMS .NET class library to describe, store, and read measurement data that is optimized for high-speed data streaming and post processing. Additionally, you can use the TDMS .NET class library to create files that you can use in LabVIEW, LabWindows/CVI, and DIAdem, and files created by these applications can be used by the TDMS .NET class library.

You can use the Measurement Studio TDMS .NET class library to perform the following operations:

- Read and write array data in a structured format from and to a `.tdms` file.
- Read and write analog waveform data or digital waveform data, including timing information, from and to a `.tdms` file.
- Use the `TdmsProperty` class to create custom properties for each level of the hierarchy by defining a name, data type, and value.

Tip For more detailed information about the TDMS class library, refer to the *Using the Measurement Studio TDMS .NET Library* section in the *NI Measurement Studio Help*.

TDM Excel Add-In

You can use the TDM Excel Add-In to load NI `.tdm` and `.tdms` files into Microsoft Excel. Use the toolbar from within Excel to choose which properties are loaded into Excel at the file, group, and channel levels, including custom properties.

To uninstall the TDM Excel Add-In, select **Start»Control Panel»Add or Remove Programs**, select **National Instruments Software** from the list, and click the **Change/Remove** button. Then select NI TDM Excel Add-in from the list, and click the **Remove** button.

Refer to NI Developer Zone, `zone.ni.com`, for more information about the TDM Excel Add-In.

User Interface

The Measurement Studio user interface controls are in the Windows Forms and Web Forms .NET class libraries. The following sections list the functionality included with the Measurement Studio Windows Forms and Web Forms controls.

Refer to the following table for the UI controls provided by Measurement Studio.

User Interface Controls	Windows Forms	Web Forms
Waveform graph	✓	✓
Scatter graph	✓	✓
Digital waveform graph	✓	✓
Complex graph	✓	✓
Intensity graph	✓	
Legend	✓	✓
Knob	✓	✓
Gauge	✓	✓
Meter	✓	✓
Slide	✓	✓
Thermometer	✓	✓
Tank	✓	✓
Numeric edit	✓	✓
Switch	✓	✓
LED	✓	✓
Property editor	✓	
Array controls	✓	
AutoRefresh control		✓
InstrumentControlStrip control	✓	

Windows Forms Controls

The Windows Forms .NET class library is in the `NationalInstruments.UI.WindowsForms` namespace. The Windows Forms class library encapsulates the following Measurement Studio user interface controls:

- Waveform graph
- Scatter graph
- Digital waveform graph
- Complex graph
- Intensity graph
- Legend
- Knob
- Gauge
- Meter
- Slide
- Thermometer
- Tank
- Numeric edit
- Switch
- LED
- Property editor
- Array controls

Use this class library to add measurement-specific user interface controls to your application. You can configure the controls programmatically at design time, through the Properties window in the Windows Forms Designer, or at run time with the property editor control. The following sections describe each of the Measurement Studio Windows Forms user interface controls.

Tip For more information about using the .NET user interface controls, refer to the *Using the Measurement Studio Windows Forms .NET Controls* section in the *NI Measurement Studio Help*.

Waveform Graph and Scatter Graph Controls

Use the Measurement Studio waveform graph and scatter graph controls, as shown in Figure 3-1, to display two-dimensional data on a Windows Forms user interface. Use the waveform graph to display two-dimensional linear data. You explicitly specify each value in one dimension and provide an initial value and interval to implicitly specify the values in the other dimension. Use the scatter graph to display two-dimensional linear or nonlinear data: you explicitly specify each value in both dimensions.

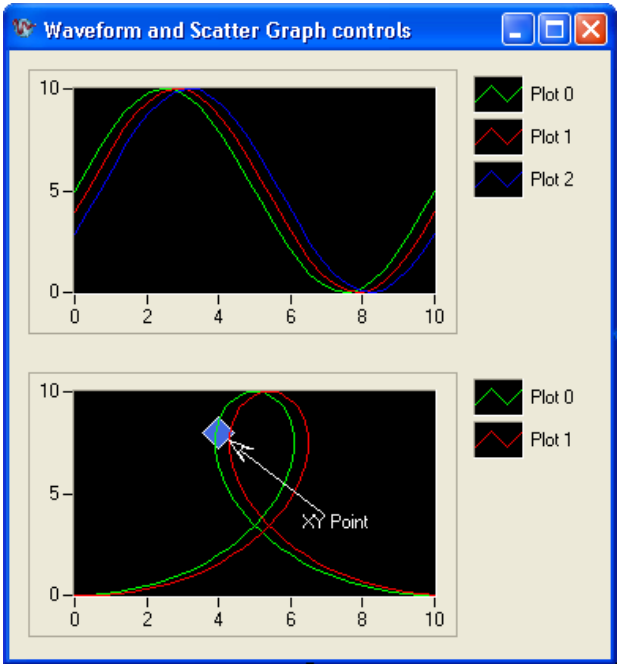


Figure 3-1. Waveform Graph Windows Forms Control with Cursors and Scatter Graph Windows Forms Control with XY Point Annotation; Both Graphs Have Corresponding Legends

With the waveform graph and scatter graph controls and the classes that interface with the controls, you can perform the following operations:

Plot Operations

- Plot and chart arrays of double-precision floating point values, analog waveforms, and complex waveforms.
- Configure a graph to contain multiple plots to show separate but related data on the same graph. You can configure a graph to automatically generate different colors for different plots.
- Draw lines or fills from a plot to an X value, Y value, or another plot.
- Specify plots in the scatter graph control as X and Y data. Specify plots in the waveform graph control as X or Y data and optionally with date and time scaling.
- Use the extensible plot and plot area drawing capabilities and events to customize the graph appearance.
- Use plot data tooltips to display X and Y coordinates when a user hovers the mouse over a data point.
- Create custom point and line styles for plots.
- Specify anti-aliased plots for plot lines.
- Calculate and display error bands.

Axis Operations

- Configure a graph to include multiple axes or independent ranges so that plot data fits the graph plot area.
- Configure the axis modes to: fixed; autoscaling, including autoscaling based on the visible data only; strip chart; or scope chart.
- Use logarithmic axes with configurable bases.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.
- Display origin lines.
- Display captions on the axis.
- Display grid lines.
- Position the axis to display on one or both sides of the graph's plot area.
- Configure major, minor, and custom divisions and origin lines.

Cursor Operations

- Use cursors to identify key points in plots and the plot area.
- Configure cursor snap modes as fixed, floating, nearest point, or to plot.
- Use cursor labels to display X and Y data coordinates in a customized format that the cursor crosshair points to, and customize the text font and colors of the label.
- Create custom point and line styles for cursors.
- Interactively move the cursor by clicking and dragging the vertical or horizontal crosshair or the center of the cursor.
- Programmatically move the cursor to the previous position, to the next position, or to a specified coordinate.
- Create custom mouse cursors programmatically or at design time using the mouse cursor style editor.

Annotation Operations

- Configure text labels, arrows, and drawing shapes to annotate a point anywhere in the plot area of the graph.
- Configure range area, text labels, and arrows to annotate a range in the plot area of the graph.
- Show tooltips configured to display data or other custom text.

Additional Operations

- Pan and zoom interactively, as well as programmatically.
- Copy the graph as a BMP, GIF, JPEG, or PNG image to the clipboard or a file.
- Perform hit testing of mouse cursor coordinates.
- Bind a plot to a data source on the waveform graph.

Tip For more information about using the waveform and scatter graph controls, refer to the *Using the Measurement Studio Windows Forms Scatter and Waveform Graph .NET Controls* section in the *NI Measurement Studio Help*.

Digital Waveform Graph Control

Use the Measurement Studio digital waveform graph control, as shown in Figure 3-2, to display `DigitalWaveform` data on a Windows Forms user interface.

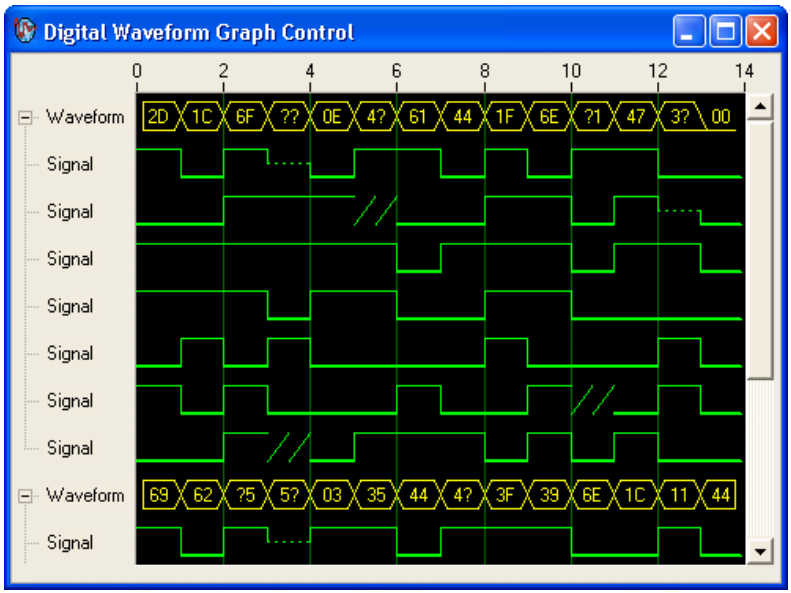


Figure 3-2. Digital Graph Windows Forms Control

With the digital waveform graph control and the classes that interface with the control, you can perform the following operations:

Plot Operations

- Plot digital waveform data. Data values can represent up to eight different digital states.
- Configure plot labels on the y-axis.
- Configure plot templates to customize plots that are implicitly created from plotted data.
- Specify anti-aliased digital plots.
- Expand and collapse signal plots interactively or programmatically.
- Display tooltips.

Waveform Sample and Signal State Operations

- Simultaneously display waveforms and signals or display signals only.
- Create custom waveform sample and signal state styles.
- Configure the appearance of sample and state labels.
- Create custom waveform sample and signal state labels.

Axis Operations

- Configure the axis modes to fixed, exact autoscaling, or loose autoscaling.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.
- Display captions on the axis.
- Display grid lines.
- Position the axis to display on one or both sides of the plot area of the graph.
- Configure major, minor, and custom divisions.

Additional Operations

- Display data in sample or time mode.
- Perform hit testing of mouse cursor coordinates.
- Pan with scroll bars.
- Configure the style and mode of scroll bars.
- Create custom scroll bars.
- Pan and zoom interactively and programmatically.
- Copy the graph as a BMP, GIF, JPEG, or PNG image to the clipboard or a file.
- Create custom mouse cursors programmatically or at design time using the mouse cursor style editor.

