Measurement Studio™

Evaluation Guide

April 2008
350836E-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 3390150, 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

Patents

For patents covering National Instruments products, refer to the appropriate location: **Help»Patents** in your software, the `patents.txt` file on your media, or `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

About This Manual

Chapter 1
Measurement Studio Overview and Installation Instructions

Chapter 2
Measurement Studio .NET Class Libraries

About This Manual

The *Measurement Studio Evaluation Guide* introduces the concepts associated with the Measurement Studio class libraries and development tools. This guide assumes that you have a general working knowledge of Microsoft Visual Studio, including .NET Windows, ASP.NET, and MFC.

How To Use this Manual

The Measurement Studio Evaluation package contains the Measurement Studio evaluation software, which correlates to the Measurement Studio Enterprise Edition. For a comparison of features and functionality in Measurement Studio editions, refer to the *Package Comparison Chart* in the *Measurement Studio User Manual.* Any applications you build with the Measurement Studio Evaluation package have a 30-day evaluation period.

Measurement Studio includes support for Visual Studio 6.0, Visual Studio .NET 2003, Visual Studio 2005. and Visual Studio 2008. This manual documents Measurement Studio for Visual Studio 2005/2008. The Measurement Studio support for Visual Studio .NET 2003, Visual Studio 2005, and Visual Studio 2008 CD includes separate, parallel sets of class libraries, integration features, and support documentation for developing with Visual Studio .NET 2003, Visual Studio 2005, and Visual Studio 2008. For help with Visual Studio 6.0, refer to the *Measurement Studio Support for Visual Studio 6.0 Readme* located on the Measurement Studio for Visual Studio 6.0 CD.

The *Measurement Studio Evaluation Guide* is organized into three chapters. Chapter 1, *Measurement Studio Overview and Installation Instructions*, is an overview of National Instruments, virtual instrumentation, and Measurement Studio. This chapter includes installation requirements and installation and evaluation instructions. Chapter 2, *Measurement Studio .NET Class Libraries*, includes information about Measurement Studio features and functionality. Visual C++ .NET support documentation is not included in this Evaluation Guide. For information about Visual C++ features and functionality, refer to the Measurement Studio installed help. Chapter 3, *Getting Started with Measurement Studio*, includes walkthroughs that guide you through step-by-step instructions on how to develop with Measurement Studio features.

Use this guide 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`>.

[ ] — Square brackets enclose optional items—for example, [`response`].

» — The » symbol leads you through nested menu items and dialog box options to a final action. The sequence **File»Page Setup»Options** directs you to pull down the **File** menu, select the **Page Setup** item, and select **Options** 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 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 should enter from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, programs, device names, functions, operations, variables, filenames, and extensions.

1

Measurement Studio Overview and Installation Instructions

The following sections contain information and instructions for installing the Measurement Studio Evaluation Package, which correlates to the Measurement Studio Enterprise Edition. For a comparison of features and functionality in Measurement Studio editions, refer to the *Package Comparison Chart* section in the *Measurement Studio User Manual.* Visual C++ .NET support documentation is not included in this Evaluation Guide. For information about Visual C++ features and functionality, refer to the Measurement Studio installed help.

Note Any applications you build with the Measurement Studio Evaluation package have a 30-day evaluation period.

Learning More About NI and Measurement Studio

NI Platform

National Instruments is committed to providing software and hardware for engineers and scientists who develop measurement and automation applications. NI provides high performance, tight integration, and rapid application development of virtual instruments at a lower cost than traditional measurement instruments.

Virtual Instrumentation

Virtual instruments represent a fundamental shift from traditional hardware-centered instrumentation systems to software-centered systems that exploit the computing power, productivity, display, and connectivity capabilities of popular desktop computers and workstations. With virtual instruments, engineers and scientists build user-defined measurement and automation systems that suit their needs exactly, instead of being limited by traditional vendor-defined instruments.

What Is Measurement Studio?

Measurement Studio is the software tool for creating virtual instruments with Microsoft Visual Studio. Measurement Studio is an integrated suite of tools and class libraries that are designed for developers using Microsoft .NET Windows, ASP.NET, and MFC to develop measurement and automation 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** or browse the evaluation CD to view the release notes.

This evaluation package includes Measurement Studio tools only for Visual C#, Visual Basic .NET, and Visual C++ .NET. Support for Visual C++ .NET is included in Visual Studio 2005 support only. To evaluate Visual Basic 6.0 and Visual C++ 6.0 support, you can purchase the Measurement Studio Professional Edition or Measurement Studio Enterprise Edition. Refer to `ni.com/mstudio` for details.

Tip As you work through this manual, you will see italicized references to relevant Measurement Studio help topics. To find these topics, use the table of contents in the *NI Measurement Studio Help*. 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**. The help launches in a stand-alone help viewer.
- From Visual Studio, select **Help»Contents** to view the Visual Studio table of contents. The *NI Measurement Studio Help* is listed in the table of contents.
- From Visual Studio, select **Measurement Studio»NI Measurement Studio Help**. The help launches within the application.

Why Should I Use Measurement Studio?

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

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 and Visual C++ classes. You can use Measurement Studio to develop a complete measurement and automation application that includes data acquisition, analysis, and presentation functionalities.

Installation Requirements

To use Measurement Studio, your computer must have the following:

- Microsoft Windows Vista/XP/2000 for Visual Studio 2005 or Microsoft Windows Vista/XP for Visual Studio 2008

Note If you have Windows Vista installed you must also have both Visual Studio 2005 Service Pack 1 and Visual Studio Service Pack 1 Update for Windows Vista installed on your machine for Measurement Studio to function properly.

- Microsoft .NET Framework 2.0 for Visual Studio 2005 or Microsoft .NET Framework 3.5 for Visual Studio 2008 (required only for the Measurement Studio .NET class libraries)
- Standard, Professional, or Team System edition of Microsoft Visual Studio 2005 or Standard, Professional, or Team System edition of Microsoft Visual Studio 2008 (required to use the Measurement Studio integrated tools) or Visual C#, Visual Basic .NET, or Visual C++ Express editions of Microsoft Visual Studio 2005 or Microsoft Visual Studio 2008
- Intel Pentium III class processor, 1 GHz or higher
- Video display—1024 × 768, 256 colors (16-bit color recommended for user interface controls)
- Minimum of 256 MB of RAM (512 MB or higher recommended)
- Minimum of 385 MB of free hard disk space for Visual Studio 2005 support or minimum of 200 MB of free hard disk space for Visual Studio 2008 support
- Microsoft-compatible mouse
- Microsoft Internet Explorer 6.0 or later

Optional Installation—In order for links from Measurement Studio help topics to .NET Framework help topics to work, you must install the Microsoft .NET Framework SDK 2.0 or Microsoft .NET Framework SDK 3.5.

Driver Support

To use .NET class libraries that interface to National Instruments device drivers, NI-DAQmx, NI-VISA and NI-488.2, and the MAX (Measurement & Automation Explorer) configuration utility, you must install the underlying device drivers in addition to the .NET class libraries. You can run the underlying device driver installers from the NI Device Drivers CD. 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 CD.

Installing Measurement Studio

Complete the following steps to install Measurement Studio. These steps describe a typical installation. Please carefully review all additional licensing and warning dialog boxes.

National Instruments recommends that you exit all programs before running the Measurement Studio installer. Applications that run in the background, such as virus scanning utilities, might cause the installer to take longer than average to complete.

Note There are separate installers for Measurement Studio support for Visual Studio 2005 and Measurement Studio support for Visual Studio 2008. Repeat the installation instructions to install support for both. When installing support for more than one version of Visual Studio, you can reduce installation time by running the Device Drivers CD installer only once. To do this, ensure that the Device Drivers CD feature is enabled only for the **last** Measurement Studio Visual Studio support installer that you run.

The option to browse for an installation location is valid only if you have not already installed any Measurement Studio features for the version of Visual Studio or the .NET Framework that you are installing. If you have any Measurement Studio features installed, then Measurement Studio installs to the same root directory to which you installed other Measurement Studio features.

Complete the following steps to install Measurement Studio:

1. Log on as an administrator or as a user with administrator privileges.
2. Launch `Autorun.exe`, either from the installation CD or from the location to which you extracted the downloaded CD image.
3. Select the version of Visual Studio you want to install support for.
4. Follow the instructions that appear on the screen.

Note If you want to upgrade a Windows XP machine to Windows Vista, National Instruments recommends first uninstalling all NI software, including both application software and drivers.

Installing Hardware Drivers for Visual Studio 2008 Support

Visual Studio 2008 .NET class library support for National Instruments hardware drivers is included on the Measurement Studio 8.5 CD, under the VS2008 Driver Support feature in the feature tree. To install support for NI-DAQmx, NI-VISA, NI-488.2, or MAX, you must install the appropriate feature from the Measurement Studio 8.5 CD and you must install the underlying device driver from the NI Device Drivers CD or from a product-specific driver installer. Refer to the *Driver Support* section for information on obtaining device driver installers.

To install support for NI-DAQmx:

1. In the NI Measurement Studio 8.5 installer, enable the **VS2008 Driver Support».NET Framework 3.5 Languages Support for NI-DAQmx** feature.
2. In the NI Device Drivers installer, enable the **Data Acquisition» NI-DAQmx** feature.

To install support for NI-VISA:

1. In the NI Measurement Studio 8.5 installer, enable the **VS2008 Driver Support».NET Framework 3.5 Languages Support for NI-VISA** feature. If you want to use the Instrument I/O Assistant inside Visual Studio 2008, enable the **VS2008 Driver Support»VS2008 DotNET IIOAssistant Support** feature.
2. In the NI Device Drivers installer, enable the **Instrument Control» NI-VISA** feature.

To install support for NI-488.2:

1. In the NI Measurement Studio 8.5 installer, enable the **VS2008 Driver Support».NET Framework 3.5 Languages Support for NI-488.2** feature.
2. In the NI Device Drivers installer, enable the **Instrument Control» NI-488.2** feature.

To install support for MAX:

1. In the NI Measurement Studio 8.5 installer, enable the **VS2008 Driver Support».NET Framework 3.5 Languages Support for NI MAX** feature.
2. In the NI Device Drivers installer, enable the **NI Measurement and Automation Explorer** feature.

Installing Hardware Drivers for Visual Studio 2005 Support

The .NET and C++ class libraries for Visual Studio 2005 support for National Instruments hardware drivers are included in the Driver CD installer.

Installing the Current Version of Measurement Studio over Previous Versions of Measurement Studio

Note You can have only one version of Measurement Studio installed on a system for each version of Visual Studio or the .NET Framework installed on the system. For example, you can have Measurement Studio 8.1.2 for Visual Studio 2005 installed on the same system as Measurement Studio 8.5 for Visual Studio 2008, but you cannot have Measurement Studio 8.1.2 for Visual Studio 2005 installed on the same system as Measurement Studio 8.5 for Visual Studio 2005.

If you install a newer version of Measurement Studio on a machine that has a prior version of Measurement Studio installed, the newer version installer replaces the prior version functionality, including class libraries. However, the prior version assemblies remain in the global assembly cache (GAC); therefore, applications that reference the prior version continue to use the prior version .NET assemblies.

Note This does not apply to `NationalInstruments.Common.dll`. `NationalInstruments.Common.dll` uses a publisher policy file to redirect applications to always use the newest version of `NationalInstruments.Common.dll` installed on the system, for each version of the .NET Framework. `NationalInstruments.Common.dll` is backward-compatible.

2

Measurement Studio .NET Class Libraries

This chapter provides overview information about features and functionality included in Measurement Studio support for Visual Studio 2005 and Visual Studio 2008. Refer to the *NI Measurement Studio Help* for detailed information about these features. Refer to Chapter 3, *Getting Started with Measurement Studio*, for step-by-step instructions on developing Measurement Studio applications.

Measurement Studio includes the following features and functionality:

- User interface controls
- Analysis class libraries
- Common class library
- DataSocket class library
- Network Variable class library
- Data acquisition
- Instrument control
- Integration into the Visual Studio environment

User Interface Controls

Measurement Studio includes managed .NET user interface Windows Forms and Web Forms controls designed specifically for test and measurement applications.

The Measurement Studio ASP.NET Web Forms controls are designed to provide a rich user interface experience through the Web browser. The browsers are divided into two broad categories: uplevel and downlevel. Uplevel browsers include recent versions of Microsoft Internet Explorer 5 and later versions and Mozilla Firefox. All other browsers are downlevel browsers.

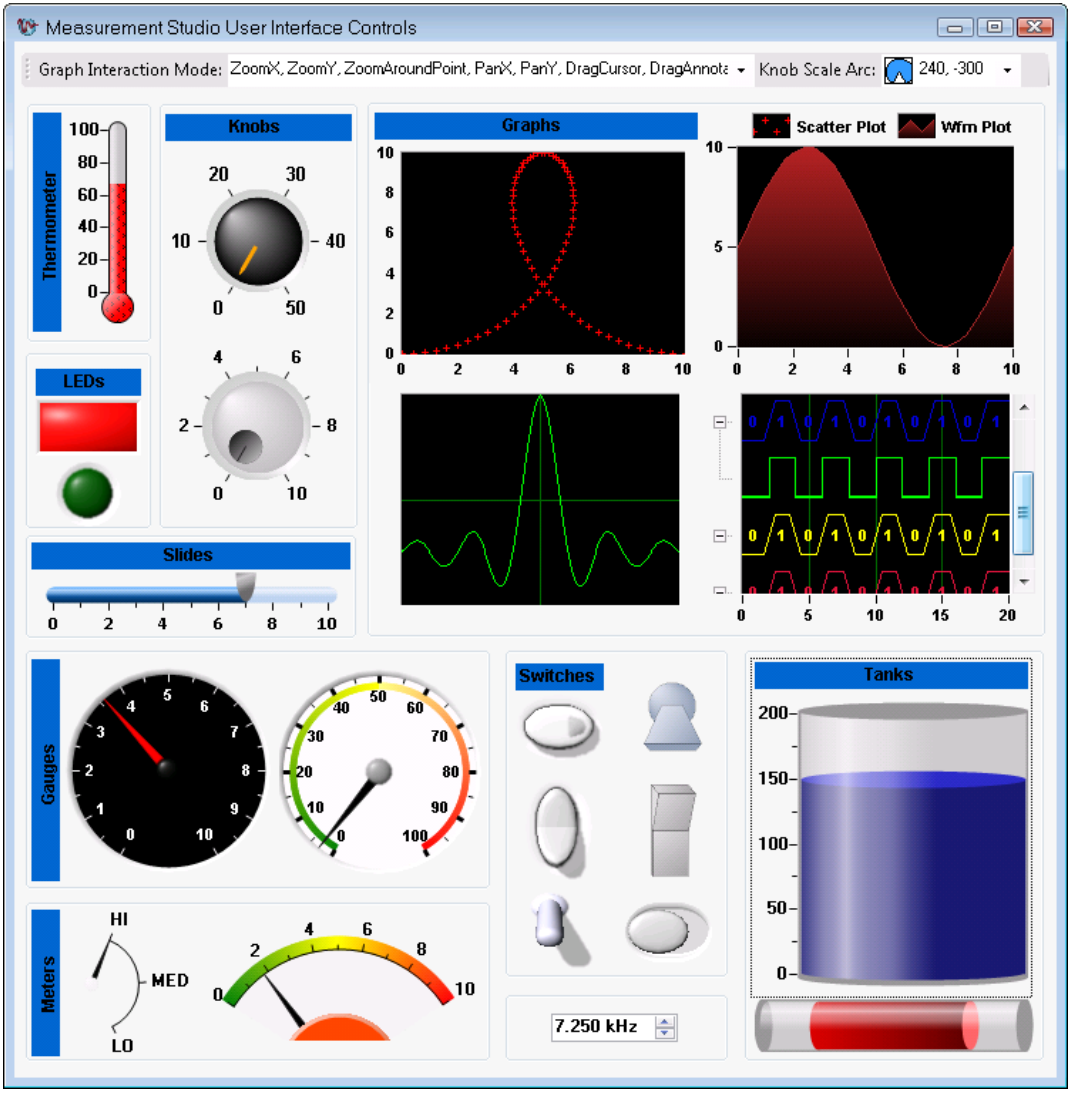
Note All Measurement Studio ASP.NET Web Forms controls for Visual Studio 2008 are designed to work with ASP.NET AJAX controls. The Measurement Studio ASP.NET Web Forms controls for Visual Studio 2005 are not designed to work with Microsoft ASP.NET AJAX controls.