Tip For more information about using the digital waveform graph control, refer to the *Using the Measurement Studio Windows Forms Digital Waveform Graph .NET Control* section in the *NI Measurement Studio Help*.

Complex Graph Control

Use the Measurement Studio complex graph control, as shown in Figure 3-3, to display `ComplexDouble` data on a Windows Forms user interface. A `ComplexDouble` consists of a real part and an imaginary part. You can use a waveform graph to plot complex waveform data.

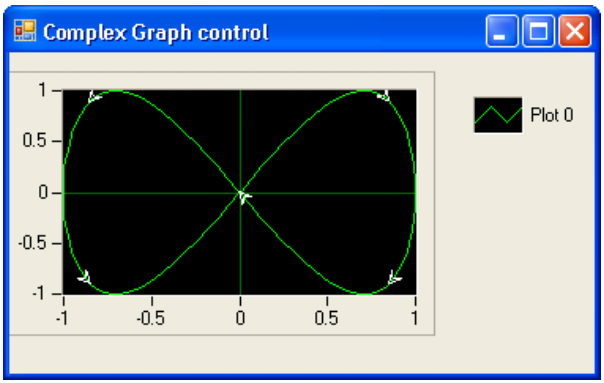


Figure 3-3. Complex Graph Windows Forms Control

With the complex graph control and the classes that interface with the control, you can perform the following operations:

Plot Operations

- Plot and chart `ComplexDouble` data.
- Configure a graph to contain multiple plots to show separate but related data on the same graph. You can configure a graph to automatically generate different colors for different plots.
- Draw lines or fills from a plot to an X value, Y value, or another plot.
- Use the extensible plot and plot area drawing capabilities and events to customize the graph appearance.
- Configure the plot to display arrows. The arrows indicate the direction of the complex data.
- Create custom point and line styles for plots.
- Specify anti-aliased plots for plot lines.
- Calculate and display error bands
- Display tooltips

Axis Operations

- Configure a graph to include multiple axes or independent ranges so that plot data fits the graph plot area.
- Configure the axis modes to: fixed; autoscaling, including autoscaling based on the visible data only; strip chart; or scope chart.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.

- Display origin lines and grid lines.
- Configure major, minor, and custom divisions and origin lines.
- Position the axis to display on one or both sides of the plot area of the graph.
- Display captions on the axis.

Cursor Operations

- Use cursors to identify key points in plots and the plot area.
- Configure cursor snap modes as fixed, floating, nearest point, or to plot.
- Use cursor labels to display X and Y data coordinates that the cursor crosshair points to, and customize the text font and colors of the label.
- Create custom point and line styles for cursors.
- Configure the graph to display cursors that are used to determine the real, imaginary, magnitude, and phase data coordinates of a point on the plot area.
- Create custom mouse cursors programmatically or at design time using the mouse cursor style editor.

Annotation Operations

- Configure text labels, arrows, and drawing shapes to annotate a point anywhere in the plot area of the graph.
- Configure range area, text labels, and arrows to annotate a range in the plot area of the graph.
- Annotate points and ranges of real, imaginary, and magnitude values.
- Annotate and label a range of magnitude values for a particular phase.

Additional Operations

- Pan and zoom interactively.
- Copy the graph as a BMP, GIF, JPEG, or PNG image to the clipboard or a file.

Tip For more information about using the complex graph control, refer to the *Using the Measurement Studio Windows Forms Complex Graph .NET Control* section in the *NI Measurement Studio Help*.

Intensity Graph Control

Use the intensity graph to plot 3D data on a 2D plot area using color to represent the third dimension. The intensity graph accepts a 2D array of data, where each data value is rendered as a block of color on the graph. The data value to color mapping can be specified by the user.

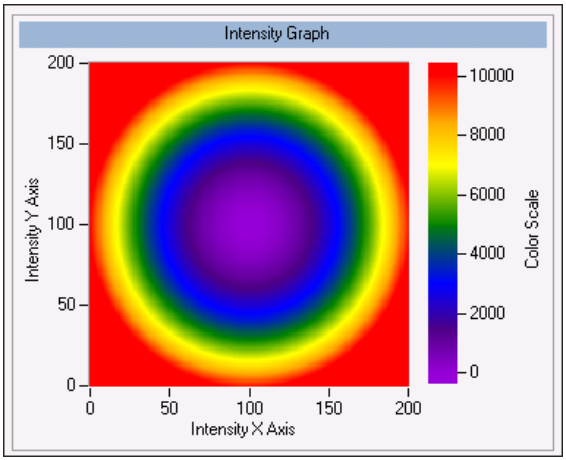


Figure 3-4. Intensity Graph Windows Forms Control

With the intensity graph control and the classes that interface with the control, you can perform the following operations:

Plot Operations

- Plot and chart data using Plot, PlotXAppend, and PlotYAppend methods.
- Interpolate data for every pixel from the input data to get better visual detail when the set of input data is sparse.
- Configure a graph to contain multiple plots to show separate but related data on the same graph.
- Use the extensible plot and plot area drawing capabilities and events to customize the graph appearance.

ColorScale Operations

- Configure the mapping from value to color for plotting.
- Configure a graph to include multiple color scales with independent color mappings for different plots.
- Configure major, minor, and custom divisions.

Axis Operations

- Configure a graph to include multiple axes with independent ranges to fit disparate data in a single graph plot area.
- Configure the axis modes to fixed, exact autoscaling, loose autoscaling, strip chart, or scope chart.
- Configure major, minor, custom divisions, and origin lines.

Additional Operations

- Use cursors and annotations to annotate points or regions in the plot area.
- Pan, zoom, move cursor, and move annotation caption interactively.
- Interactively change the range of an axis or color scale at run time by clicking on the end labels.
- Copy the graph as a BMP, GIF, JPEG, or PNG image to the clipboard or a file.
- Derive from and extend base classes to allow for easy programmatic customizability.

Tip For more information about using the intensity graph control, refer to the *Using the Measurement Studio Windows Forms Intensity Graph .NET Control* section in the *NI Measurement Studio Help*.

Legend Control

Use the Measurement Studio legend control, as shown in Figure 3-1, to display symbols and descriptions for a specific set of elements of another object, such as the plots or cursors of a graph. When you associate the legend control with another object, any changes you make to that object are automatically reflected in the legend. For example, if you associate the legend control with the plots of a graph, any changes you make in the plots collection editor are automatically reflected in the legend.

Tip For more information about using the legend control, refer to the *Using the Measurement Studio Windows Forms Legend .NET Control* section in the *NI Measurement Studio Help*.

Numeric Controls

Use the Measurement Studio numeric controls to display numerical information, on a Windows Forms user interface, with the look of scientific instruments. The numeric controls include a knob, gauge, meter, slide, thermometer, and tank. The following sections describe operations available with the controls and the classes that interface with them.

With all of the numeric controls and the classes that interface with them, you can perform the following operations:

- Configure the scale to be linear or logarithmic and toggle the visibility of the scale.
- Fill the scale and configure the range, color, dimensions, and style of the fill.
- Connect to the Measurement Studio .NET numeric edit control so that if you change the value of one control, it changes the value of the other control.
- Customize the appearance of the control using 3D lab styles or classic 2D styles and change the color and length of ticks and labels.
- Configure the format of value labels to engineering or date/time.
- Display tooltips reflecting the current value of the pointer.
- Interactively change the value of the control by clicking or dragging and moving the pointer with the mouse.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.
- Programmatically move the pointer to the previous or next value.
- Perform hit testing of mouse cursor coordinates.
- Specify the image format of the control as BMP, GIF, JPEG, or PNG.

Use the Measurement Studio knob, gauge, and meter controls, as shown in Figure 3-5, to input and display numeric data on your user interface.

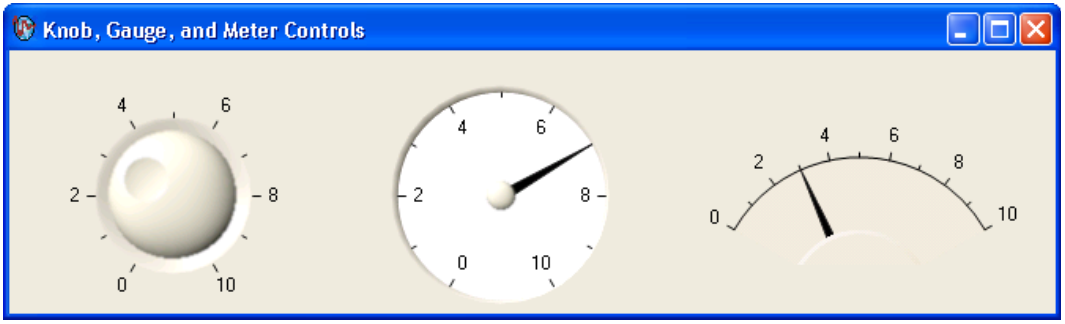


Figure 3-5. Knob, Gauge, and Meter Windows Forms Controls

With the knob, gauge, and meter controls and the classes that interface with the controls, you can perform the following operations:

- Specify the start and sweep angle of the arc programmatically or from the Properties window.
- Use automatic division spacing, custom divisions, and invert the scale.

Use the Measurement Studio slide, tank, and thermometer controls, as shown in Figure 3-6, to input and display numeric data on your interface.

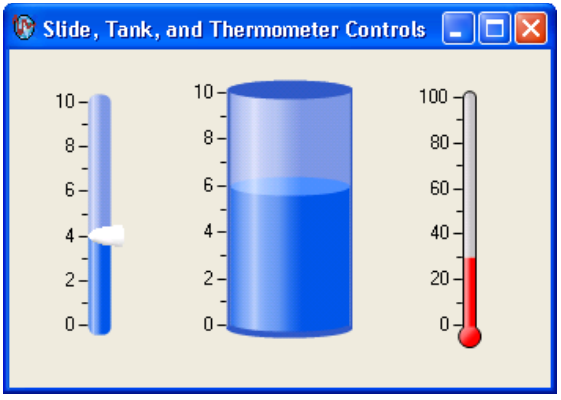


Figure 3-6. .NET Slide, Tank, and Thermometer Controls

With the slide, tank, and thermometer controls and the classes that interface with them, you can perform the following operations:

- Fill to the minimum or maximum value of the scale.
- Position the scale horizontally with left, right, or both and position the scale vertically with top, bottom, or both.

Tip For more information about using the Windows Forms numeric controls, refer to the *Knob*, *Gauge*, *Meter*, *Slide*, *Tank*, or *Thermometer Class* sections in the *NI Measurement Studio Help*.