By using Measurement Studio controls, you can focus on creating an end solution, instead of developing UI components.

Table 2-1. Measurement Studio User Interface Controls

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

The following figure shows a sample of the Measurement Studio UI controls.



Graph and Legend Controls

Measurement Studio includes four graphs: the waveform graph, scatter graph, digital waveform graph, and complex graph. Use the graphs to display data in the application type you need.

You can use the Measurement Studio waveform graph and scatter graph controls, as shown in Figure 2-1, to display two-dimensional data on a Windows Forms user interface or in a Web browser. 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. You can use the scatter graph to display two-dimensional linear or nonlinear data by explicitly specifying each value in both dimensions.

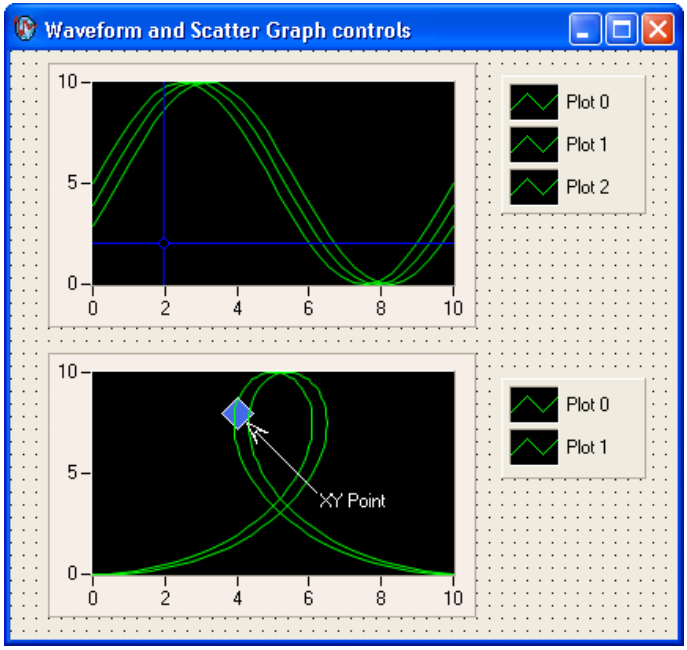


Figure 2-1. Waveform Graph Windows Forms Control with Cursors and Scatter Graph Windows Forms Control with XY Point Annotation; Both Graphs Have Corresponding Legends

You can use the Measurement Studio digital waveform graph control, as shown in Figure 2-2, to display `DigitalWaveform` data on a Windows Forms user interface or in a Web browser.

You can use the Measurement Studio complex graph control to display `ComplexDouble` data on a Windows Forms user interface or in a Web browser. A `ComplexDouble` number consists of a real part and an imaginary part. You can use a waveform graph to plot complex waveform data.

You can use the Measurement Studio legend control, as shown in Figure 2-1, to display symbols and descriptions for a specific set of elements of another object, such as the plots or cursors of a graph.

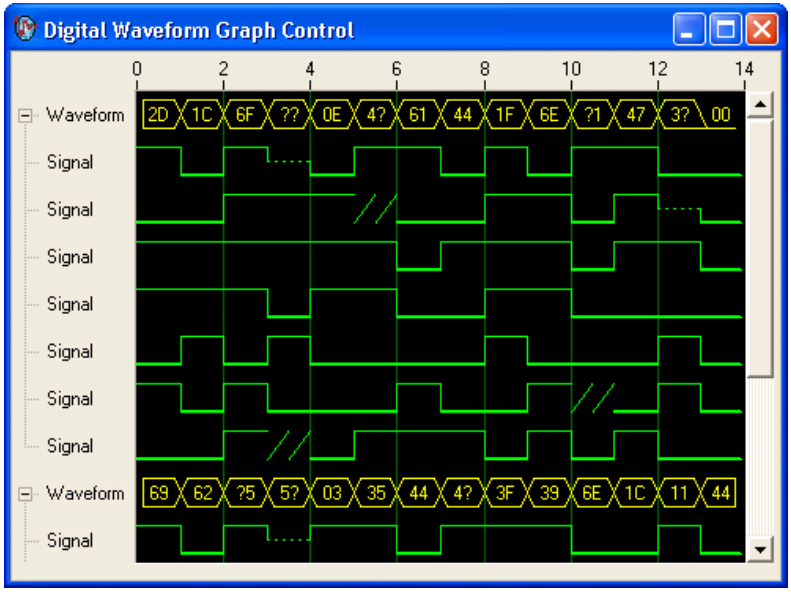


Figure 2-2. .NET Digital Graph

The following sections list the operations you can perform with the Measurement Studio graphs.

Note The following sections include a sample of the functionality available with the graph controls; however, for a complete list of graph control functionality, refer to the *Measurement Studio User Manual* by selecting **Start»All Programs»National Instruments»<Measurement Studio>»Measurement Studio Documentation» User Manual**.

Waveform and Scatter Graph Controls

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

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.
- 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.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.

Cursor Operations

- Use cursors to identify key points in plots and the plot area.

Annotation Operations

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

Additional Operations

- Pan and zoom interactively, as well as programatically.
- Bind a plot to a data source on the waveform graph.

Digital Waveform Graph Control

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

Plot Operations

- Plot digital waveform data. Data values can represent up to eight different digital states.
- Expand and collapse signal plots interactively.

Waveform Sample and Signal State Operations

- Create custom waveform sample and signal state labels.

Axis Operations

- Configure the axis modes to be 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.

Additional Operations

- Display data in sample or time mode.
- Pan and zoom interactively and programmatically.

Complex Graph Control

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

Plot Operations

- Plot and chart `ComplexDouble` data.
- Configure a graph to contain multiple plots to show separate but related data on the same graph.
- Configure the plot to display arrows. The arrows indicate the direction of the complex data.
- 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.
- Interactively change the range of an axis and invert the axis at run time by clicking on the axis end labels.

Cursor Operations

- Use cursors to identify key points in plots and the plot area.

Annotation Operations

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

Additional Operations

- Pan and zoom interactively.

Tip For more information about using the waveform, scatter, digital waveform, and complex graph and legend controls, refer to the *Using the Measurement Studio Graph .NET Controls* and *Using the Measurement Studio Legend .NET Control* sections in the *NI Measurement Studio Help*.

Numeric Controls

Use the Measurement Studio numeric controls to display numerical information, on a Windows Forms or Web 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.

Note The following sections include a sample of the functionality available with the numeric controls; however, for a complete list of numeric control functionality, refer to the *Measurement Studio User Manual* by selecting **Start»All Programs»National Instruments»<Measurement Studio>»Measurement Studio Documentation» User Manual**.

With the numeric controls and the classes that interface with them, you can perform many operations, including:

- 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.

Use the Measurement Studio knob, gauge, and meter controls, as shown in Figure 2-3, to input and display numeric data on your user interface.

Figure 2-3. Knob, Gauge, and Meter .NET Controls

In addition to the functionality included with all numeric controls, for the knob, gauge, and meter controls, you can specify the start and sweep angle of the arc programmatically or from the Properties window.

Use the Measurement Studio slide, tank, and thermometer controls, as shown in Figure 2-4, to input and display numeric data on your interface.

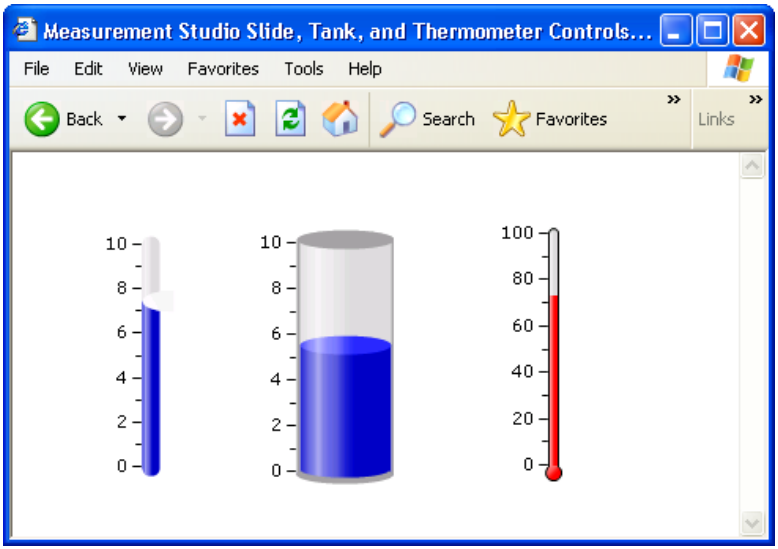


Figure 2-4. Slide, Tank, and Thermometer .NET Controls

In addition to the functionality included with all numeric controls, for the slide, tank, and thermometer controls, you can 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 and 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 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 control, Windows Forms NumericUpDown control, or a Web Forms TextBox control.

Note The following section includes a sample of the functionality available with the numeric edit control; however, for a complete list of numeric edit control functionality, refer to the *Measurement Studio User Manual* by selecting **Start»All Programs»National Instruments»<Measurement Studio>»Measurement Studio Documentation»User Manual**.

With the numeric edit control and the classes that interface with the control you can perform many operations, including:

- 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.

Tip For more information about using the Windows Forms or 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 on a Windows Forms or Web Forms user interface. You typically use a switch control, as shown in Figure 2-5, to receive and control Boolean input on an application user interface.

Figure 2-5. Switch Control in Vertical Toggle 3D Style

You typically use an LED control, as shown in Figure 2-6, to indicate a Boolean value on an application user interface.

Figure 2-6. LED Control in Square 3D Style

Note The following section includes a sample of the functionality available with the Boolean controls; however, for a complete list of Boolean control functionality, refer to the *Measurement Studio User Manual* by selecting **Start»All Programs»National Instruments»<Measurement Studio>»Measurement Studio Documentation» User Manual**.

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

- 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.

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 or the *Using the Measurement Studio Web Forms Switch and LED .NET Controls* section in the *NI Measurement Studio Help*.

Additional Controls

Property Editor Control

Use the Measurement Studio property editor control, as shown in Figure 2-7, to configure properties for Windows Forms controls at run time.

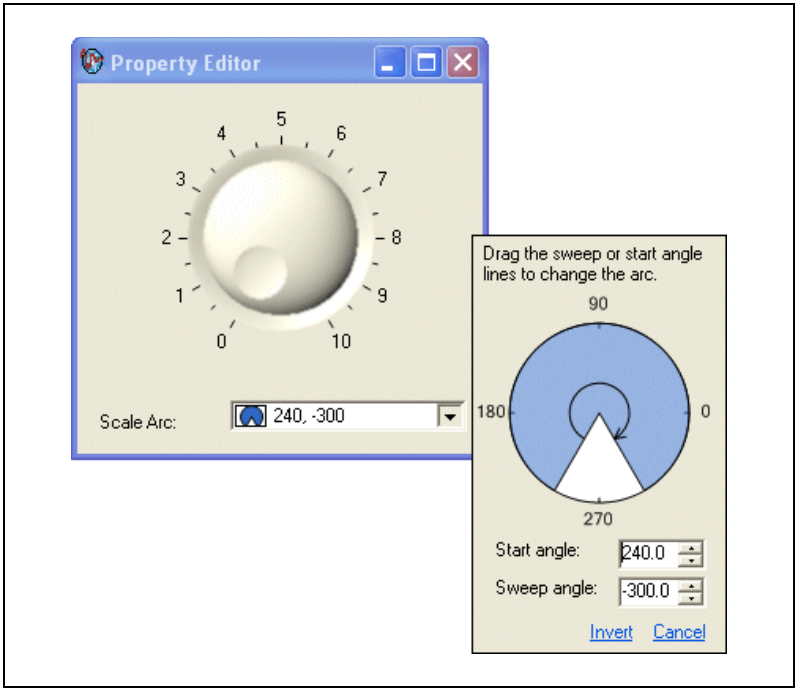


Figure 2-7. Property Editor Control for the Knob Control Scale Arc Property

Note The following section includes a sample of the functionality available with the property editor control; however, for a complete list of property editor control functionality, refer to the *Measurement Studio User Manual* by selecting **Start»All Programs»National Instruments»<Measurement Studio>»Measurement Studio Documentation»User Manual**.

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

- 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.

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*.

Instrument Control Strip 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 2-8 shows the Measurement Studio instrument control strip control.

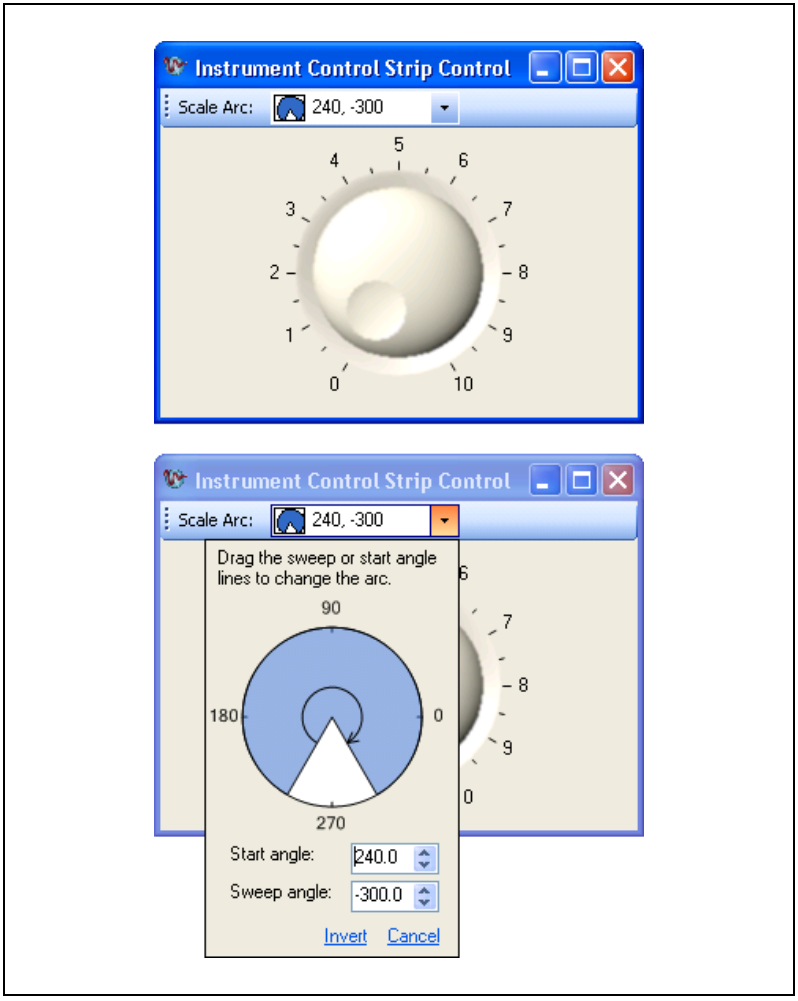


Figure 2-8. Instrument Control Strip Control

With the instrument control strip control and the classes that interface with the control, you can perform the following operations:

- Use the instrument control strip control as a toolbar for editing property values of another control through the associated editors at run time.
- Edit multiple property values of controls with one instrument control strip control.