Numeric Edit Control

Use the Measurement Studio numeric edit control, as shown in Figure 3-7, to display numeric values and to provide a way by which end users can edit numeric values. Typically, you use a numeric edit control to input or display `double` numerical data instead of using a Windows Forms TextBox or NumericUpDown control.

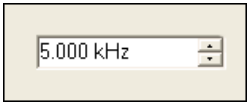


Figure 3-7. Numeric Edit Windows Forms Control

With the numeric edit control and the classes that interface with the control you can perform the following operations:

- Use up and down buttons for easy incrementing and decrementing.
- Perform range checking.
- Set the minimum range value to negative infinity and the maximum range value to positive infinity.
- Create custom formats or use built-in numeric formats including generic, engineering, and simple double. You can use these numeric formats with other Measurement Studio user interface controls, such as the waveform graph and numeric pointer controls.
- Connect to a Measurement Studio numeric control so that if you change the value of one control, it changes the value of the other control.
- Set the coercion mode property to discrete or continuous values. This property configures the control to allow entry or display of either a discrete set of values or any value.
- Set the interaction mode to keyboard and mouse, keyboard only, mouse only, or none.

Tip For more information about using the Windows Forms numeric edit control, refer to the *NumericEdit Class* section in the *NI Measurement Studio Help*.

Switch and LED Controls

Use the Measurement Studio switch and LED controls as Boolean controls on a Windows Forms user interface. You typically use a switch control, as shown in Figure 3-8, to receive and control Boolean input on an application user interface.

Figure 3-8. Switch Windows Forms Control in Vertical Toggle 3D Style

You typically use an LED control, as shown in Figure 3-9, to indicate a Boolean value on an application user interface.

Figure 3-9. LED Windows Forms Control in Square 3D style

With the switch and LED controls and the classes that interface with the controls, you can perform the following operations:

- Receive notification before or after the state of the control changes.
- Configure how the control behaves when you click the control with the mouse or press the spacebar when the control has focus.
- Configure the appearance of the control.
- Make the control background transparent.
- Configure the LED control to blink while it is on or off and configure the rate at which the LED control blinks.

Tip For more information about using the switch and LED controls, refer to the *Using the Measurement Studio Windows Forms Switch and LED .NET Controls* section in the *NI Measurement Studio Help*.

Property Editor Control

Use the Measurement Studio property editor control, as shown in Figure 3-10, to configure properties for Windows Forms controls at run time.

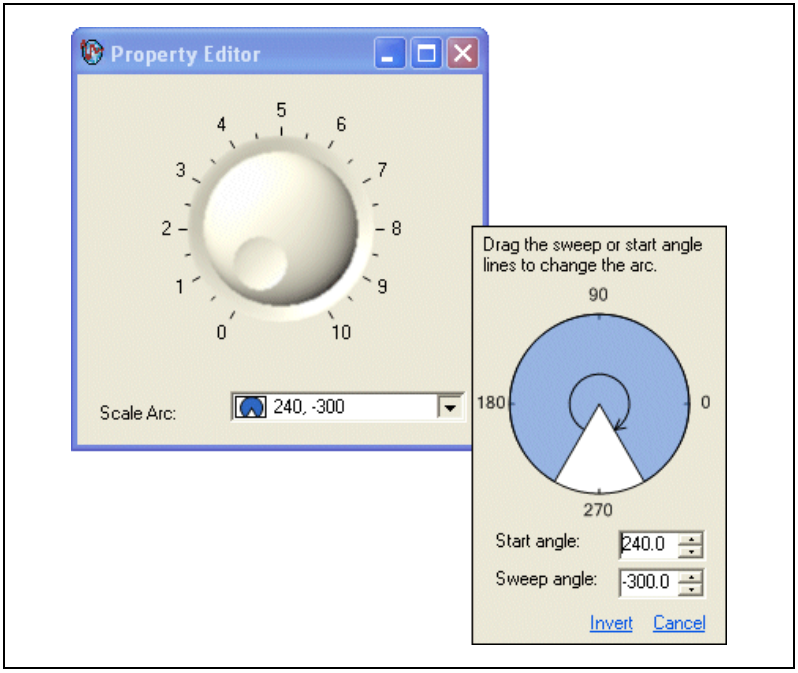


Figure 3-10. Property Editor Windows Forms Control for the Knob Control Scale Arc Property

With the property editor control and the classes that interface with the control, you can perform the following operations:

- Edit any .NET type at run time, including collections.
- Edit expandable properties that represent nested properties of another object, such as major divisions of an axis.
- Display custom editors and type converters for properties.
- Connect to a Windows Forms control so that if you change the value of a property of the control, the Property Editor will update to reflect the change.
- Configure the display mode as a visual representation of the value, text-only, or both.
- Set the interaction mode to edit values or indicator.

Tip For more information about using the property editor control, refer to the *Using the Measurement Studio Property Editor Control* topic in the *NI Measurement Studio Help*.

Windows Forms Array Controls

You can create an array of Measurement Studio controls that behave as a single unit. For example, you can use these array controls to visualize and control ports of a digital line or values of an array. Measurement Studio includes switch, LED, and numeric edit array controls. You can create control arrays of other controls if those controls meet the constraints of the generic type parameter `TControl`.

Switch and LED Array Controls

Use the Measurement Studio switch and LED array controls as an array of Boolean controls on a Windows Forms user interface. You typically use a switch array control, as shown in Figure 3-11 to control ports of a digital line or values of an array. You typically use an LED array control, as shown in Figure, to visualize ports of a digital line or values of an array.

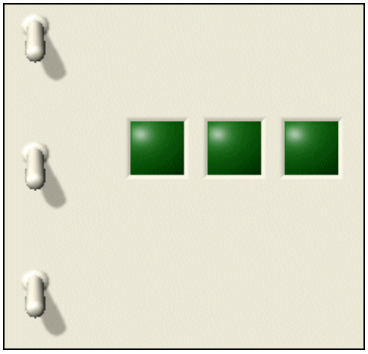


Figure 3-11. Switch and LED Array Controls

With the switch and LED array controls and the classes that interface with the controls, you can perform the following operations:

- Set values by passing an array of data.
- Modify the number of controls displayed based on the length of the specified values.
- Receive notification before or after the state of the control changes.
- Configure how the control behaves when you click it with the mouse or press the spacebar when the control has focus.
- Configure the appearance of the control.
- Make the control background transparent.

- Configure the LED controls to blink while they are on or off and configure the rate at which the LED controls blink.
- Configure the layout of the control to be horizontal or vertical.
- Bind the value of the control to a data source.
- Mark an array of Boolean controls so that only one can be true at a time.

Tip For more information about using the switch and LED array controls, refer to the *Using the Measurement Studio Control Array .NET Controls* topic in the *NI Measurement Studio Help*.

Numeric Edit Array Control

Use the Measurement Studio numeric edit array control, as shown in Figure 3-12 to control and visualize values of an array of `double` values. With the numeric edit array control and the classes that interface with the control you can perform the following operations:

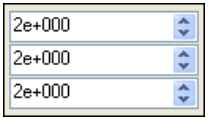


Figure 3-12. Numeric Edit Array Control

- Set values by passing an array of data.
- Modify the number of controls displayed based on the length of the array of values you specify.
- Use up and down buttons for easy incrementing and decrementing.
- Perform range checking.
- Set the minimum range value to negative infinity and the maximum range value to positive infinity.
- Create custom formats or use built-in numeric formats including generic, engineering, and simple double.
- Connect to a numeric control so that if you change the value of one control, it changes the value of the other control.
- Set the coercion mode property to discrete or continuous values. This property configures the control to allow entry or display of either a discrete set of values or any value.
- Set the interaction mode to keyboard and mouse, keyboard only, mouse only, or none.

- Use the edit box to select text programmatically and to validate text values.
- Configure the layout of the control to be horizontal or vertical.
- Bind the value of the control to a data source.

Tip For more information about using the numeric edit array control, refer to the *Using the Measurement Studio Control Array .NET Controls* topic in the *NI Measurement Studio Help*.

InstrumentControlStrip Control

You can use the InstrumentControlStrip control as a toolbar for editing property values of another control through the associated editors at run time. For example, you can populate the InstrumentControlStrip with ToolStripPropertyEditor items that edit property values of a waveform graph through the associated editors at run time. The editor displayed by the ToolStripPropertyEditor is the same editor that displays when you edit the property at design time.

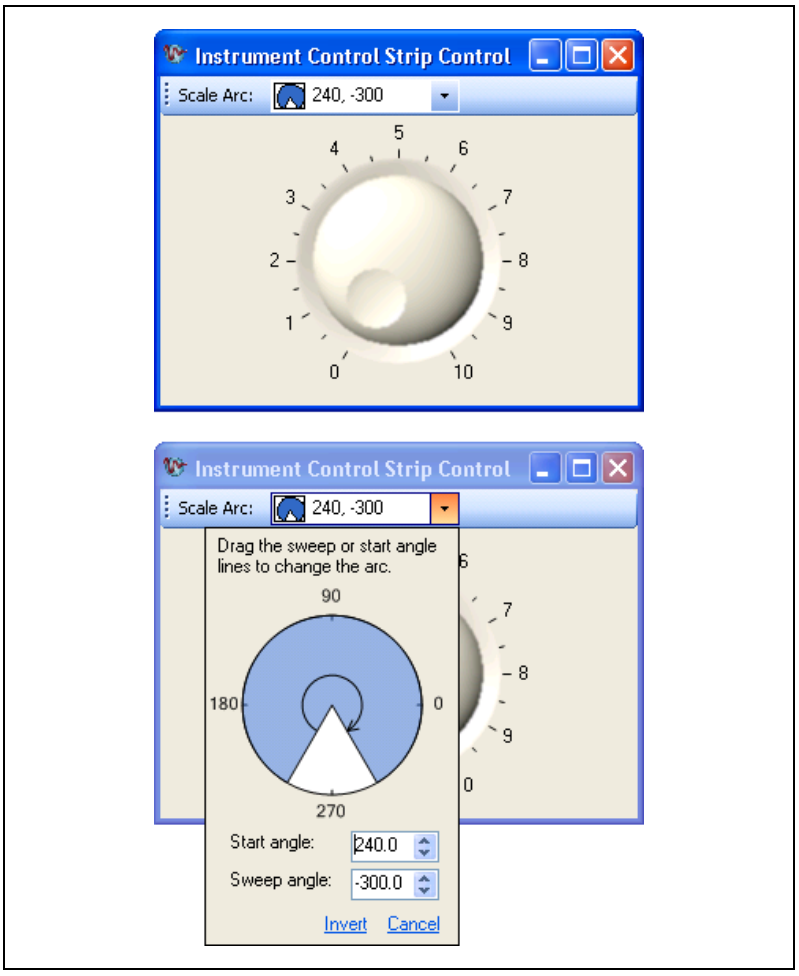


Figure 3-13. InstrumentControlStrip Control

Tip For more information about the InstrumentControlStrip control, refer to *Using the Measurement Studio Windows Forms Instrument Control Strip .NET Control* topic in the *NI Measurement Studio Help*.

ASP.NET Web Forms Controls

The Measurement Studio ASP.NET user interface controls are in the Web Forms .NET class library. The Web Forms .NET class library is in the `NationalInstruments.UI.WebForms` namespace. The Web Forms class library encapsulates the following Measurement Studio user interface controls:

- Waveform graph
- Scatter graph
- Digital waveform graph
- Complex graph
- Legend
- Knob
- Gauge
- Meter
- Slide
- Thermometer
- Tank
- Numeric edit
- Switch
- LED
- AutoRefresh

Note All Measurement Studio ASP.NET Web Forms controls for Visual Studio 2008 are designed to work with ASP.NET AJAX controls. The Measurement Studio ASP.NET Web Forms controls for Visual Studio 2005 are not designed to work with Microsoft ASP.NET AJAX controls.

Use this class library to add measurement-specific user interface controls to your Web application. You can configure the controls programmatically at design time or through the Properties window in the Web Forms Designer. The following sections describe each of the Measurement Studio Web Forms user interface controls.

Waveform Graph and Scatter Graph Controls

Use the Measurement Studio waveform graph and scatter graph controls, as shown in Figure 3-14, to display two-dimensional data on a Web-based user interface. Use the waveform graph to display two-dimensional linear data. You explicitly specify each value in one dimension and provide an initial value and interval to implicitly specify the values in the other dimension. Use the scatter graph to display two-dimensional linear or nonlinear data: you explicitly specify each value in both dimensions.

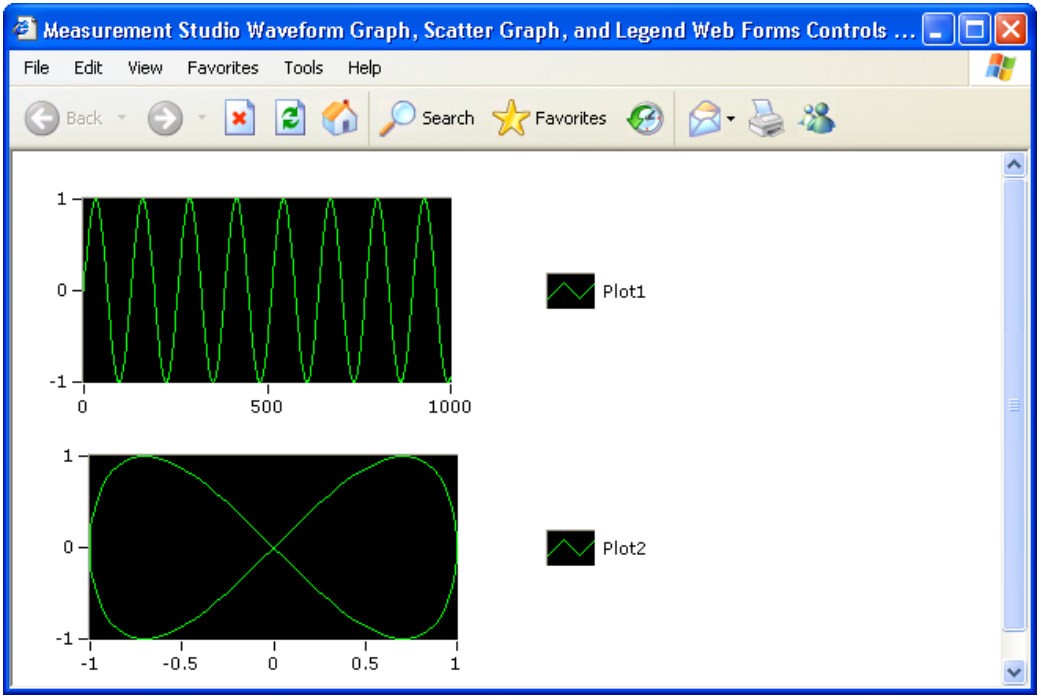


Figure 3-14. Waveform Graph and Scatter Graph Web Forms Controls; Both Graphs Have Corresponding Legends

With the waveform graph and scatter graph controls and the classes that interface with the controls, you can perform the following operations:

Plot Operations

- Plot and chart arrays of double-precision floating point values, analog waveforms, and complex waveforms.
- Configure a graph to contain multiple plots to show separate but related data on the same graph. You can configure a graph to automatically generate different colors for different plots.
- Draw lines or fills from a plot to an X value, Y value, or another plot.
- Specify plots in the scatter graph control as X and Y data. Specify plots in the waveform graph control as X or Y data and optionally with date and time scaling.
- Use the extensible plot and plot area drawing capabilities and events to customize the graph appearance.
- Create custom point and line styles for plots.
- Specify anti-aliased plots for plot lines.
- Calculate and display error bands.
- Configure the plot to specify how data is saved and restored across HTTP requests.

Axis Operations

- Configure a graph to include multiple axes or independent ranges so that plot data fits the graph plot area.
- Configure the axis modes to: fixed; autoscaling, including autoscaling based on the visible data only; strip chart; or scope chart.
- Use logarithmic axes with configurable bases.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.
- Configure major, minor, and custom divisions and origin lines.

Cursor Operations

- Use cursors to identify key points in plots and the plot area.
- Configure cursor snap modes to be floating, nearest point, or to plot.
- Use cursor labels to display X and Y data coordinates in a customized format that the cursor crosshair points to, and customize the text font and colors of the label.

- Create custom point and line styles for cursors.
- Interactively move the cursor by clicking and dragging the vertical or horizontal crosshair or the center of the cursor.
- Programmatically move the cursor to the previous position, to the next position, or to a specified coordinate.
- Create custom mouse cursors programmatically or at design time using the mouse cursor style editor.

Annotation Operations

- Configure text labels, arrows, and drawing shapes to annotate a point anywhere in the plot area of the graph.
- Configure range area, text labels, and arrows to annotate a range in the plot area of the graph.

Additional Operations

- Zoom interactively and programmatically.
- Specify the image format of the control as BMP, GIF, JPEG, or PNG.

Tip For more information about using the waveform and scatter graph controls, refer to the *Using the Measurement Studio Web Forms Scatter and Waveform Graph .NET Controls* section in the *NI Measurement Studio Help*.

Digital Waveform Graph Control

Use the Measurement Studio digital waveform graph control, as shown in Figure 3-15, to display `DigitalWaveform` data in an ASP.NET Web application.

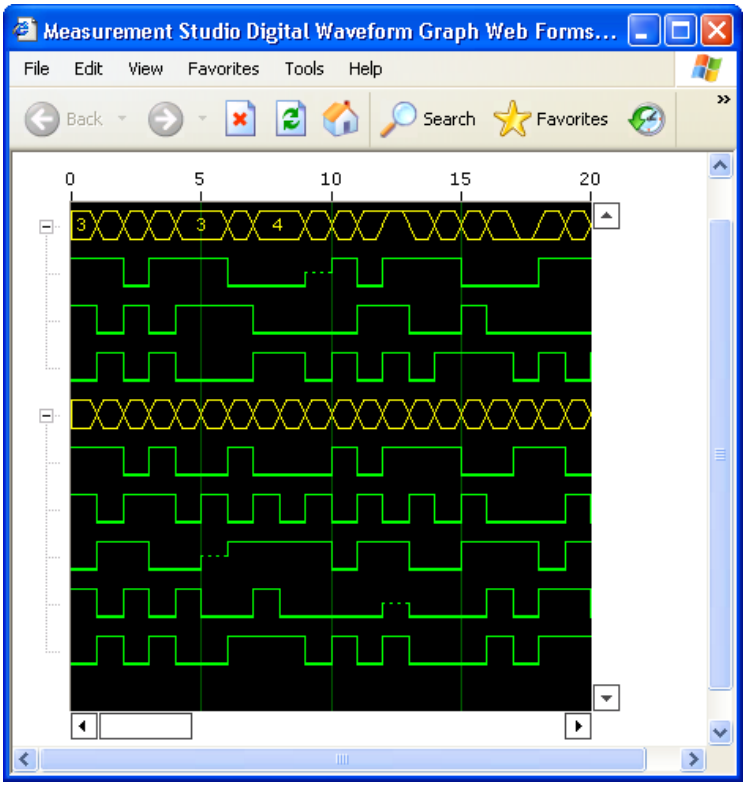


Figure 3-15. Digital Waveform Graph Web Forms Control

With the digital waveform graph control and the classes that interface with the control, you can perform the following operations:

Plot Operations

- Plot digital waveform data, including digital signal state data and timing information.
- Configure plot labels on the y-axis.
- Configure plot templates to customize plots that are implicitly created from plotted data.
- Specify anti-aliased digital plots.
- Expand and collapse signal plots interactively as well as programmatically.

Waveform Sample and Signal State Operations

- Simultaneously display waveforms and signals or display signals only.
- Create custom waveform sample and signal state styles.
- Configure the appearance of sample and state labels.
- Create custom waveform sample and signal state labels.

Axis Operations

- Configure the axis modes to fixed, exact autoscaling, or loose autoscaling.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.
- Display captions on the axis and grid lines.
- Position the axis to display on one or both sides of the graph's plot area.
- Configure major, minor, and custom divisions.

Additional Operations

- Display data in sample or time mode.
- Configure the style and mode of scroll bars.
- Create custom scroll bars.
- Zoom interactively as well as programmatically.
- Specify the image format of the control as BMP, GIF, JPEG, or PNG.
- Create custom mouse cursors programmatically or at design time using the mouse cursor style editor.

Tip For more information about using the digital waveform graph control, refer to the *Using the Measurement Studio Web Forms Digital Waveform Graph .NET Control* section in the *NI Measurement Studio Help*.

Complex Graph Control

Use the Measurement Studio complex graph control, as shown in Figure 3-16, to display `ComplexDouble` data on a ASP.NET Web application. A `ComplexDouble` consists of a real part and an imaginary part. You can use a waveform graph to plot complex waveform data.