- Add other types of controls, such as the tool strip button or tool strip label control, to the instrument control strip control.
- Customize the appearance of the control.

Tip For more information, refer to the *Using the Instrument Control Strip 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`.

Note The following sections include a sample of the functionality available with the array controls; however, for a complete list of array control functionality, refer to the *Measurement Studio User Manual* by selecting **Start»All Programs»National Instruments»<Measurement Studio>»Measurement Studio Documentation» User Manual**.

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 2-9, to control ports of a digital line or values of an array. You typically use an LED array control, as shown in Figure 2-9, to visualize ports of a digital line or values of an array.

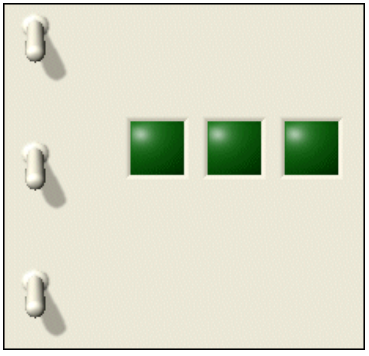


Figure 2-9. Switch and LED Array Controls

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

- Set values by passing an array of data.
- Modify the number of controls displayed based on the length of the specified values.

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 2-10 to control and visualize values of an array of `double` values.

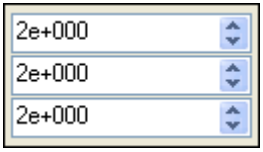


Figure 2-10. Numeric Edit Array Control

With the numeric edit array control and the classes that interface with the control, you can perform many operations, including:

- 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.

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*.

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 downlevel 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.

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 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 package 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, and cycle RMS average functions
- Special functions

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` and select **Visual Basic, Visual Basic .NET, C++, and C# with Measurement Studio**.

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 `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*.

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.

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*.

Network Variable

The Measurement Studio Network Variable .NET class library includes three namespaces: `NationalInstruments.NetworkVariable`, `NationalInstruments.NetworkVariable.WindowsForms`, and `NationalInstruments.NetworkVariable.WebForms`. Use the Network Variable class library to transfer live measurement data between applications and servers over the network.

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 the LabVIEW DSC Run-Time System.

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.Browser` classes to discover network variables and processes.
- Use the `NationalInstruments.NetworkVariable.ServerProcess`, `NationalInstruments.NetworkVariable.ServerProcessInfo`, `NationalInstruments.NetworkVariable.ServerVariable`, and `NationalInstruments.NetworkVariable.ServerVariableInfo` classes to explicitly create 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 `NationalInstruments.NetworkVariable.WindowsForms` class.

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.

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

Hardware Connectivity

NI is committed to providing seamless connectivity for data acquisition and instrument control devices, allowing you to quickly collect data and easily configure your test and measurement application. Measurement Studio hardware connectivity includes integrated data acquisition and instrument control functionality using the DAQ and instrument control assistants.

Note To use .NET class libraries that interface to National Instruments device drivers, NI-DAQmx, NI-VISA and NI-488.2, and the MAX (Measurement & Automation Explorer) configuration utility, you must install the underlying device drivers in addition to the .NET class libraries. You can run the underlying device driver installers from the NI Device Drivers CD. 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 CD.

Data Acquisition

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`. 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 DAQ, visit `ni.com/dataacquisition`.

Creating an NI-DAQmx Application

To create a Measurement Studio NI-DAQmx application, use the DAQ Assistant. The DAQ Assistant integrates into Visual Studio as a code designer. Use the Add New Item wizard to add an NI-DAQmx task to your project, and use the DAQ Assistant user interface, as shown in Figure 2-11, to interactively create and configure the NI-DAQmx task. The DAQ Assistant automatically generates a Visual Basic .NET, Visual C#, 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.

For step-by-step instructions on how to create DAQ applications, refer to the *Walkthrough: Creating a Measurement Studio NI-DAQmx Application* section in Chapter 3, *Getting Started with Measurement Studio*.

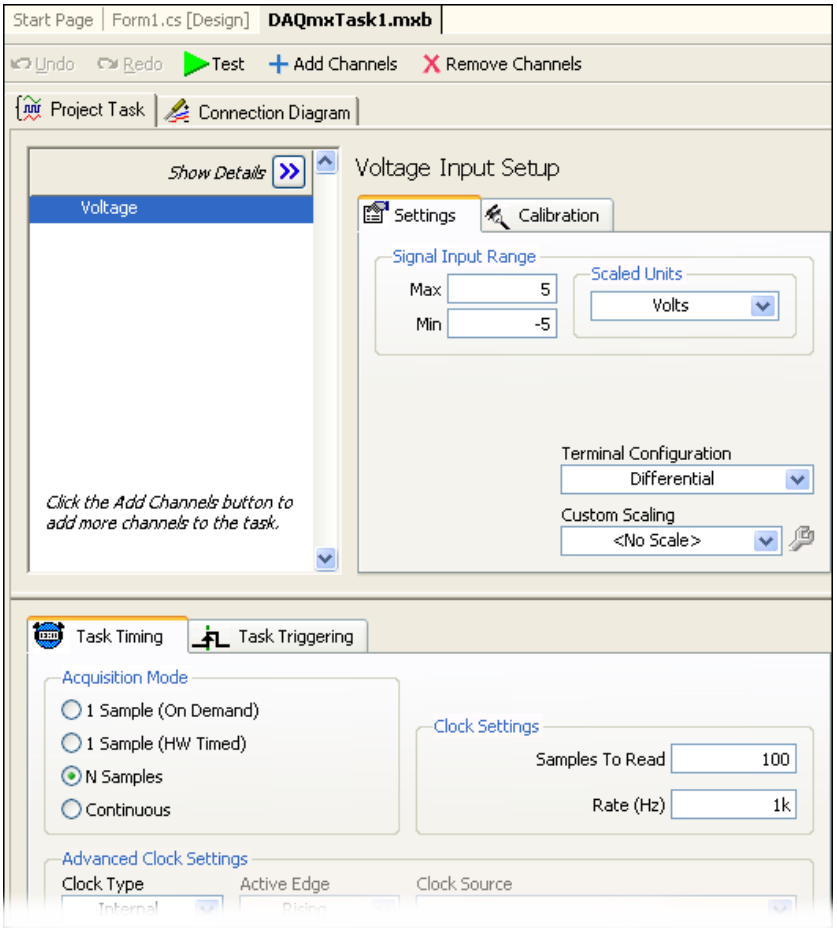


Figure 2-11. DAQ Assistant

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, Visual C#, 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*. For more information on how to create a user control, refer to the *Using a DAQmx Task Class in a Project* topic in the *NI Measurement Studio Help*.

Creating an NI-DAQmx User Interface

Using the Configure DAQ Component UI wizard, as shown in Figure 2-12, 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. The Configure DAQ Component UI wizard is available only in Visual Studio 2005.

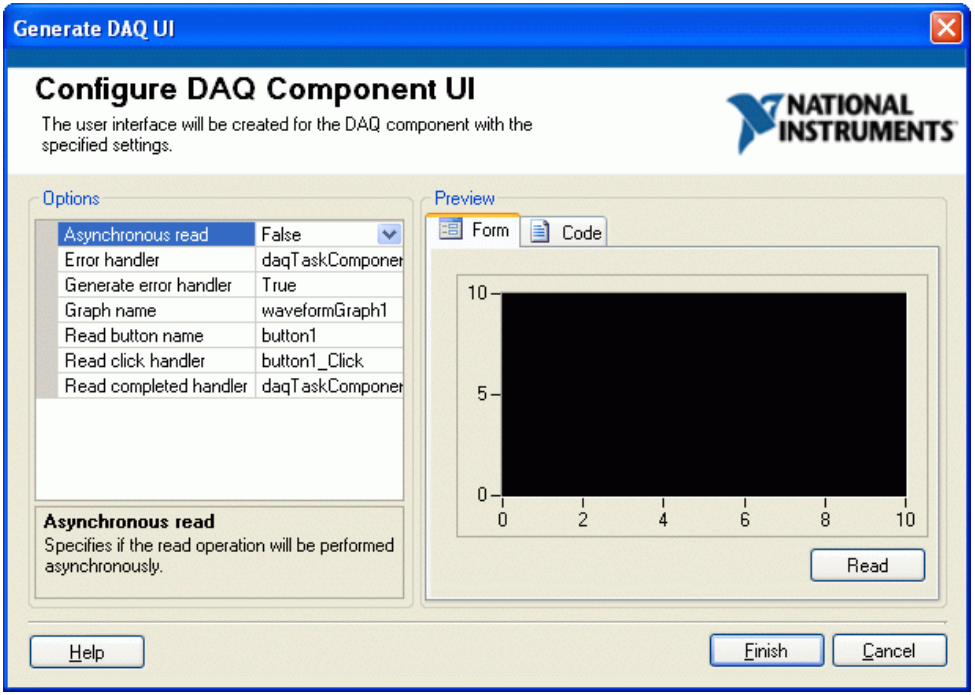


Figure 2-12. Configure DAQ Component UI Wizard

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*.

Instrument Control

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`. 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, refer to `ni.com/gpib`.

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 easily 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 3, *Getting Started with Measurement Studio*. For more information about NI-VISA, refer to `ni.com/visa`.

Modular Instruments Support

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.

Tip 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 at NI Developer Zone, `ni.com/devzone`.

Creating a Measurement Studio Instrument Control Application

To create a Measurement Studio instrument control application, use the Instrument I/O Assistant. The Instrument I/O Assistant, as shown in Figure 2-13, integrates into Visual Studio as a code designer. Use the Add New Item wizard to add an instrumentation task to your project, and 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, Visual C#, 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 3, *Getting Started with Measurement Studio*, the *Walkthrough: Creating a Measurement Studio Instrument I/O Application* section for step-by-step instructions on how to use the Instrument I/O Assistant.

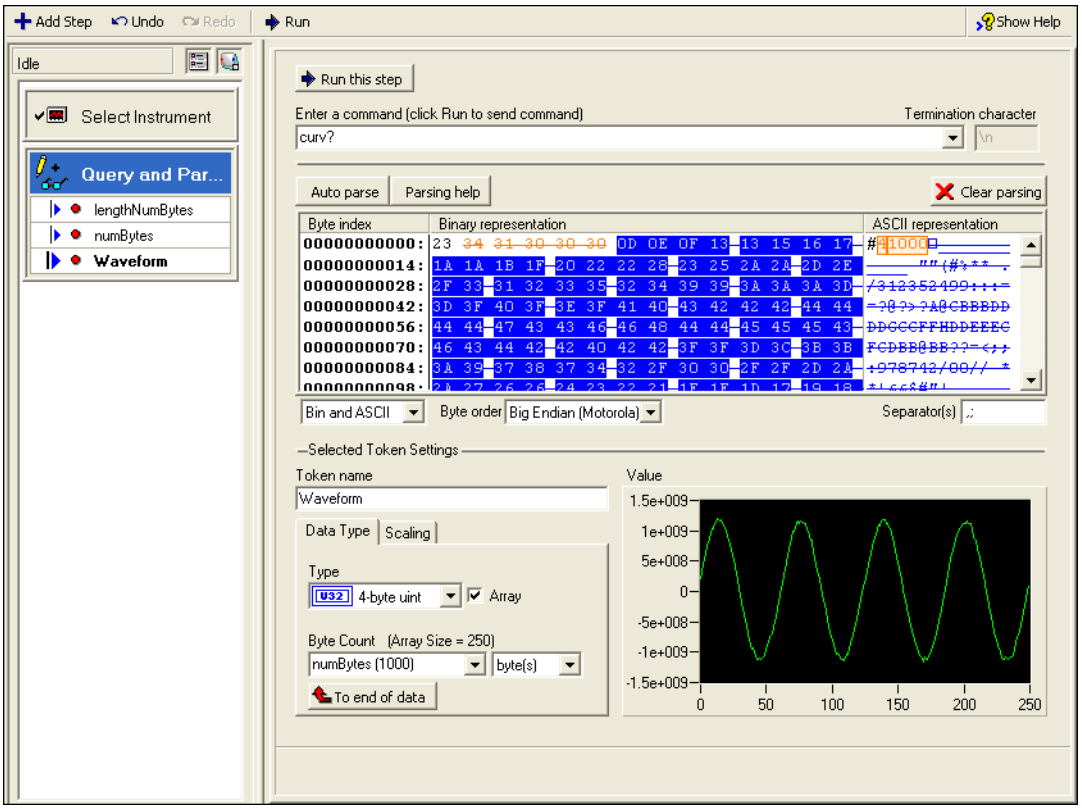


Figure 2-13. Instrument I/O Assistant

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*.

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 2-14, 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.

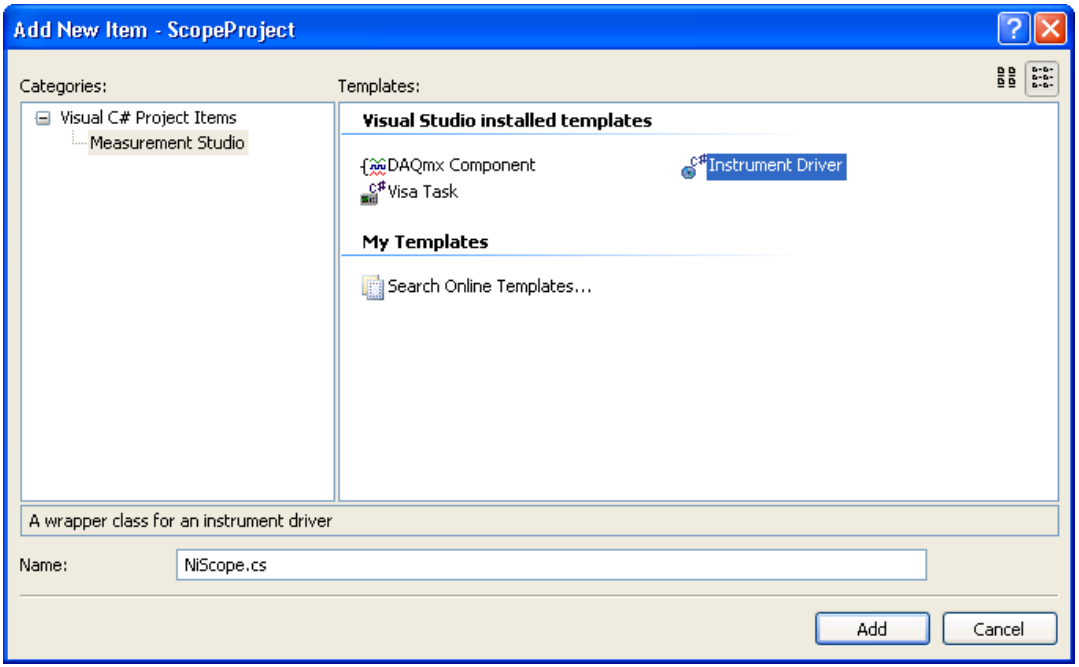


Figure 2-14. Launching the Measurement Studio .NET Instrument Driver Wizard from the Add New Item Wizard

Tip For information about the .NET instrument driver wizard, refer to the *Using Instrument Drivers in Measurement Studio Applications* section in the *NI Measurement Studio Help*.

Measurement Studio Integration with Visual Studio

Measurement Studio seamlessly integrates into Visual Studio, allowing you to quickly create test and measurement applications without ever leaving the Visual Studio environment.

Measurement Studio Menu

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

- **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, Visual C#, 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»Variable Manager**—Use Variable 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»Variable Manager** to access this menu item.
- **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 Online Resources»Measurement Studio Home Page**—Use the Measurement Studio Web site at `ni.com/mstudio` to find Measurement Studio news, support, downloads, and evaluation software. Select **Measurement Studio Online Resources»Measurement Studio Home Page** to access this menu item.
- **Measurement Studio Online Resources»Instrument Driver Network**—Use the NI Instrument Driver Network at `ni.com/idnet` as a central resource for downloading, developing, and submitting instrument drivers. Select **Measurement Studio Online Resources»Instrument Driver Network** to access this menu item.

- **Measurement Studio Online Resources»Discussion Forums**—Use the NI Discussion Forums at `forums.ni.com` to participate in discussion forums and exchange code with measurement and automation developers around the world. Select **Measurement Studio Online Resources»Discussion Forums** to access this menu item.
- **Measurement Studio Online Resources»Search Technical Support**—Use NI Technical Support at `ni.com/support` to find support resources available for most products, including software drivers and updates, KnowledgeBase articles, product manuals, step-by-step troubleshooting wizards, conformity documentation, example code, tutorials and application notes, instrument drivers, discussion forums, and a measurement glossary. Select **Measurement Studio Online Resources»Search Technical Support** to access this menu item.
- **Measurement Studio Online Resources»NI Developer Zone**—NI Developer Zone, `zone.ni.com`, provides access to online example programs, tutorials, technical news, and a Measurement Studio Discussion Forum where you can participate in discussion forums for Visual Basic 6.0, Visual C++, and .NET Languages. Select **Measurement Studio Online Resources»NI Developer Zone** to access this menu item.
- **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.
- **Preferences**—Use the Measurement Studio Preferences dialog box to configure Measurement Studio settings, such as conversion options and add-in preferences. Select **Tools»Options** to access this menu item.

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, Visual C#, ASP.NET, and Visual C++. Refer to the following sections, *Walkthrough: Creating an Application with Windows Forms Controls and Analysis* or *Walkthrough: Creating an Application with Web Forms Controls and Analysis*, 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 2-15, 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 project
- Measurement Studio Visual C++ project (Visual Studio 2005 only)
- Measurement Studio Visual C++ project with LabWindows/CVI libraries (Visual Studio 2005 only)

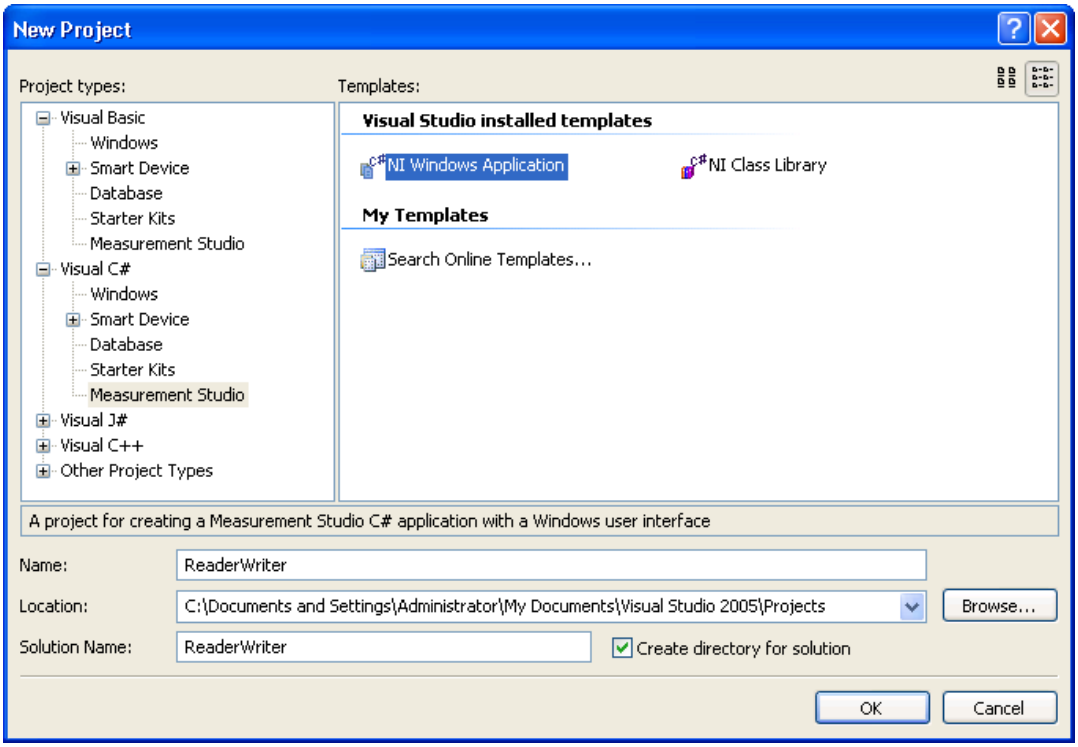


Figure 2-15. New Project Dialog Box in Visual Studio 2005

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*.

3

Getting Started 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 with Measurement Studio* section and the *Getting Started with the Measurement Studio Class Libraries* section of the *NI Measurement Studio Help* for more information about the functionality of these tools and features.

Measurement Studio Walkthroughs

Use the following walkthroughs to help you develop Measurement Studio applications in Visual Studio 2005 or Visual Studio 2008:

- *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 Web 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 package installed for Visual Studio 2005 or later. This walkthrough will not work with the Measurement Studio Standard package.

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.

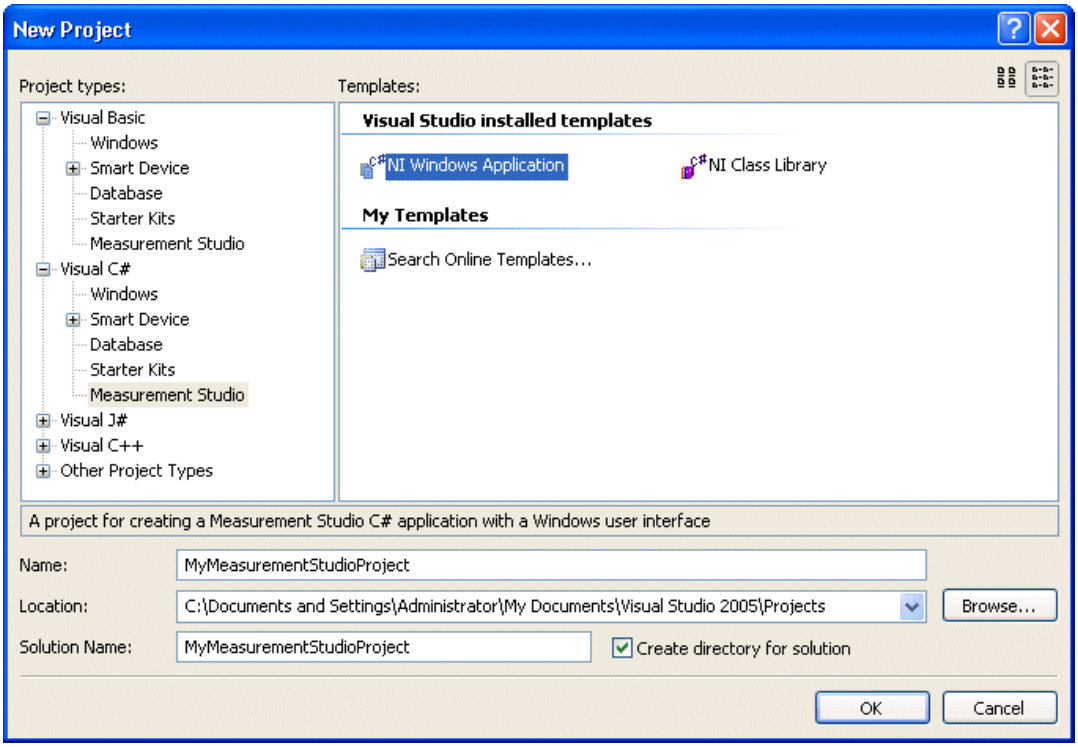
Before you begin

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 package) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise package) for Visual Studio 2008

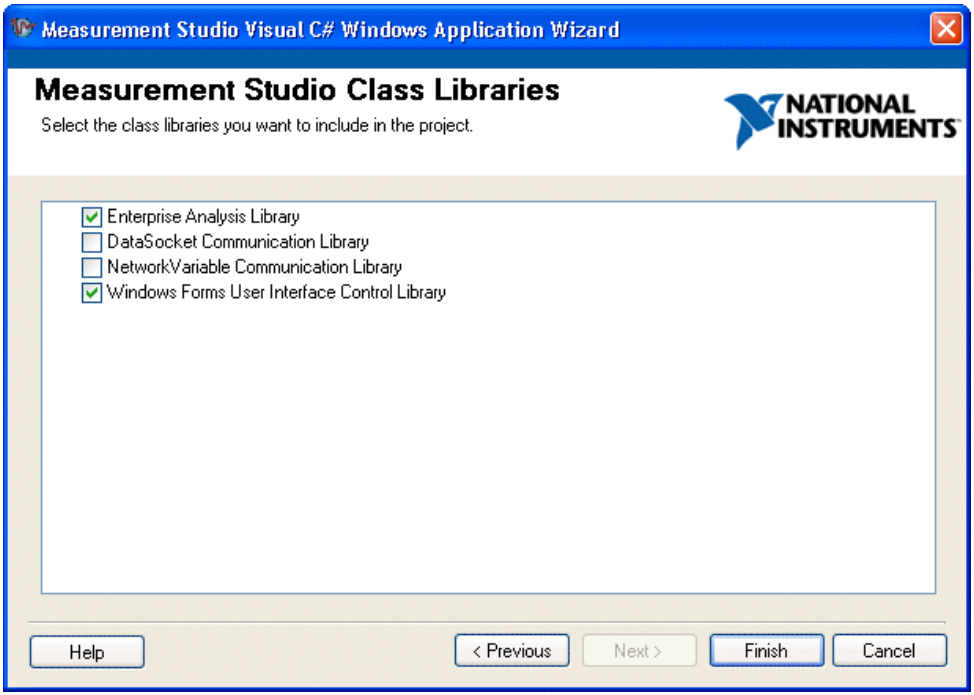
Setting up the project

1. Select **Start»All Programs»Microsoft Visual Studio 2005» Microsoft Visual Studio 2005** or **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**. Specify `MyMeasurementStudioProject` for **Name** and specify a **Location** of your choice.
5. Click **OK**. The Measurement Studio Application Wizard launches.

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



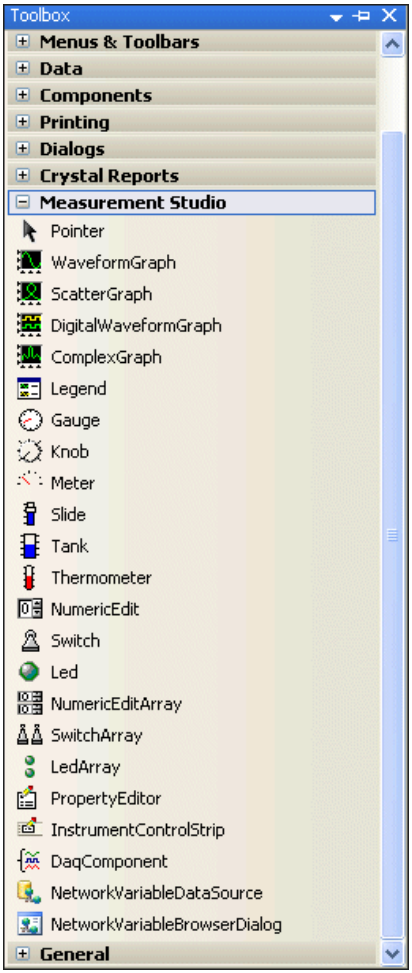
Tip If you are working with an existing project, you can access the Add/Remove Class Libraries dialog box by selecting **Measurement Studio»View .NET Class Library Wizard**.

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

Adding user interface controls to the project

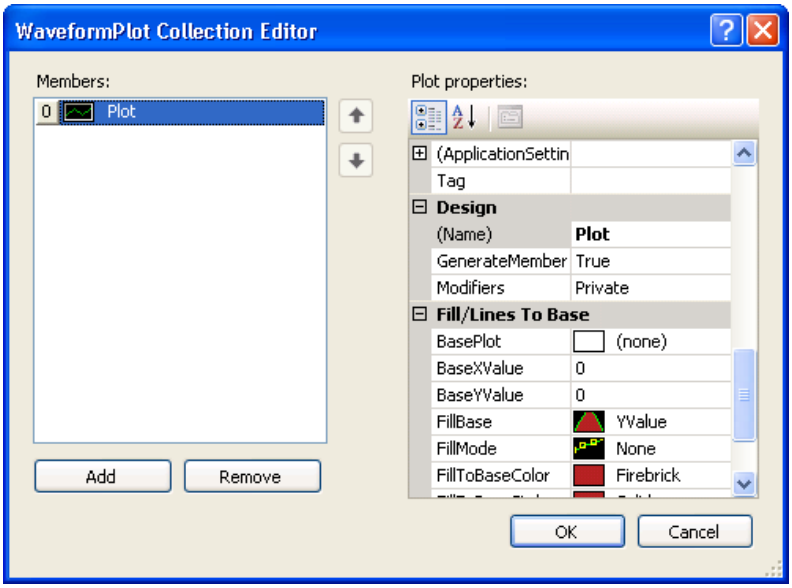
1. Select **View»Toolbox** to display the Toolbox. The Toolbox contains components and controls that you can add to your project.
2. Expand the **All Windows Forms** group. The All Windows Forms group contains controls and components included in the `System.WindowsForms` namespace.
3. Select the **Button** 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.