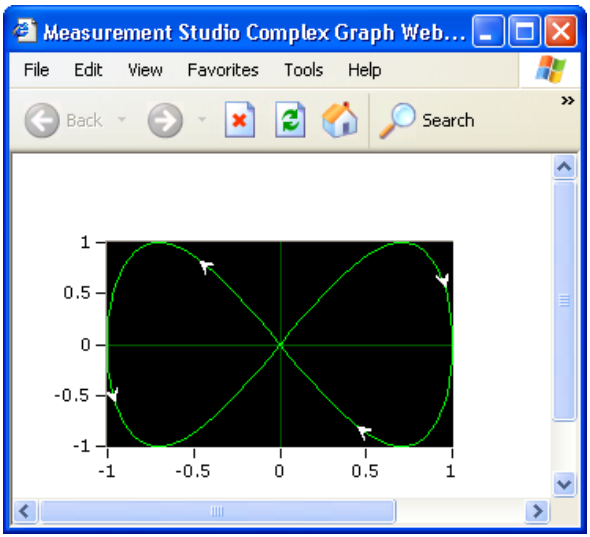


Figure 3-16. Complex Graph Web Forms Control

With the complex graph control and the classes that interface with the control, you can perform the following operations:

Plot Operations

- Plot and chart `ComplexDouble` data.
- Configure a graph to contain multiple plots to show separate but related data on the same graph. You can configure a graph to automatically generate different colors for different plots.
- Draw lines or fills from a plot to an X value, Y value, or another plot.
- Use the extensible plot and plot area drawing capabilities and events to customize the graph appearance.
- Configure the plot to display arrows. The arrows indicate the direction of the complex data.
- Create custom point and line styles for plots.
- Specify anti-aliased plots for plot lines.
- Calculate and display error bands.
- Configure the plot to specify how data is saved and restored across HTTP requests.

Axis Operations

- Configure a graph to include multiple axes or independent ranges so that plot data fits the graph plot area.
- Configure the axis modes to: fixed; autoscaling, including autoscaling based on the visible data only; strip chart; or scope chart.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.
- Display origin lines, captions on the axis, and grid lines.
- Position the axis to display on one or both sides of the plot area of teh graph.
- Configure major, minor, and custom divisions and origin lines.

Cursor Operations

- Use cursors to identify key points in plots and the plot area.
- Configure cursor snap modes to be floating, nearest point, or to plot.
- Use cursor labels to display real, imaginary, magnitude, or phase data that the cursor crosshair points to, and customize the text font and colors of the label.
- Create custom point and line styles for cursors.
- Interactively move the cursor by clicking and dragging the vertical or horizontal crosshair or the center of the cursor.
- Programmatically move the cursor to the previous position, to the next position, or to a specified coordinate.
- Create custom mouse cursors programmatically or at design time using the mouse cursor style editor.

Annotation Operations

- Configure text labels, arrows, and drawing shapes to annotate a point anywhere in the plot area of the graph.
- Configure range area, text labels, and arrows to annotate a range in the plot area of the graph.
- Annotate and label a magnitude value.
- Annotate and label a range of magnitude values for a particular phase.

Additional Operations

- Zoom interactively as well as programmatically.
- Specify the image format of the control as BMP, GIF, JPEG, or PNG.

Tip For more information about using the complex graph control, refer to the *Using the Measurement Studio Web Forms Complex Graph .NET Control* section in the *NI Measurement Studio Help*.

Legend Control

Use the Measurement Studio legend control, as shown in Figure 3-14, to display symbols and descriptions for a specific set of elements of another object, such as the plots or cursors of a graph. When you associate the legend control with another object, any changes you make to that object are automatically reflected in the legend. For example, if you associate the legend control with the plots of a graph, any changes you make in the plots collection editor are automatically reflected in the legend.

Tip For more information about using the legend control, refer to the *Using the Measurement Studio Web Forms Legend .NET Control* section in the *NI Measurement Studio Help*.

Numeric Controls

Use the Measurement Studio numeric controls to display numerical information in an ASP.NET Web application with the look of scientific instruments. The numeric controls include a knob, gauge, meter, slide, thermometer, and tank. The following sections describe operations available with the controls and the classes that interface with them.

With all of the numeric controls and the classes that interface with them, you can perform the following operations:

- Configure the scale to be linear or logarithmic and toggle the visibility of the scale.
- Fill the scale and configure the range, color, dimensions, and style of the fill.
- Connect to a Measurement Studio .NET numeric edit control so that if you change the value of one control, it changes the value of the other control.
- Customize the appearance of the control using 3D lab styles or classic 2D styles and change the color and length of ticks and labels.
- Configure the format of value labels to engineering or date/time.

- Specify the image format of the control as BMP, GIF, JPEG, or PNG.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.
- Display tooltips reflecting the current value of the pointer.
- Interactively change the value of the control by clicking the pointer with the mouse.
- Programmatically move the pointer to the previous or the next value.

Use the Measurement Studio knob, gauge, and meter controls, as shown in Figure 3-17, to input and display numeric data on your user interface.

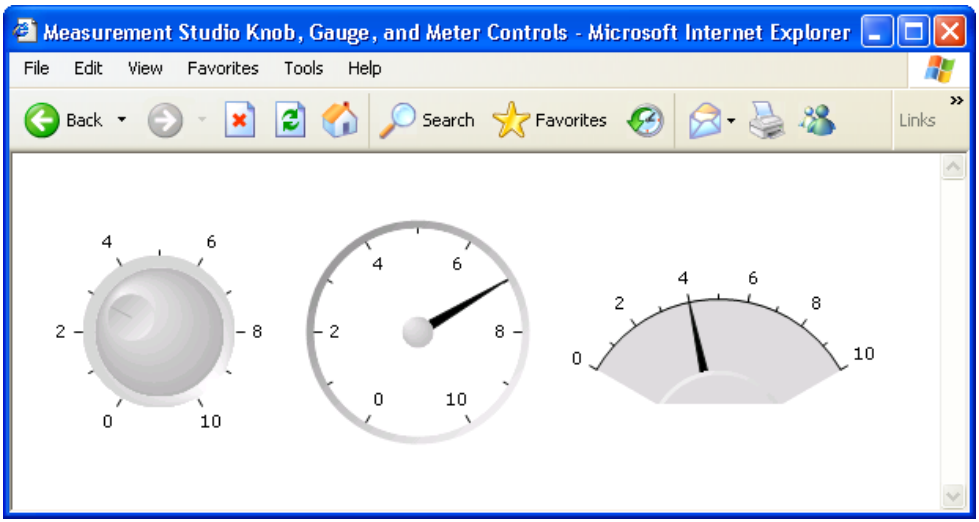


Figure 3-17. Knob, Gauge, and Meter Web Forms Controls

With the knob, gauge, and meter controls and the classes that interface with the controls, you can perform the following operations:

- Specify the start and sweep angle of the arc programmatically or from the Properties window.
- Use automatic division spacing, custom divisions, and invert the scale.

Use the Measurement Studio slide, tank, and thermometer controls, as shown in Figure 3-18, to input and display numeric data on your interface.

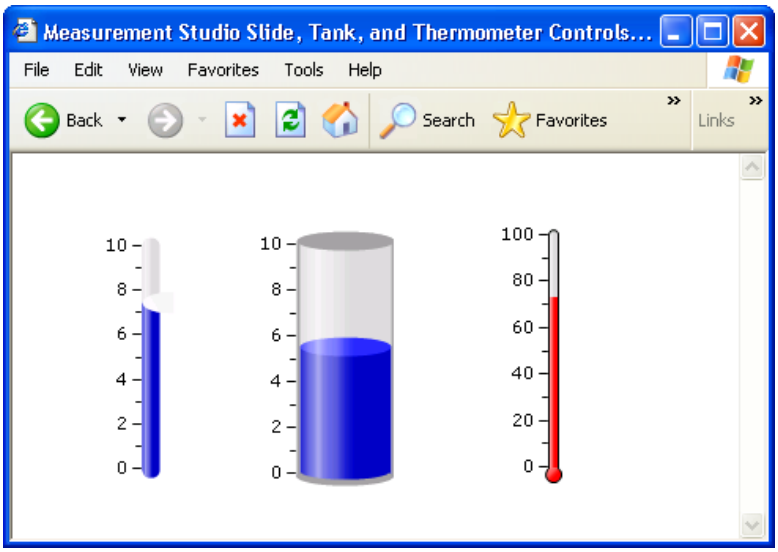


Figure 3-18. Slide, Tank, and Thermometer Web Forms Controls

With the slide, tank, and thermometer controls and the classes that interface with them, you can perform the following operations:

- Fill to the minimum or maximum value of the scale.
- Position the scale horizontally with left, right, or both and position the scale vertically with top, bottom, or both.

Tip For more information about using the Web Forms knob, gauge, meter, slide, tank, or thermometer controls, refer to the *Knob*, *Gauge*, *Meter*, *Slide*, *Tank*, or *Thermometer Class* sections in the *NI Measurement Studio Help*.

Numeric Edit Control

Use the Measurement Studio numeric edit control, as shown in Figure 3-19, to display numeric values and to provide a way by which end users can edit numeric values. Typically, you use a numeric edit control to input or display `double` numerical data instead of using a Web Forms TextBox control.

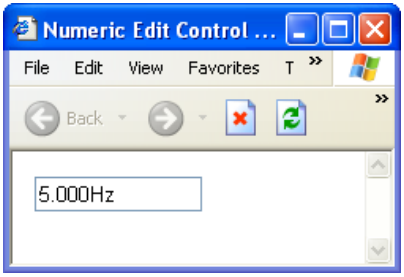


Figure 3-19. Numeric Edit Web Forms Control

With the numeric edit control and the classes that interface with the control you can perform the following operations:

- Perform range checking.
- Set the minimum range value to negative infinity and the maximum range value to positive infinity.
- Create custom formats or use built-in numeric formats including generic, engineering, and simple double. You can use these numeric formats with other Measurement Studio user interface controls, such as the waveform graph and numeric pointer controls.
- Connect to a Measurement Studio numeric control so that if you change the value of one control, it changes the value of the other control.
- Set the coercion mode property to discrete or continuous values. This property configures the control to allow entry or display of either a discrete set of values or any value.
- Validate and format data without posting back to the Web server.

Tip For more information about using the Web Forms numeric edit control, refer to the *NumericEdit Class* section in the *NI Measurement Studio Help*.