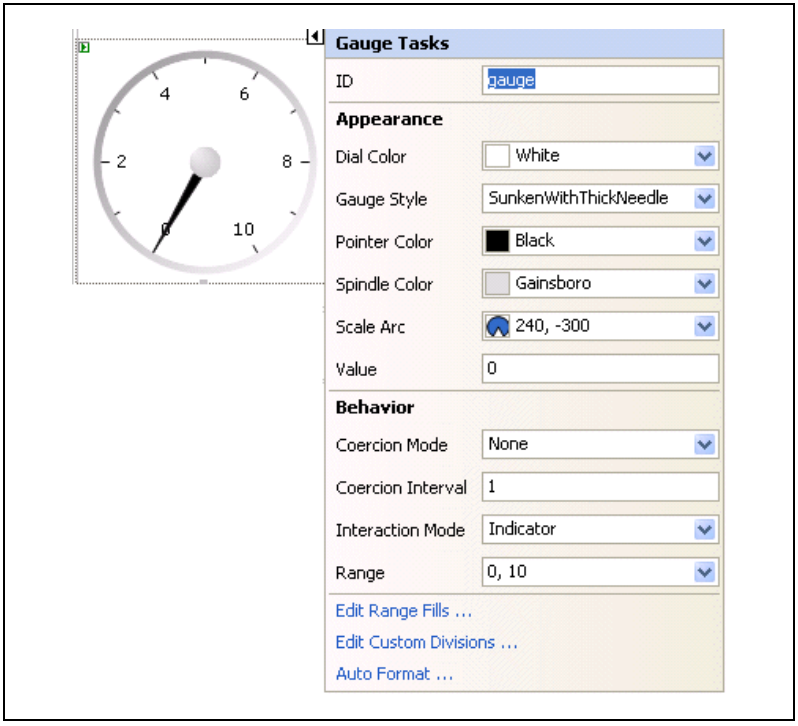
8. Right-click the waveform graph and select **Edit Plots** to display the WaveformPlot Collection Editor dialog box. You use the WaveformPlot Collection Editor dialog box to add or remove plots and to configure plot properties.



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.

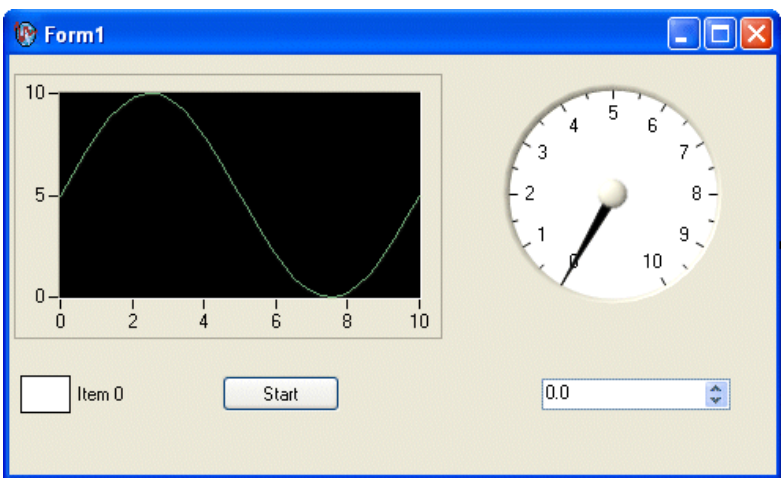
9. Type `Plot` for the Name. Click **OK**.
10. Before you add the Measurement Studio legend, numeric edit, and gauge controls, you need to resize the form to accommodate them. Select the form and use the double-sided arrow to resize it.
11. Select the **Legend** control and drag and drop it onto the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

The following screenshot shows `Form1` with the user controls.



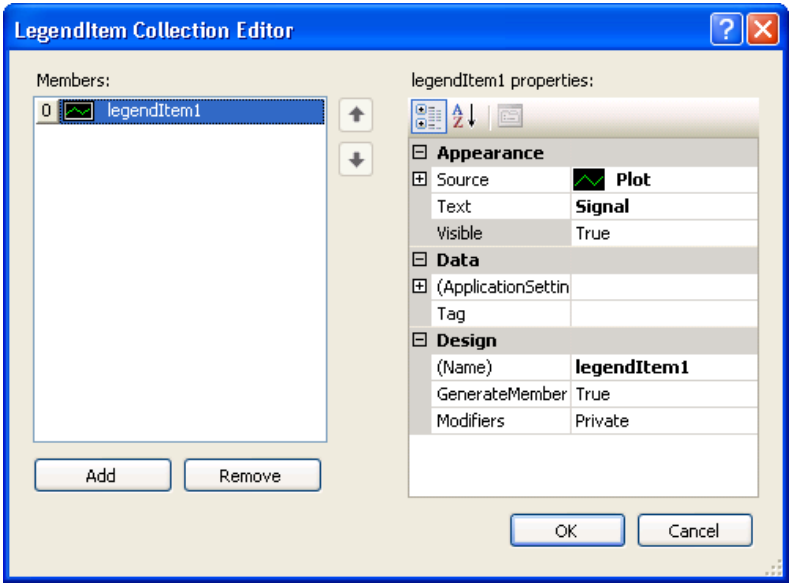
Generating, plotting, and analyzing the data

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.

```
[VB.NET]
' 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;
```

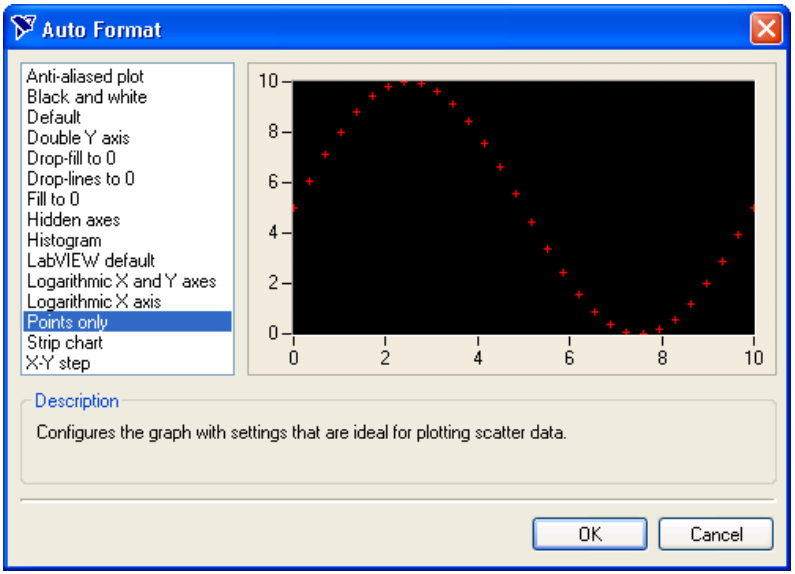
Customizing your user interface

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.



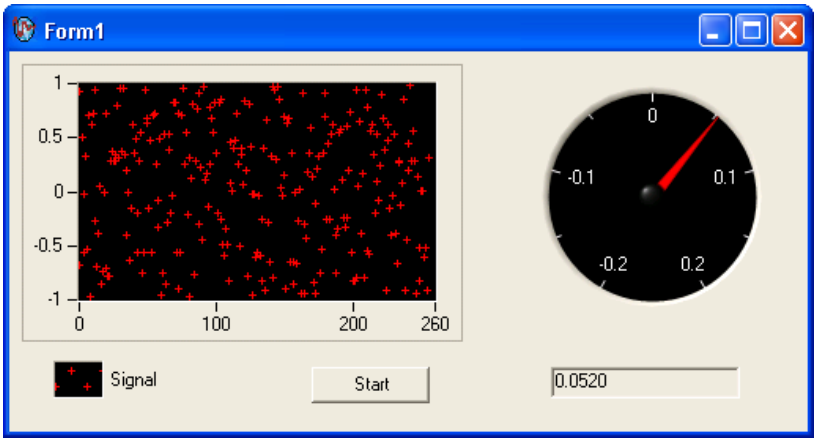
2. Select **Plot** in the **Source** drop-down list and enter `Signal` in the **Text** box. Click **OK**. Now that you have specified a legend item for the plot, changes you make to the plot will be reflected on the legend.
3. Right-click the graph and select **Auto Format** to display the Auto Format 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 Numeric Edit 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.
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 package installed for Visual Studio 2005 or later. This walkthrough will not work with the Measurement Studio Standard package.

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.

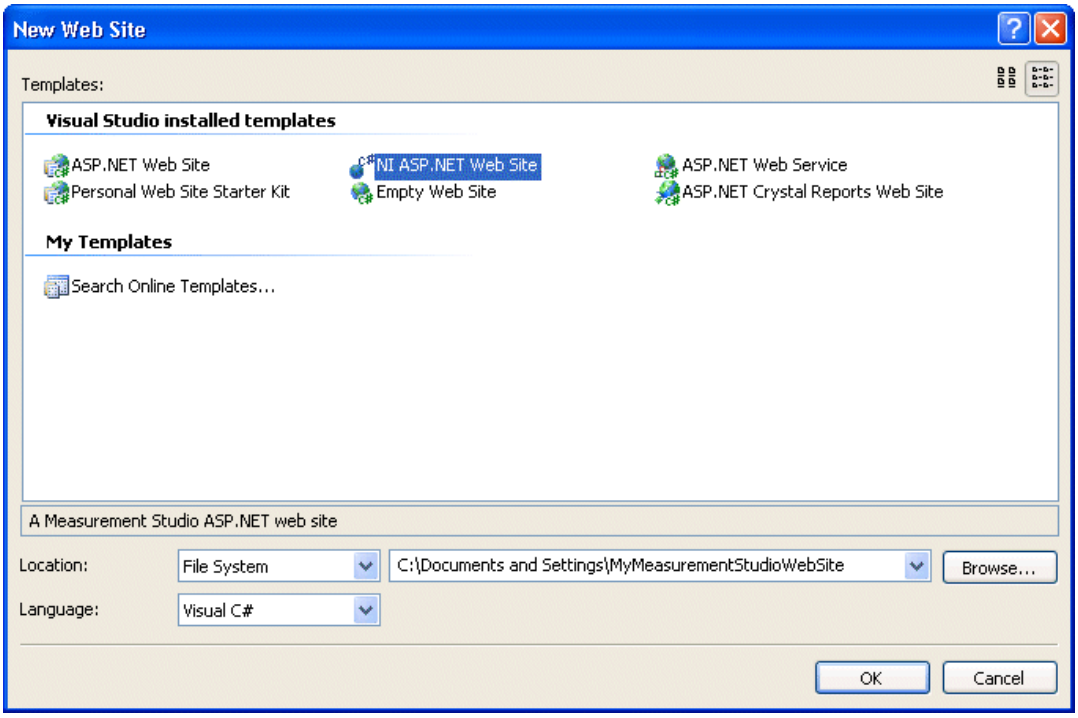
Before you begin

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 package) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise package) for Visual Studio 2008

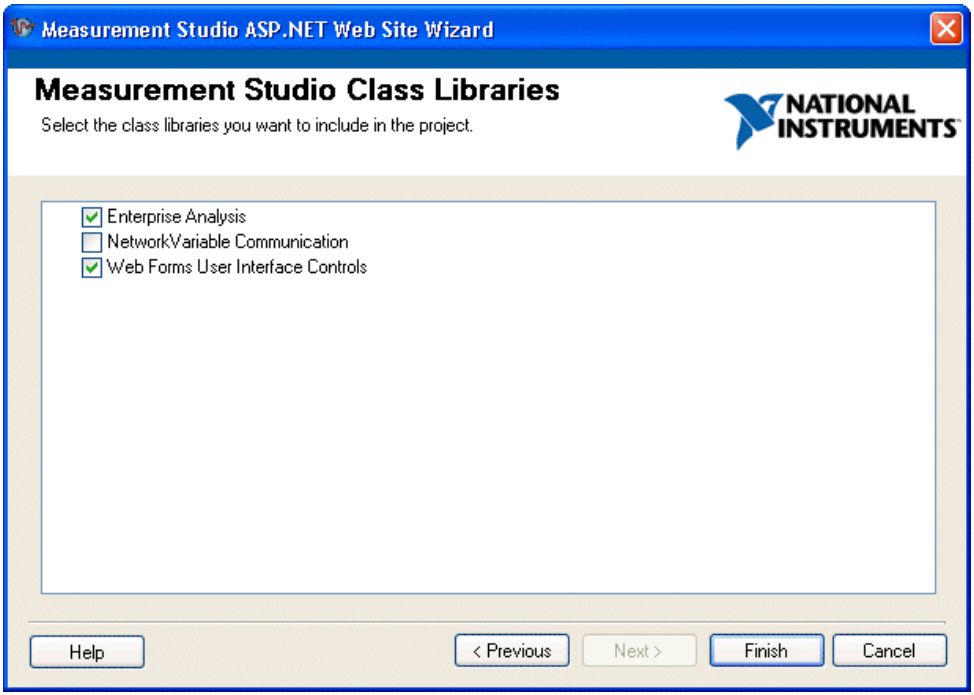
Setting up the project

1. Select **Start»All Programs»Microsoft Visual Studio 2005» Microsoft Visual Studio 2005** or **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 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 **Analysis Library** and **Web Forms User Interface Control Library**.

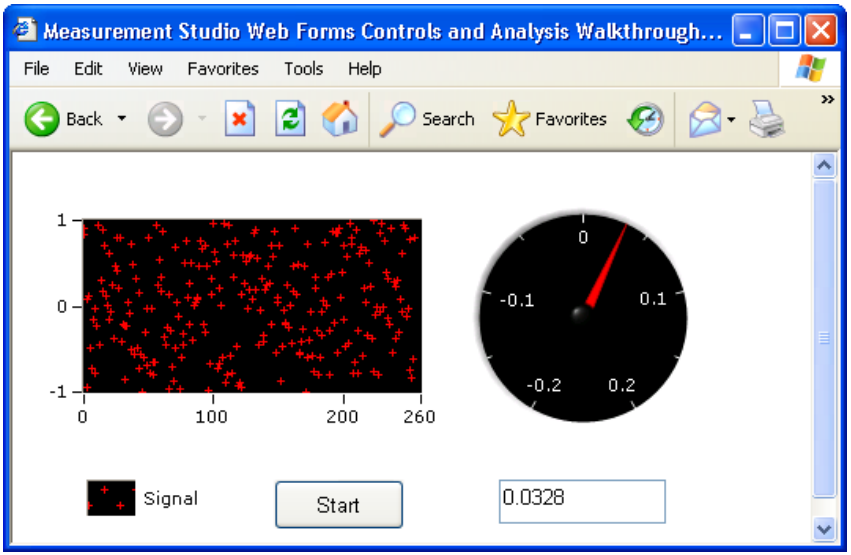


Tip If you are working with an existing project, you can access the Add/Remove Class Libraries dialog box by selecting **Measurement Studio»Add/Remove .NET Class Libraries Wizard**.

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**.

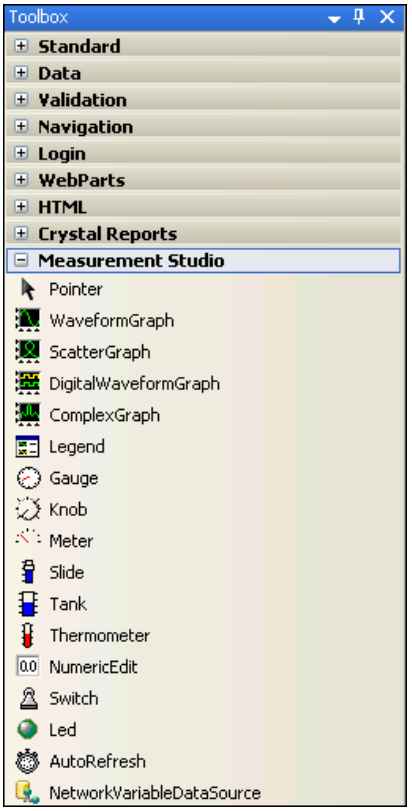
Adding user interface controls to the project

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



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 **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.
4. The default table that appears is 3 × 3. 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.
5. If you are using Visual Studio 2005, merge the top two cells of all three columns by selecting the cells, right-clicking, and selecting **Merge Cells**. If you are using Visual Studio 2008, merge the top two cells of all three columns by selecting the cells, right-clicking, and selecting **Modify»Merge Cells**.
6. Expand the **Standard** group on the Toolbox. The Standard group contains ASP.NET server controls included in the `System.Web.UI` namespace.