Switch and LED Controls

Use the Measurement Studio switch and LED controls as Boolean controls in an ASP.NET Web application. You typically use a switch control to receive and control Boolean input in an ASP.NET Web application. You typically use an LED control to indicate a Boolean value on an ASP.NET Web application. The switch and LED controls are shown in Figure 3-20

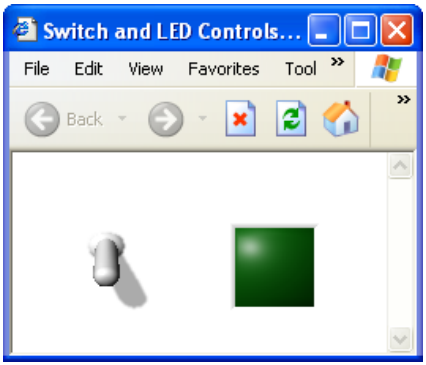


Figure 3-20. Switch Web Forms Control in Vertical Toggle 3D Style and LED Web Forms Control in Square 3D Style

With the switch and LED controls and the classes that interface with the controls, you can perform the following operations:

- Receive notification before or after the state of the control changes.
- Specify the image format of the control as BMP, GIF, JPEG, or PNG.
- Configure the appearance of the control.
- Configure the LED control to blink while it is on or off and configure the rate at which the LED control blinks.

Tip For more information about using the switch and LED controls, refer to the *Using the Measurement Studio Web Forms Switch and LED .NET Controls* section in the *NI Measurement Studio Help*.

AutoRefresh Control

Use the AutoRefresh control to update a Web control or a group of Web controls on the client at a specified interval.

The AutoRefresh control uses the ASP.NET client callback architecture to update a control or a group of controls at a specified interval. The AutoRefresh control sets up a timer inside the browser using Javascript. When the timer elapses, the AutoRefresh updates the controls in the AutoRefresh group. For down-level browsers, the controls update when the page posts back to the server. If the client browser supports client callbacks, the client-side script rendered by the AutoRefresh control uses a client callback to update the associated controls on the client without posting the page back to the server.

Note The AutoRefresh control is designed to work with the ASP.NET AJAX UpdatePanel and Timer controls in Visual Studio 2008.

AutoRefresh Callback

This feature provides a mechanism for updating the `Enabled` and `Interval`properties for `AutoRefresh` from within the `Refresh` callback, allowing you to turn off the `AutoRefresh` or change the `Interval` during an asynchronous HTTP request without causing a postback.

4

Measurement Studio Integrated Tools and Features

When you use Measurement Studio in the Visual Studio environment, you have access to measurement and automation tools and features for Visual Basic .NET and Visual C#. These integrated tools and features are designed to help you quickly and easily build measurement and automation applications.

This chapter includes the following sections to help you develop applications with Measurement Studio:

- *Measurement Studio Menu*
- *Creating a Measurement Studio Project*
- *Adding or Removing Measurement Studio .NET Class Libraries*
- *Creating a Measurement Studio NI-DAQmx Application*
- *Creating an Instrument Control Application*
- *Selecting a Measurement Studio Parameter Value*
- *Using the Instrument Driver Wizard*

Refer to the *Creating Projects with Measurement Studio Core* section in the *NI Measurement Studio Help* for more information about the functionality of these tools and features.

Measurement Studio Menu

The Measurement Studio Menu provides an easy way to access the following National Instruments resources and tools:

- **Measurement Studio Evaluation Home Page**—Launches the Try NI Measurement Studio for Visual Studio site on `ni.com`. This menu item appears only if you do not have a Measurement Studio license already activated.
- **Parameter Assistant**—Use the Measurement Studio Parameter Assistant to discover and insert valid parameter values for various Measurement Studio class libraries, such as NI-DAQmx, NI-488.2,

and NI-VISA methods. The Parameter Assistant is available only if you have Measurement Studio class libraries installed that use parameter values.

- **Add/Remove .NET Class Libraries Wizard**—Use the Measurement Studio Add/Remove Class Libraries wizard to add or remove Measurement Studio class libraries or assemblies in existing Visual Basic .NET or Visual C# projects.
- **Refresh Project License File**—Use the Refresh Project License File to update the `licenses.licx` file in a Measurement Studio project to the currently referenced Measurement Studio assemblies. The Refresh Project process works by going through the `licenses.licx` file line by line for the active project and removing each Measurement Studio licensed type that matches the Measurement Studio `PublicKeyToken`. After all Measurement Studio licensed types are removed from the `licenses.licx` file, the current Measurement Studio licensed types that are referenced by the project are added to the `licenses.licx` file. This ensures all Measurement Studio licensed types used by the project are added to the `licenses.licx` file.
- **Add 64-Bit Protection to Project**—Updates your project's platform target to x86 and updates your project to protect it from build error LC0000. Refer to *Using Measurement Studio on 64-Bit Operating Systems* and *Protecting Your Project from LC0000 Build Error* in the *NI Measurement Studio Help* for more information. This menu item is only available in Visual Studio 2005 on a 64-bit Windows OS. Visual Studio 2008 projects function correctly without this protection.
- **Remove 64-Bit Protection from Project**—Removes protection for build error LC0000 from your project. Refer to *Using Measurement Studio on 64-Bit Operating Systems* and *Protecting Your Project from LC0000 Build Error* in the *NI Measurement Studio Help* for more information. This menu item is only available in Visual Studio 2005 on a 64-bit Windows OS. Visual Studio 2008 projects function correctly without this protection.
- **Update Measurement Studio Project References**—Updates any outdated Measurement Studio references to the latest version installed on the system.
- **NI Tools»Measurement & Automation Explorer (MAX)**—Use MAX to configure NI hardware; add new channels, interfaces, and tasks; execute system diagnostics; and view devices and instruments connected to the system. Select **NI Tools»Measurement & Automation Explorer (MAX)** to access this menu item. The MAX menu option is available only if you have MAX installed.

- **NI Tools»NI Spy**—Use NI Spy to monitor, record, and display National Instruments API calls made by instrument connectivity applications. Use NI Spy to quickly locate and analyze any erroneous National Instruments API calls that an application makes and verify that the communication with an instrument is correct. Select **NI Tools»NI Spy** to access this menu item. The NI Spy menu item is available only if you have NI Spy installed.
- **NI Tools»Distributed System Manager**—Use Distributed System Manager to create new processes and variables, delete existing processes and variables, start and stop processes, create variables with specific data types or the variant data type, allow multiple writers or restrict write access to a single client, and configure server buffering. Select **NI Tools»Distributed System Manager** to access this menu item.
- **Measurement Studio Examples**—Launches the Measurement Studio example directory. You can use these example programs to help you learn and understand key concepts and to explore the functionality of Measurement Studio.
- **NI Measurement Studio Help**—Use the *NI Measurement Studio Help* to access detailed Measurement Studio help, including function reference, walkthroughs, and conceptual topic documentation on developing with Measurement Studio.
- **Measurement Studio Getting Started Guide**—Launches the *Getting Started with Measurement Studio* manual.
- **Measurement Studio Online Resources**—Includes links to the following NI Web sites:
  - **Measurement Studio Home Page**—Launches the Measurement Studio Web site at `ni.com/mstudio`, which contains Measurement Studio news, support, downloads, and evaluation software.
  - **Instrument Driver Network**—Launches the NI Instrument Driver Network Web site at `ni.com/devzone/idnet`, which is the central resource for downloading, developing, and submitting instrument drivers.
  - **Discussion Forums**—Launches the NI Discussion Forums Web site at `forums.ni.com/ni`. With NI Discussion Forums, you can participate in discussion forums and exchange code with measurement and automation developers around the world.
  - **Search Technical Support**—Launches the NI Technical Support Web site at `ni.com/support`. These support resources are available for most products at no cost to registered users and

include software drivers and updates, a KnowledgeBase, product manuals, step-by-step troubleshooting wizards, hardware schematics and conformity documentation, example code, tutorials and application notes, instrument drivers, discussion forums, and a measurement glossary.

  - **NI Developer Zone**—Launches the NI Developer Zone Web site at `zone.ni.com`, which contains an entire library of example programs, tutorial information, and technical presentations, along with a community area for sharing ideas, questions, and source code with other developers.

- **Check for Updates**—Launches the NI Update Service, which checks for updates to National Instruments software you have installed.
- **Patents**—Use the Patents dialog box to view information about NI patents.
- **Licenses**—Use the Licenses dialog box to view information about NI licenses.
- **About Measurement Studio**—Use the Measurement Studio About box to view version information.
- **Activate Measurement Studio**—Launches the NI Activation Wizard, which guides you through activating Measurement Studio. This menu item appears only if you do not have a Measurement Studio license already activated.
- **Purchase Measurement Studio**—Launches the NI Measurement Studio Web page, where you can purchase a Measurement Studio license. This menu item appears only if you have not purchased a Measurement Studio license.

Tip For more information about the resources included in the Measurement Studio Menu, refer to the *Measurement Studio Menu* topic in the *NI Measurement Studio Help*.

Creating a Measurement Studio Project

Measurement Studio includes class library and application templates that you can use to quickly create measurement applications with Visual Basic .NET and Visual C#. Refer to *Walkthrough: Creating a Measurement Studio Application with Windows Forms Controls and Analysis* or *Walkthrough: Creating a Measurement Studio Application with Web Forms Controls and Analysis* in Chapter 2, *Creating Applications with Measurement Studio*, for step-by-step instructions on how to create a Measurement Studio project. Use the Visual Studio New Project dialog

box, as shown in Figure 4-1, to access these templates and to create projects. You can create the following projects in Measurement Studio:

- Measurement Studio Visual Basic .NET project
- Measurement Studio Visual C# project
- Measurement Studio ASP.NET Web site

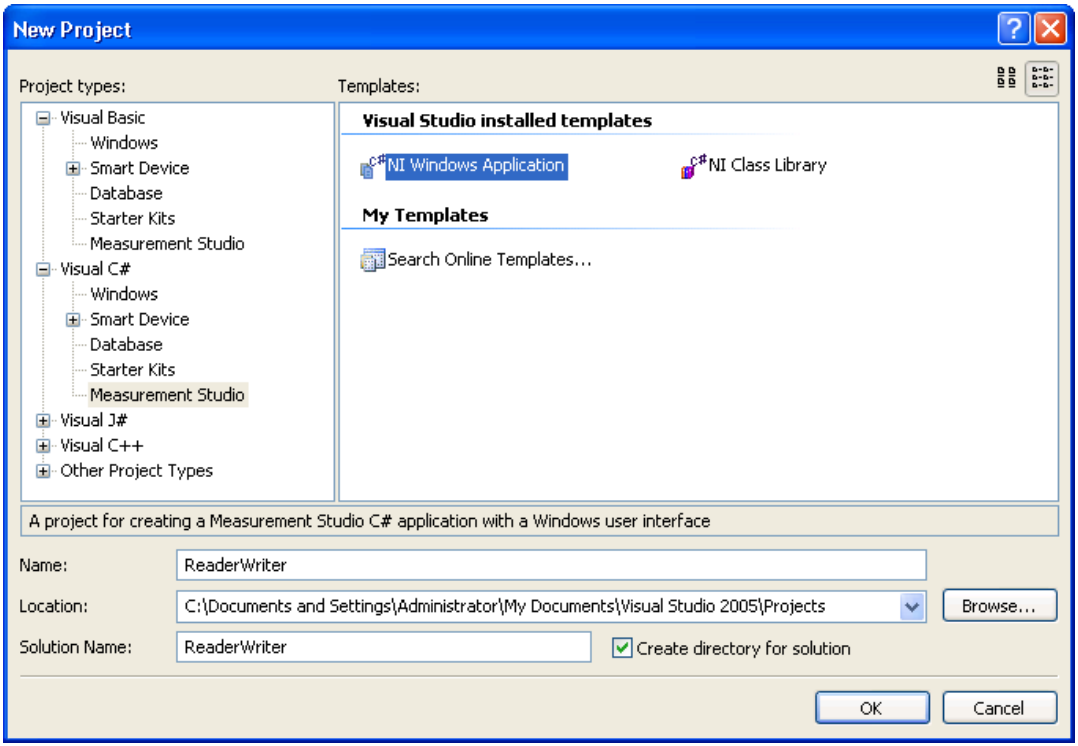


Figure 4-1. New Project Dialog Box in Visual Studio 2005

Tip For more information about using project templates to create a new Measurement Studio project, refer to the *Creating a New Measurement Studio Project* section in the *NI Measurement Studio Help*.