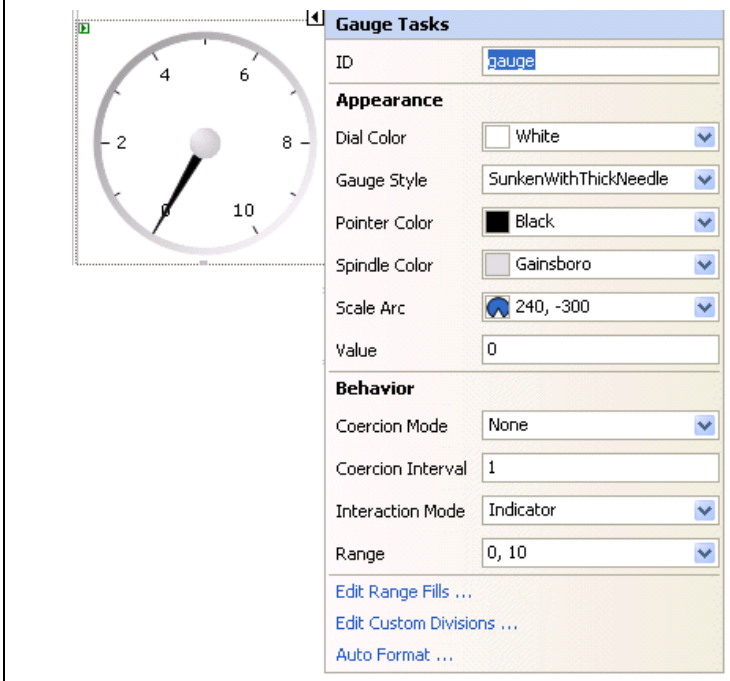
7. Select the **Button** control and drag and drop it into the lower right table cell.
8. Right-click the button and select **Properties** to display the Properties window. You configure the properties of the control in the Properties window.
9. Scroll to the Text property in the Properties window. Type `Start` for the button text.
10. Expand the **Measurement Studio** group on the Toolbox.



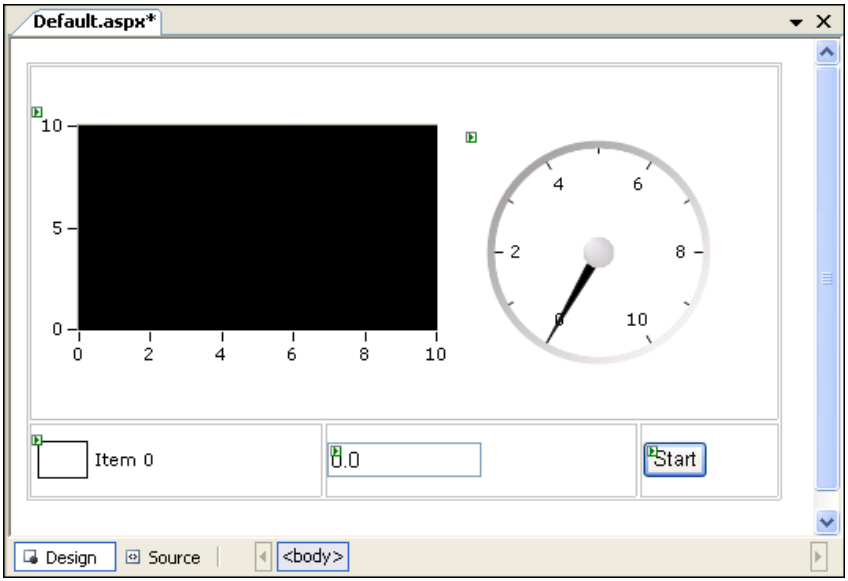
11. Select the **WaveformGraph** control and drag and drop it into the top table cell.
12. On the waveform graph smart tag, type `graph` for the name of the waveform graph ID.

Tip 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.

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



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



Generating, plotting, and analyzing the data

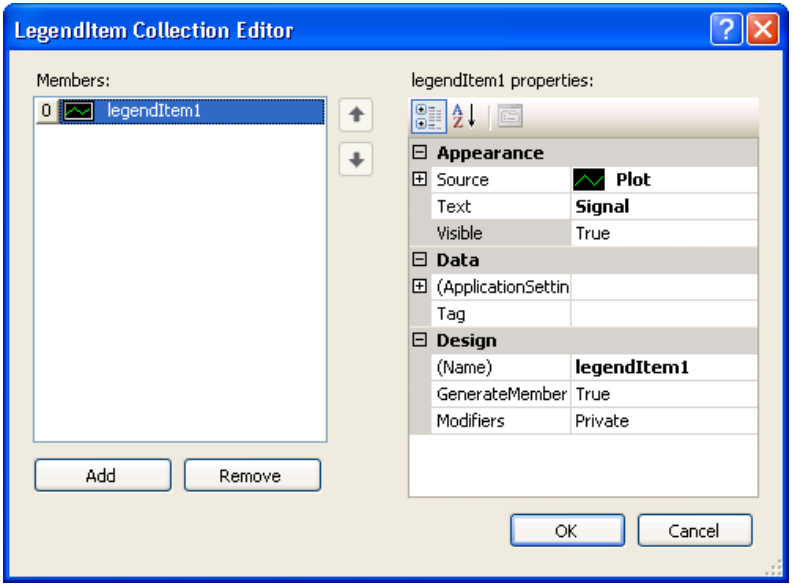
1. Double-click the button control to display the `Default.aspx.cs` code, with the cursor inside the click event handler of the button control.
2. Add the following code to generate random data, plot the data, calculate the mean of the data, and display the mean on the gauge.

```
[VB.NET]
' 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
```

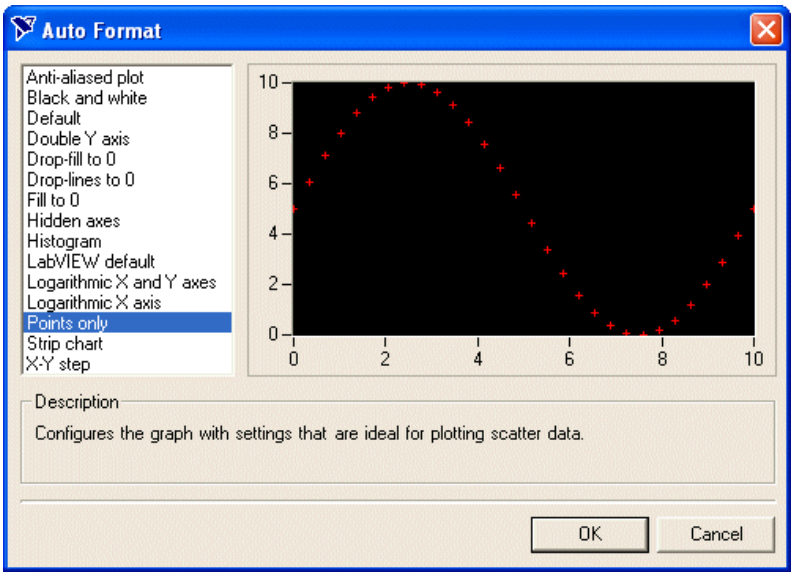
```
[C#]
// Declare and initialize an instance of WhiteNoiseSignal.
WhiteNoiseSignal whiteNoise = new WhiteNoiseSignal();

// 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.

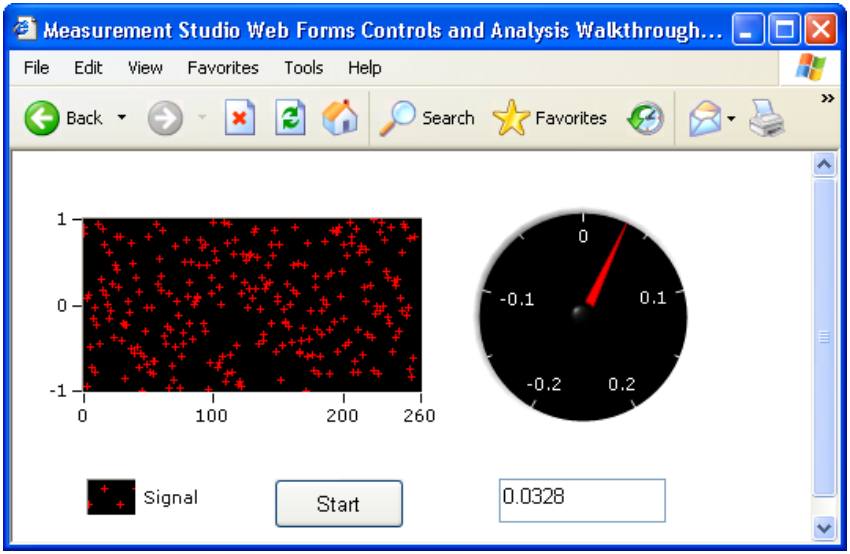


3. Select **Plots[0]** in the **Source** drop-down list and enter `Signal` in the **Text** box. Click **OK**. Now that you have specified a legend item for the plot, changes you make to the plot will be reflected on the legend.
4. Right-click the graph and select **Auto Format** to display the Auto Format 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 package installed for Visual Studio 2005 or later. This walkthrough will not work with the Measurement Studio Standard package.

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.

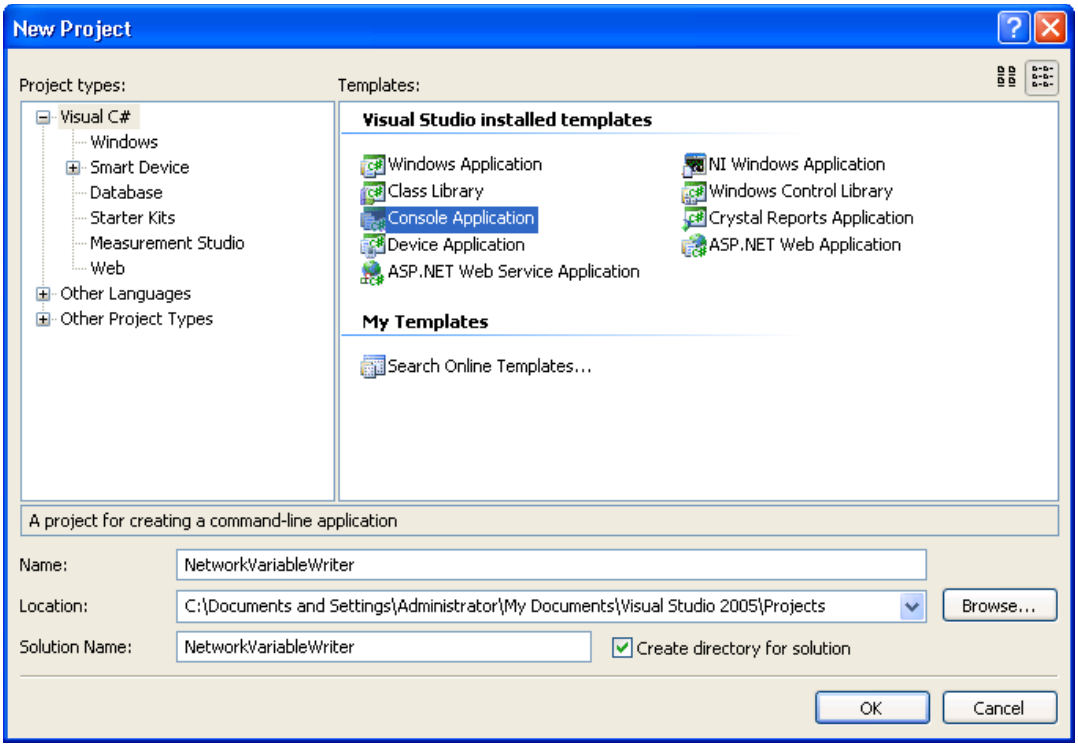
Before you begin

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 package) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise package) for Visual Studio 2008

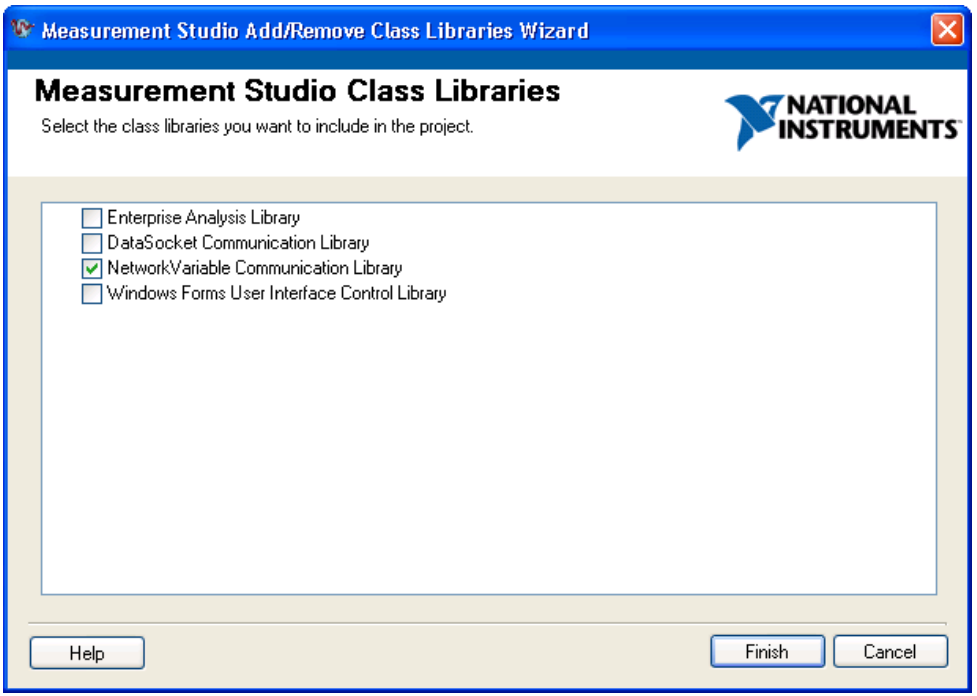
Writing an array of data to the server

1. Select **Start»All Programs»Microsoft Visual Studio 2005» Microsoft Visual Studio 2005** or **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:

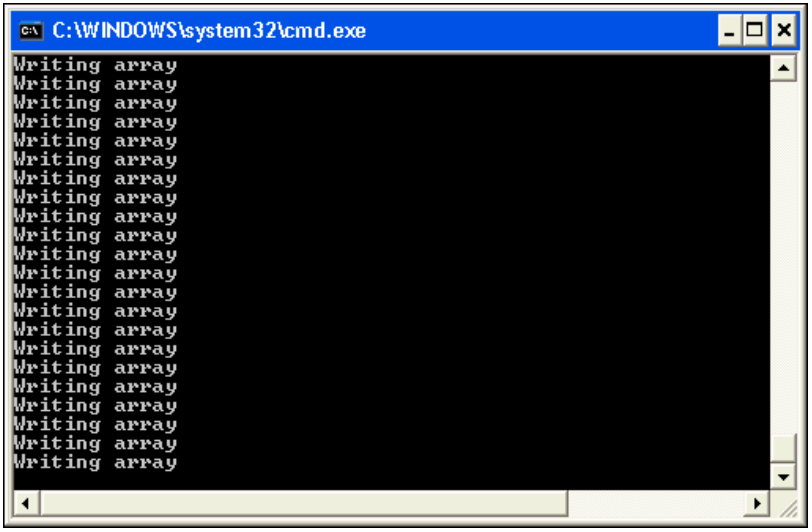
Note You should choose the appropriate code depending on whether you created a VB or C# project.

```
[VB.NET]
Imports System
Imports System.Collections.Generic
Imports System.Text
Imports System.Threading
Imports NationalInstruments.NetworkVariable

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++;
            }
        }
    }
}
```

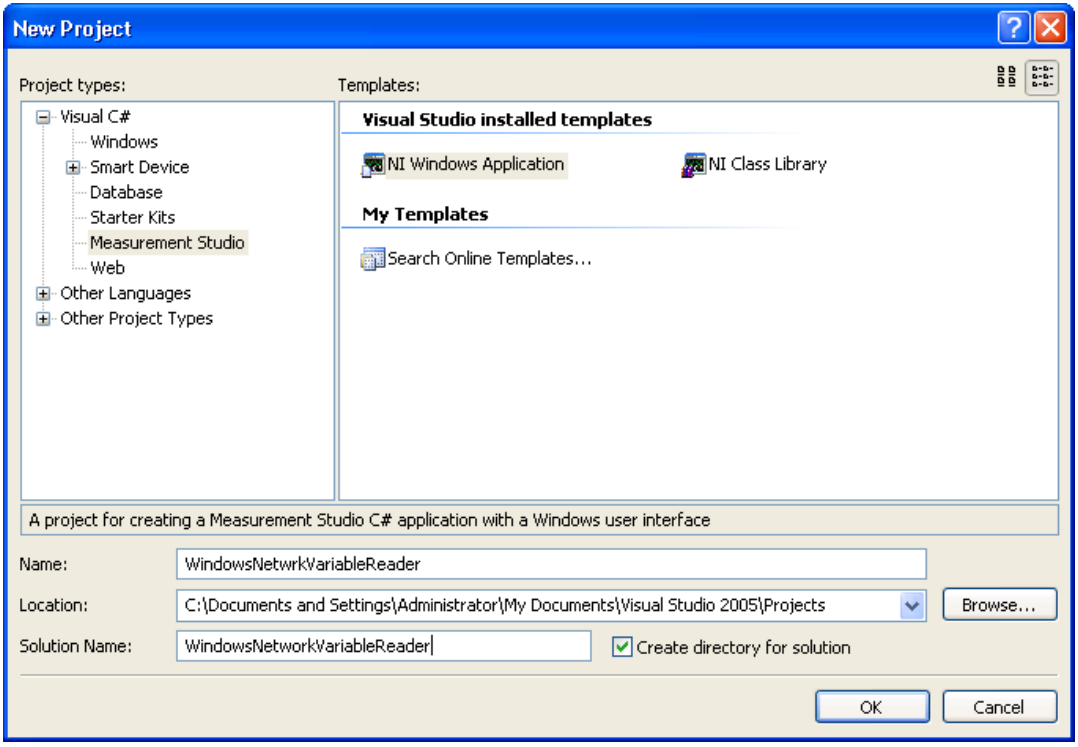
9. Select **Debug»Start Without Debugging** to run the application.



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

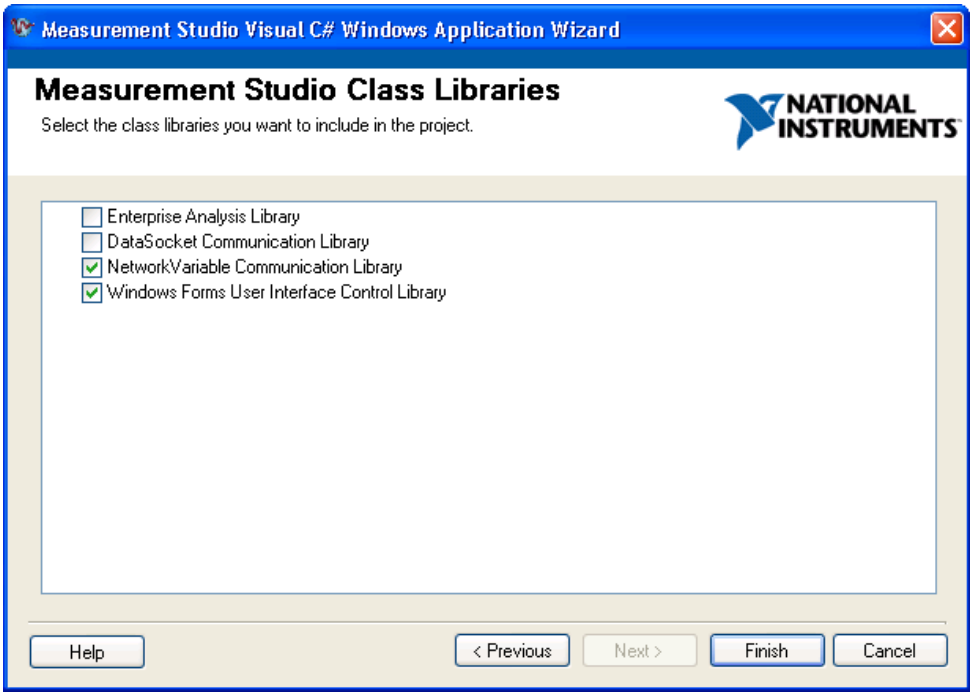
Setting up a Windows Forms project

1. Select **Start»All Programs»Microsoft Visual Studio 2005» Microsoft Visual Studio 2005**.
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**.

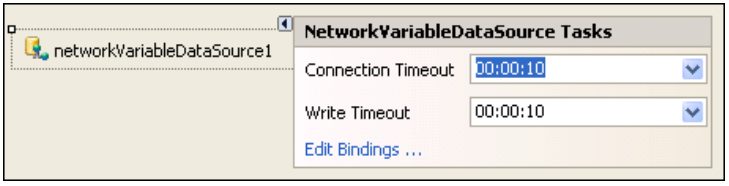


Tip If you are working with an existing project, you can access the Add/Remove Class Libraries dialog box by selecting **Measurement Studio»Add/Remove Class Libraries Wizard**.

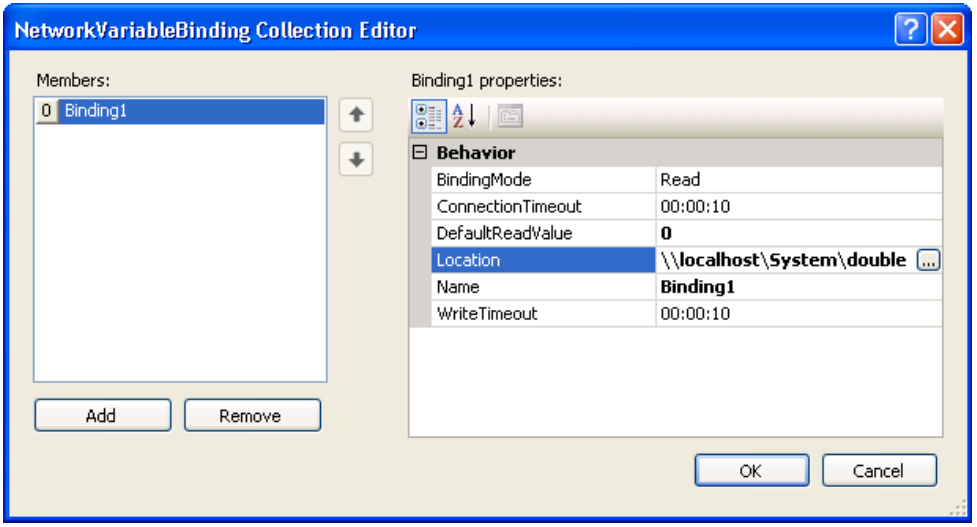
7. Click **Finish** to display `Form1` in the Windows Forms Designer.

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 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 NetworkVariableDataSource control encapsulates Network Variable 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**.



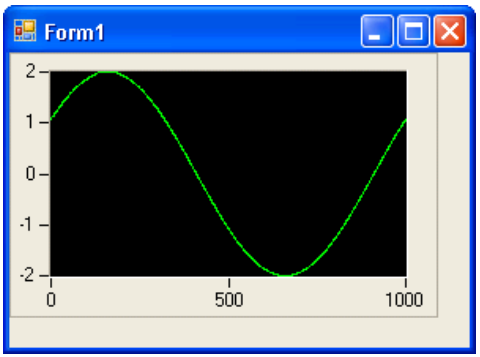
6. For the **Location**, browse to the `\\localhost\System\double` location in the Select Network Item dialog box.

7. Click **OK** to return to the NetworkVariableBinding Collection Editor dialog box.
8. After you configure the binding properties, 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 package installed for Visual Studio 2005 or later. This walkthrough will not work with the Measurement Studio Standard package.

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.

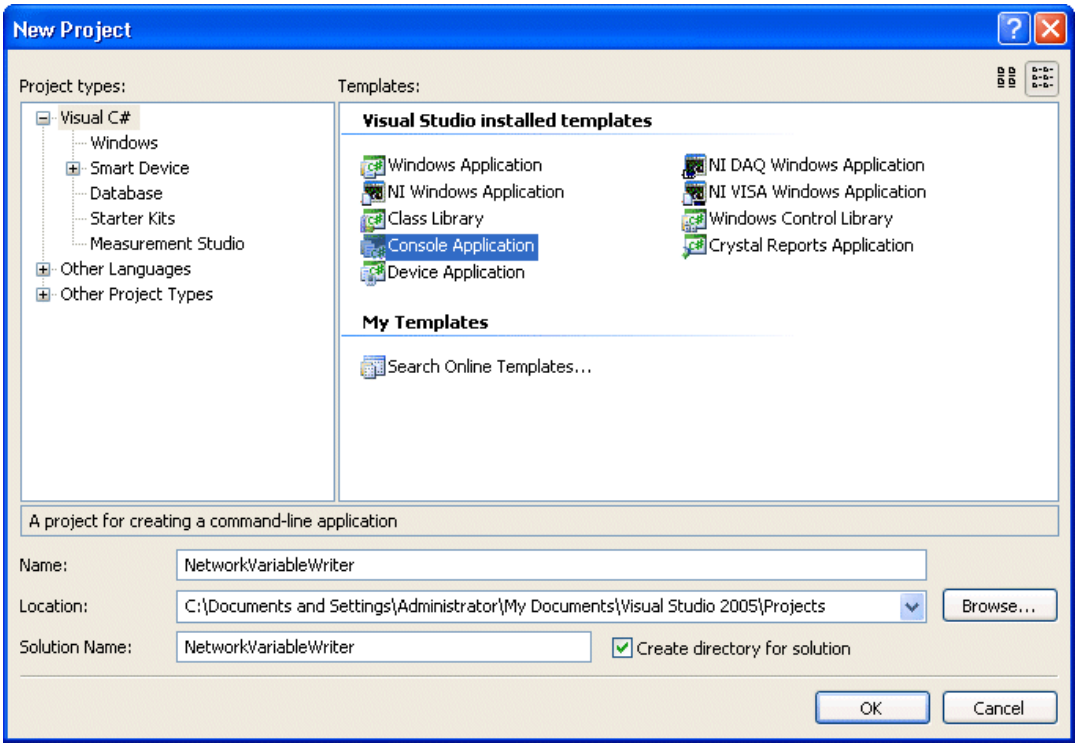
Before you begin

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 package) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise package) for Visual Studio 2008

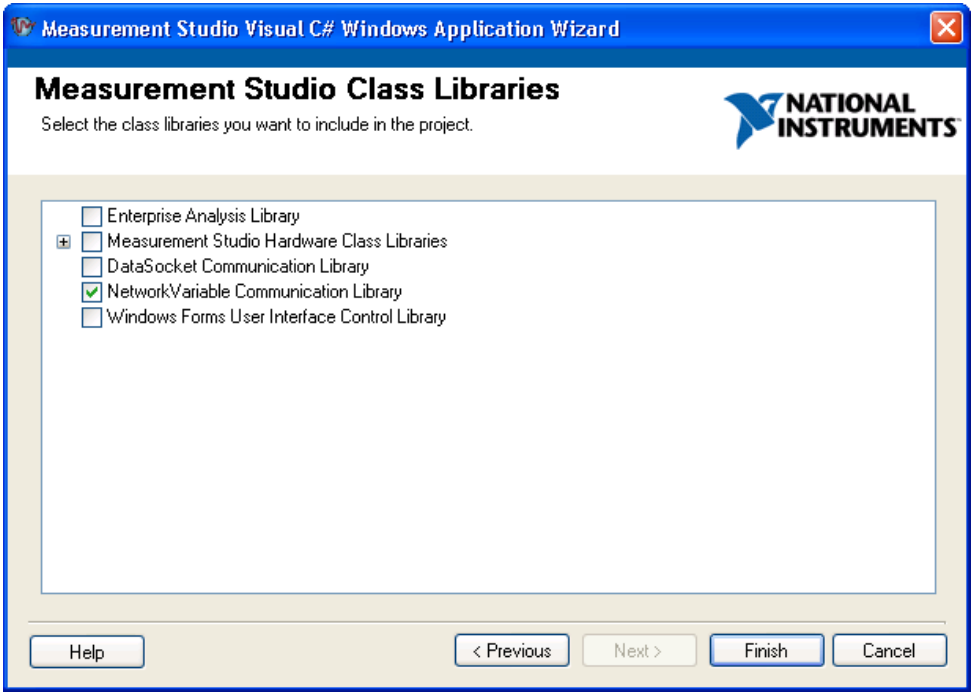
Writing an array of data to the server

1. Select **Start»All Programs»Microsoft Visual Studio 2005» Microsoft Visual Studio 2005** or **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**.

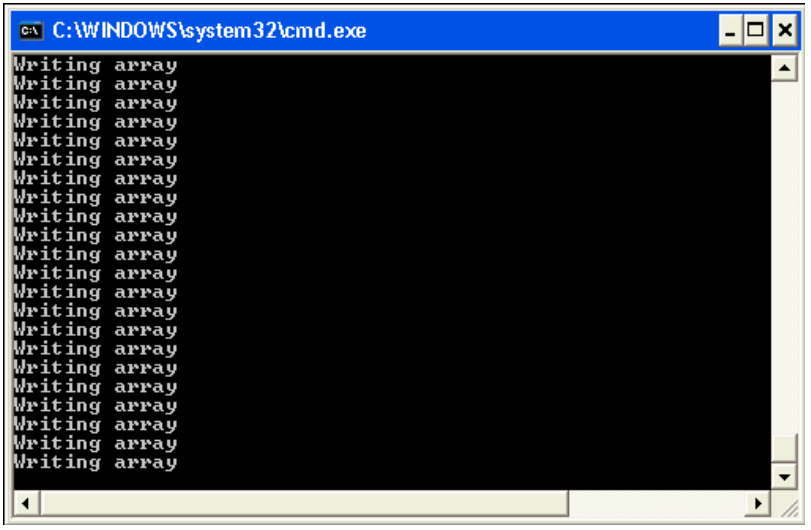


8. In `Program.cs`, add the following code to write an array of data to the server:

```
[VB.NET]
Imports NationalInstruments.NetworkVariable
Imports System.Threading
Imports System