Note For information about converting Measurement Studio projects, refer to the *Converting Measurement Studio Projects* section in the *NI Measurement Studio Help*.

Adding or Removing Measurement Studio .NET Class Libraries

To add or remove Measurement Studio .NET class libraries from a project, use the Measurement Studio Add/Remove .NET Class Libraries wizard on the Measurement Studio menu. This wizard provides an interface, as shown in Figure 4-2, that you can use to select the Measurement Studio .NET class libraries you want to add to or remove from a project.

When you exit the wizard, the wizard adds or removes the appropriate references to or from the project, thus adding or removing the functionality associated with the class library.

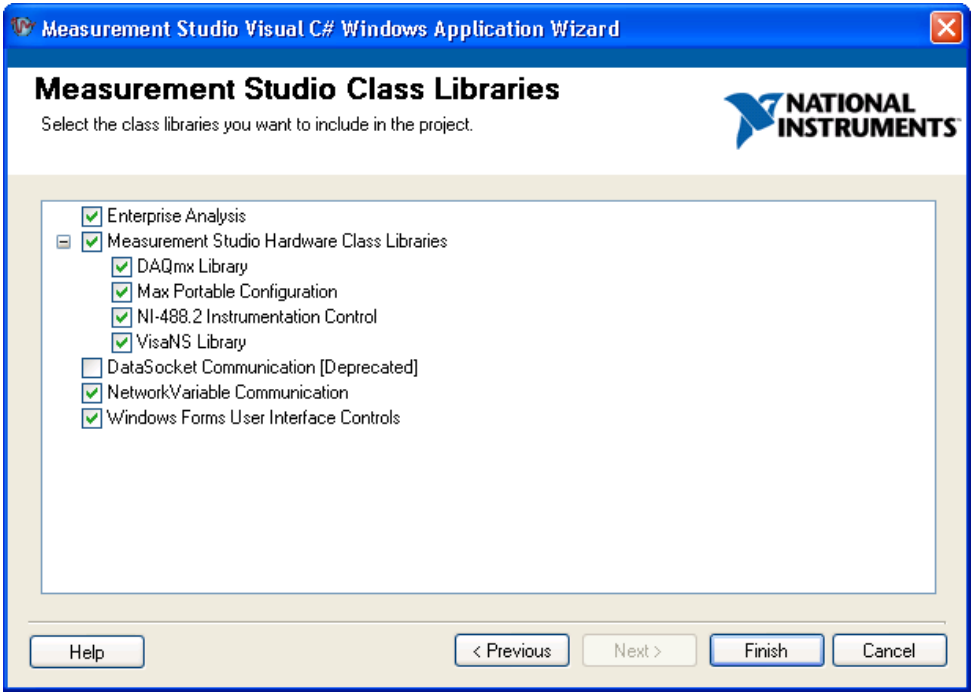


Figure 4-2. Measurement Studio Add/Remove Class Libraries Wizard for Visual Studio 2005

Tip For more information about using the Add/Remove .NET Class Libraries wizard to add or remove Measurement Studio .NET class libraries, refer to the *Adding or Removing Measurement Studio .NET Class Libraries* section in the *NI Measurement Studio Help*.

Creating a Measurement Studio NI-DAQmx Application

To create a Measurement Studio NI-DAQmx application, use the NI DAQ Windows Application template, which launches the DAQ Assistant. The DAQ Assistant integrates into Visual Studio as a code designer. Use the DAQ Assistant user interface, as shown in Figure 4-3, to interactively create and configure the NI-DAQmx task. The DAQ Assistant automatically generates a Visual Basic .NET or Visual C# class that includes the functionality you configure in the user interface.

Note The DAQ Assistant is available only if you have installed NI-DAQmx and either the Measurement Studio Professional or Measurement Studio Enterprise package.

Refer to Chapter 2, *Creating Applications with Measurement Studio*, for step-by-step instructions on how to create a DAQ application.

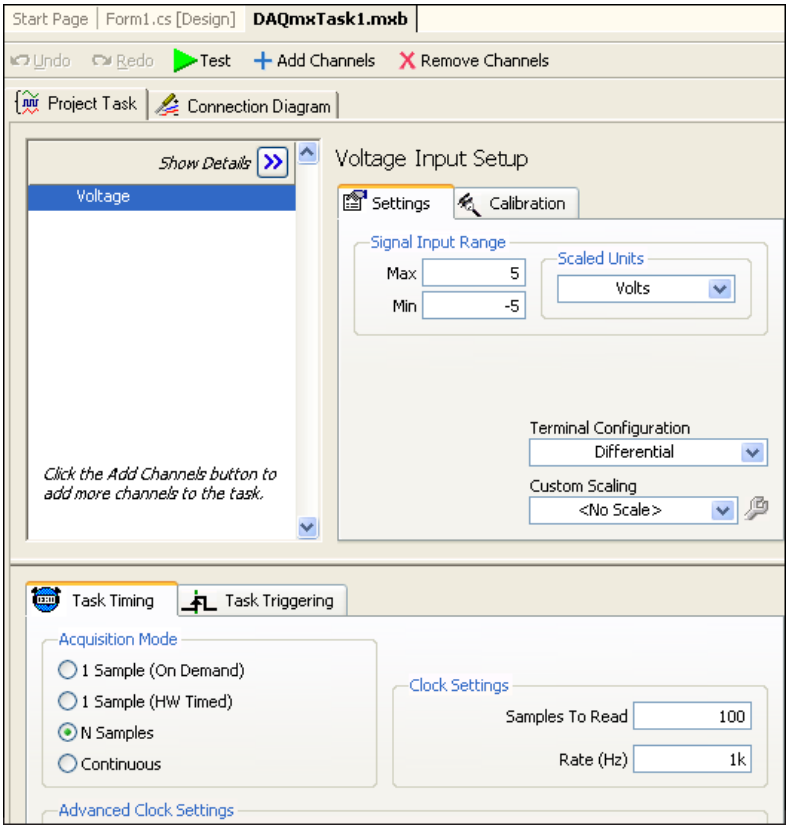


Figure 4-3. DAQ Assistant

The DAQ Assistant interactively assists you in performing the following operations:

- Creating an NI-DAQmx task class
- Configuring an NI-DAQmx task class
- Generating a Visual Basic .NET or Visual C# class that includes the functionality you configure in the user interface
- Generating code that uses an NI-DAQmx task class
- Using an NI-DAQmx task class in a project
- Generating a DAQ component that uses the task to provide appropriate operations for your measurement type.

Tip For more information about using the DAQ Assistant to create a Measurement Studio NI-DAQmx application, refer to the *Creating a Measurement Studio NI-DAQmx Application* section in the *NI Measurement Studio Help*.

Creating an NI-DAQmx User Interface

Using the Configure DAQ Component UI wizard, as shown in Figure 4-4, you can customize and preview a user interface and code for your task. The wizard also generates event handlers and code to acquire data and present it on your generated user interface.

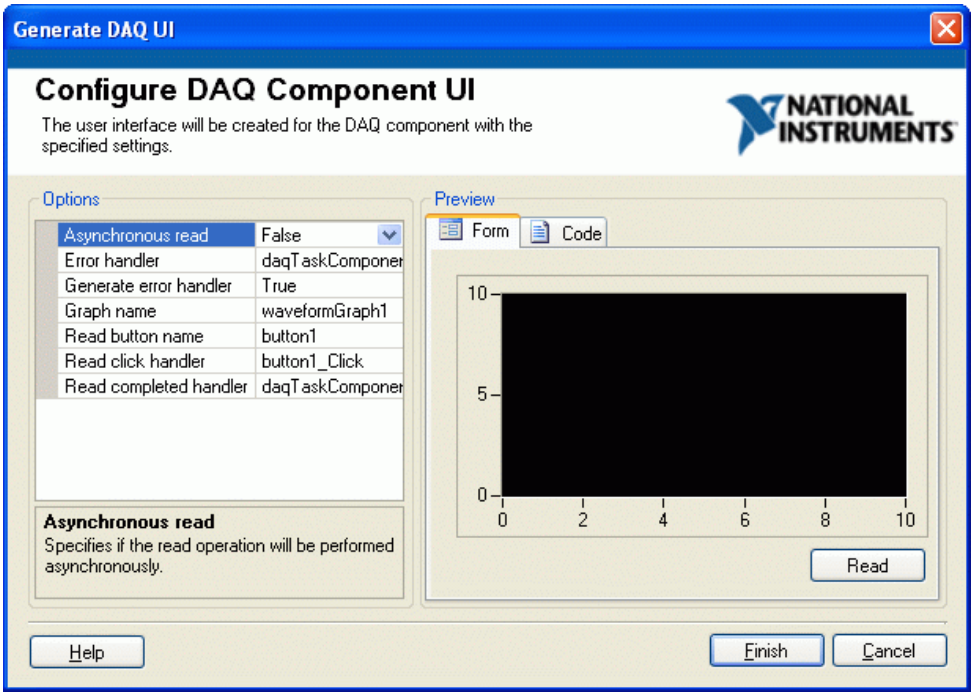


Figure 4-4. Configure DAQ Component UI Wizard

Tip For more information on how to create an NI-DAQmx user interface, refer to the *Using a .NET DAQ Component in a Project* topic in the *NI Measurement Studio Help*.

Creating an Instrument Control Application

To create a Measurement Studio instrument control application, use the NI VISA Windows Application template, which launches the Instrument I/O Assistant. The Instrument I/O Assistant, as shown in Figure 4-5, integrates into Visual Studio as a code designer. Use the Instrument I/O Assistant user interface to create and configure the instrumentation task. The Instrument I/O Assistant generates a Visual Basic .NET or Visual C# class that includes the functionality you configure in the user interface. Use this assistant to help you write code that communicates with devices such as serial, Ethernet, or GPIB instruments.

Note The Instrument I/O Assistant is available only if you have installed either the Measurement Studio Professional or Measurement Studio Enterprise package.

Refer to Chapter 2, *Creating Applications with Measurement Studio*, for step-by-step instructions on how to use the Instrument I/O Assistant.

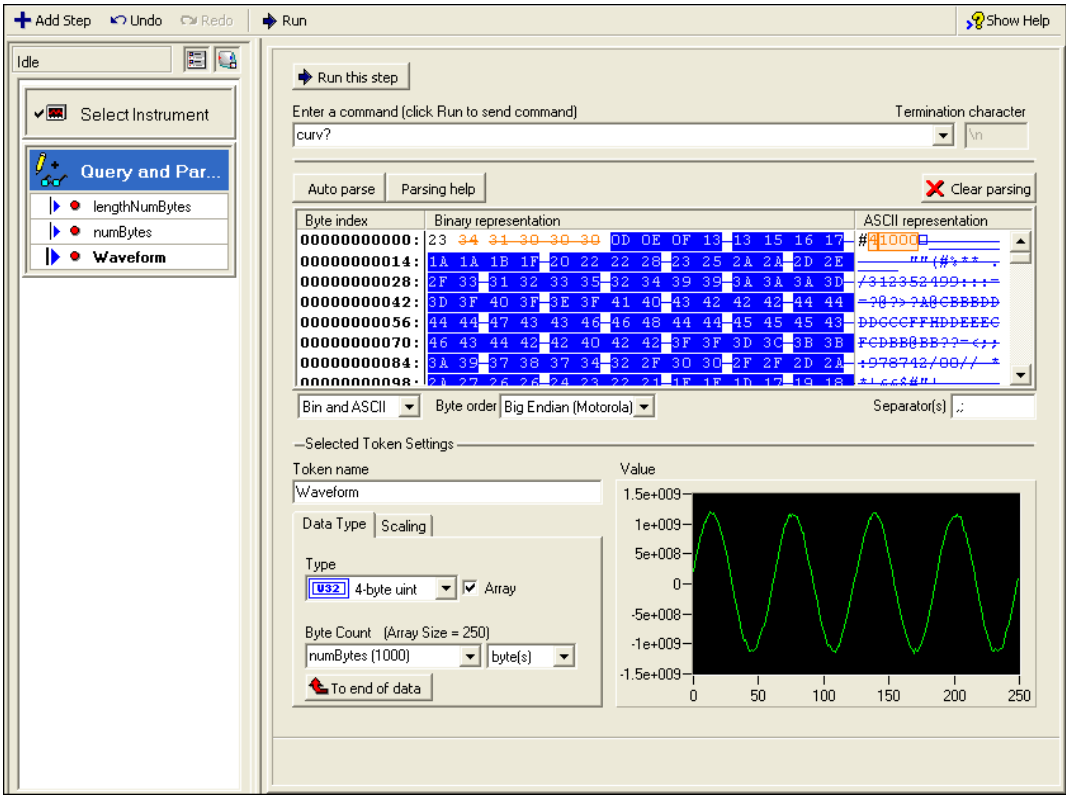


Figure 4-5. Instrument I/O Assistant

The Instrument I/O Assistant aids you in performing the following operations:

- Creating an instrumentation task class.
- Configuring an instrumentation task class to communicate with an instrument and parse data you receive from the instrument.

Tip For more information about using the Instrument I/O Assistant to create a Measurement Studio instrument control application, refer to the *Creating a Measurement Studio Instrument Control Application* section of the *NI Measurement Studio Help*.

Selecting a Measurement Studio Parameter Value

To access I/O devices or resources, you must specify string constants or scalar values for many method parameters and property values. Use the Measurement Studio Parameter Assistant, available from the Measurement Studio menu, to discover and insert into your code valid parameter values for methods and various Measurement Studio class libraries, such as NI-DAQmx, NI-488.2, and NI-VISA.

Note The Parameter Assistant is available only if you have the DataSocket class library installed.

With the Parameter Assistant, you can select the correct parameter value for a device or resource, as shown in Figure 4-6, based on your current system configuration. Click the **Insert Selected Item** button on the Parameter Assistant to insert the value into the current location in the active source file.

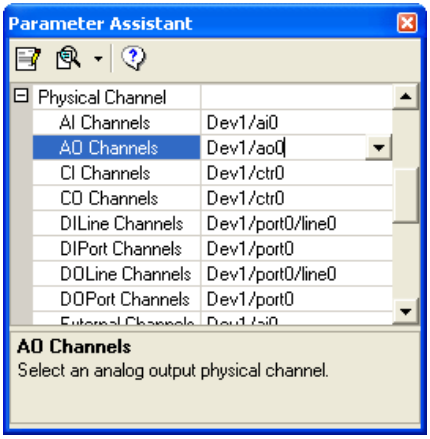


Figure 4-6. Measurement Studio Parameter Assistant

Tip For information about using the Measurement Studio Parameter Assistant to select a parameter value, refer to the *Selecting a Measurement Studio Parameter Value* section in the *NI Measurement Studio Help*.

Using the Instrument Driver Wizard

To use an IVI or VXI *plug&play* instrument driver with a C DLL in a Measurement Studio .NET application, use the Measurement Studio .NET Instrument Driver wizard to create .NET entry points to the C DLL functions you need to call from your application. Use the Add New Item wizard to select the .NET Instrument Driver Wizard.

The Measurement Studio .NET Instrument Driver wizard, as shown in Figure 4-7, generates a .NET wrapper class for calling into IVI, VXI *plug&play*, and legacy instrument drivers based on the instrument driver function panel, header file, and an optional `.sub` file for IVI drivers. The wizard can generate both Visual C# and Visual Basic .NET source code. After completing the wizard, a new instrument driver wrapper class is added to your project and opened in the source code editor.

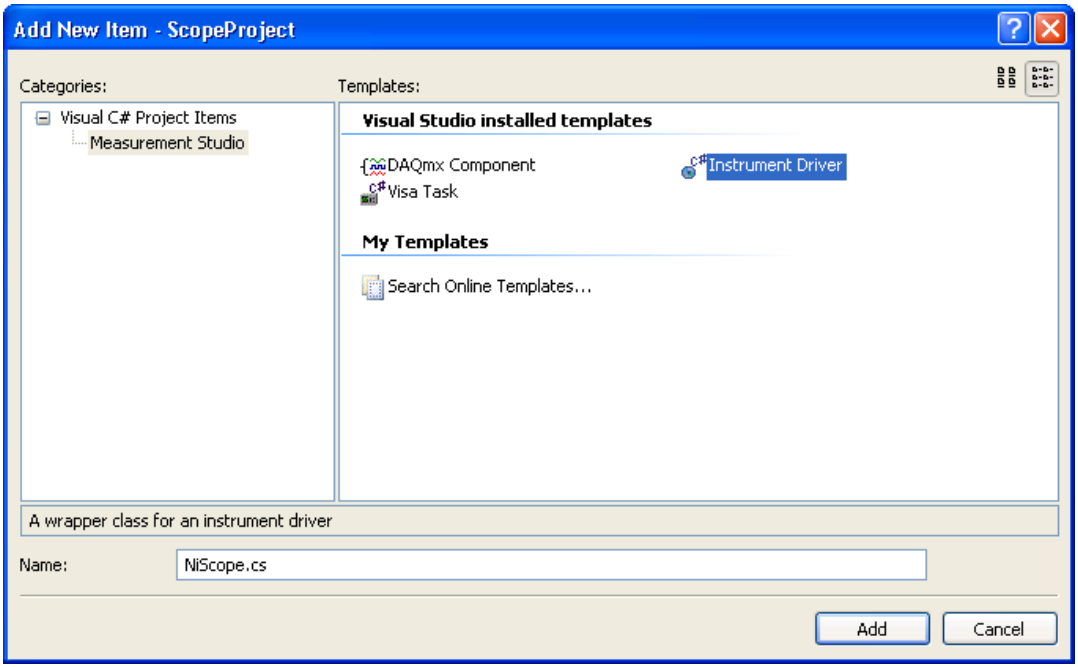


Figure 4-7. Launching the Measurement Studio .NET Instrument Driver Wizard from the Add New Item Wizard

Tip For information about the .NET instrument driver wizard, refer to the *Calling Instrument Drivers from .NET Languages* topic in the *NI Measurement Studio Help*.

A

Technical Support and Professional Services

Visit the following sections of the award-winning National Instruments Web site at ni.com for technical support and professional services:

- **Support**—Technical support at ni.com/support includes the following resources:
  - **Self-Help Technical Resources**—For answers and solutions, visit ni.com/support for software drivers and updates, a searchable KnowledgeBase, product manuals, step-by-step troubleshooting wizards, thousands of example programs, tutorials, application notes, instrument drivers, and so on. Registered users also receive access to the NI Discussion Forums at ni.com/forums. NI Applications Engineers make sure every question submitted online receives an answer.
  - **Standard Service Program Membership**—This program entitles members to direct access to NI Applications Engineers via phone and email for one-to-one technical support as well as exclusive access to on demand training modules via the Services Resource Center. NI offers complementary membership for a full year after purchase, after which you may renew to continue your benefits.

    For information about other technical support options in your area, visit ni.com/services, or contact your local office at ni.com/contact.
- **Training and Certification**—Visit ni.com/training for self-paced training, eLearning virtual classrooms, interactive CDs, and Certification program information. You also can register for instructor-led, hands-on courses at locations around the world.
- **System Integration**—If you have time constraints, limited in-house technical resources, or other project challenges, National Instruments Alliance Partner members can help. To learn more, call your local NI office or visit ni.com/alliance.

If you searched ni.com and could not find the answers you need, contact your local office or NI corporate headquarters. Phone numbers for our worldwide offices are listed at the front of this manual. You also can visit the Worldwide Offices section of ni.com/niglobal to access the branch office Web sites, which provide up-to-date contact information, support phone numbers, email addresses, and current events.

Index

H

I

K

L

M

N

O

P

R

S

T

U

W