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.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[]> 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++;
            }
        }
    }
}
```

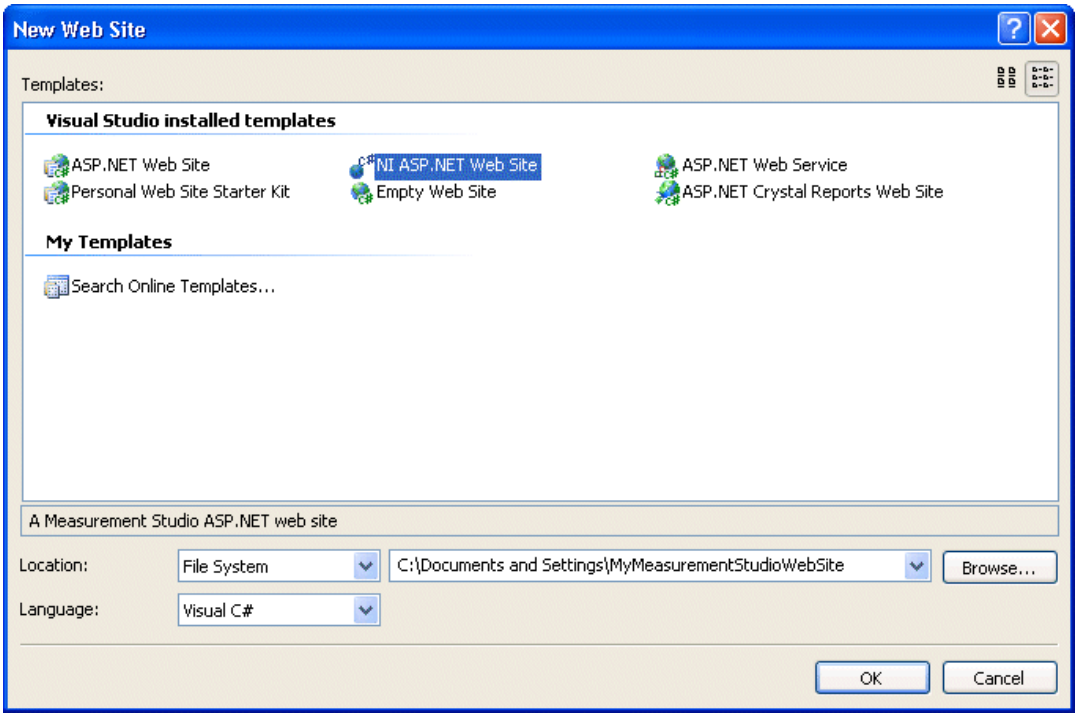
9. Select **Debug»Start Without Debugging** to run the application.



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

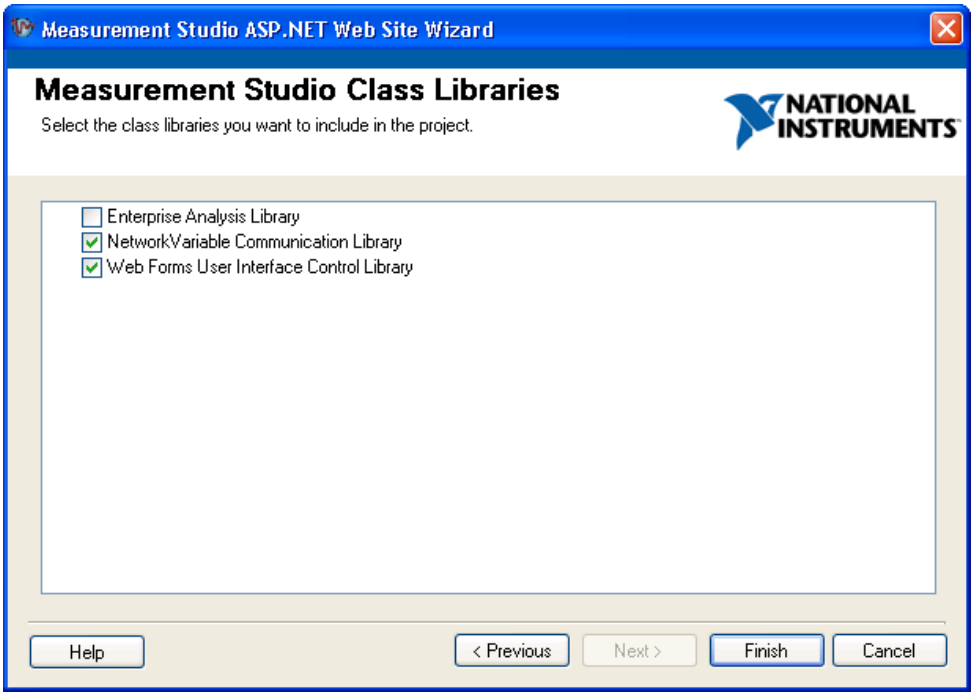
Setting up a Web Forms project

1. Select **Start»All Programs»Microsoft Visual Studio 2005» Microsoft Visual Studio 2005** or **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**.

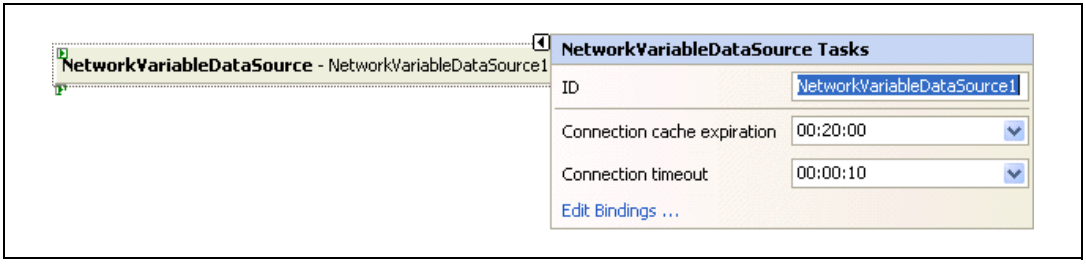


Tip If you are working with an existing project, you can access the Add/Remove Class Libraries dialog box by selecting **Measurement Studio»Add/Remove Class Libraries Wizard**.

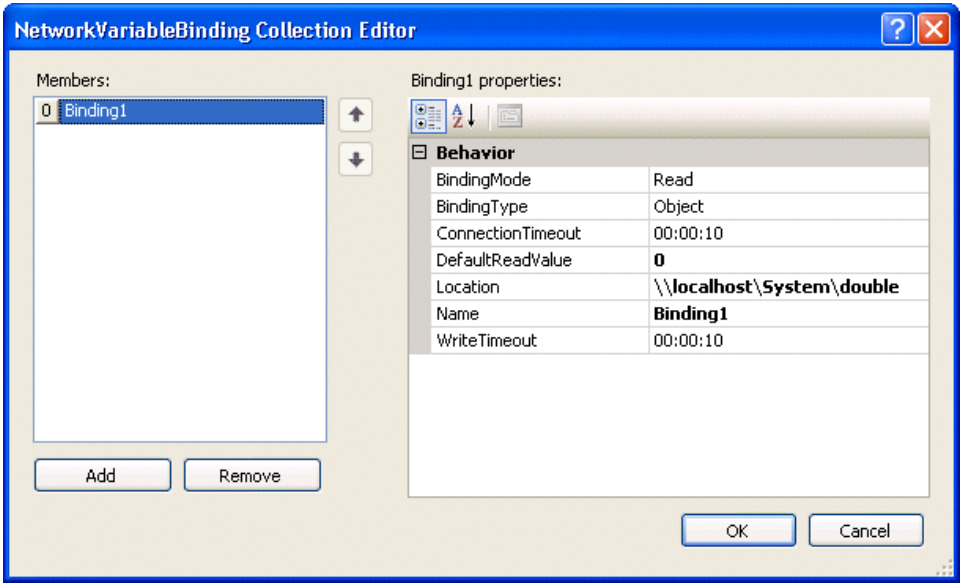
7. Click **Finish** to display `Default.aspx` in the Web Forms Designer.
8. You can rename the title of your Web page. Click inside the `<title>` tag and rename the title to **Measurement Studio Network Variable and Web Forms Controls Walkthrough**.

Configuring the network variable data source control

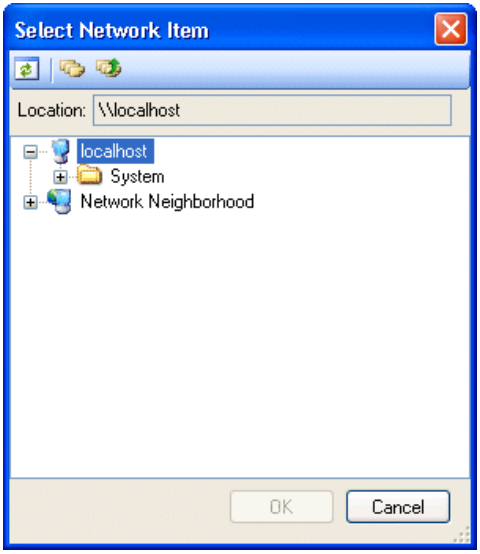
1. Click **Design** in the lower left corner to switch from Source View to Design View.
2. Select **View»Toolbox** to display the Toolbox. The toolbox contains components and controls that you can add to your project.
3. Expand the **Measurement Studio** group on the Toolbox.
4. Select the NetworkVariableDataSource control in the toolbox and drag and drop it on the form. The 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 NetworkVariableDataSource control encapsulates Network Variable 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**.



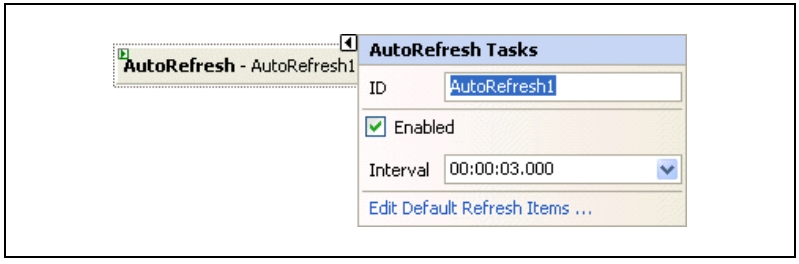
7. Browse to the `\\localhost\System\double` location in the Select Network Item dialog box.



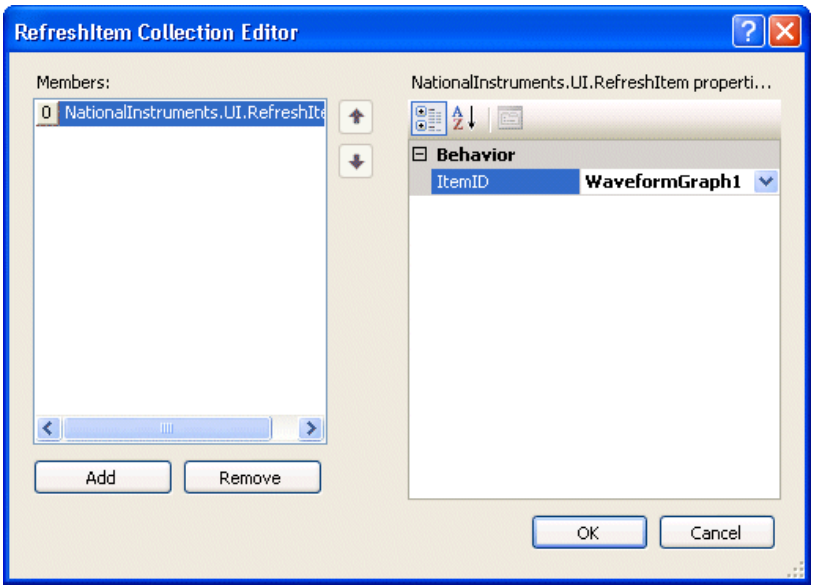
8. Click **OK** to return to the NetworkVariableBinding Collection Editor dialog box.
9. After you configure the binding properties, click **OK** to return to the ASP.NET Designer.

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.



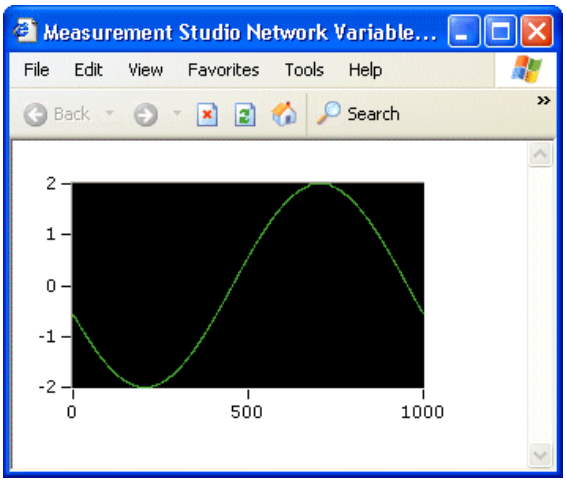
4. Select **Add**. Select **WaveformGraph1** for the ItemID and click **OK**.



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:

```
[VB.NET]
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 applicationby 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:

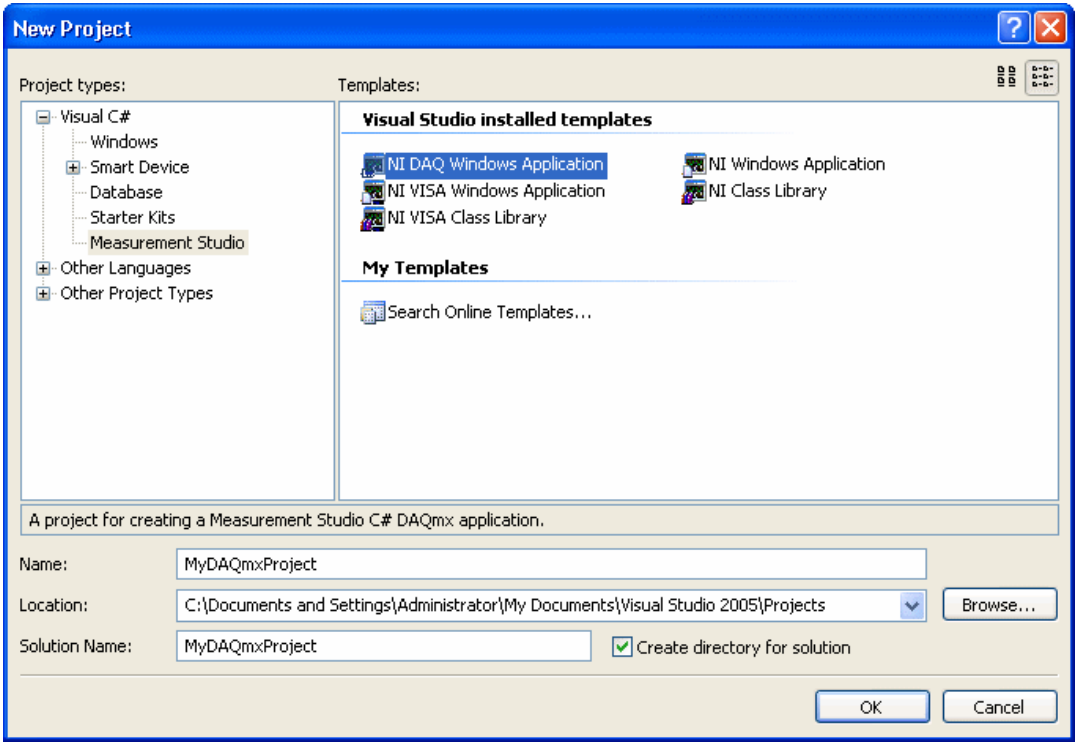
- Visual Studio 2005 or Visual Studio 2008
- Measurement Studio 8.0.1 or later (Professional or Enterprise package) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise package) for Visual Studio 2008
- NI-DAQmx 8.1 or later for Visual Studio 2005 or NI-DAQmx 8.7.1 or 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 a simulated device 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.

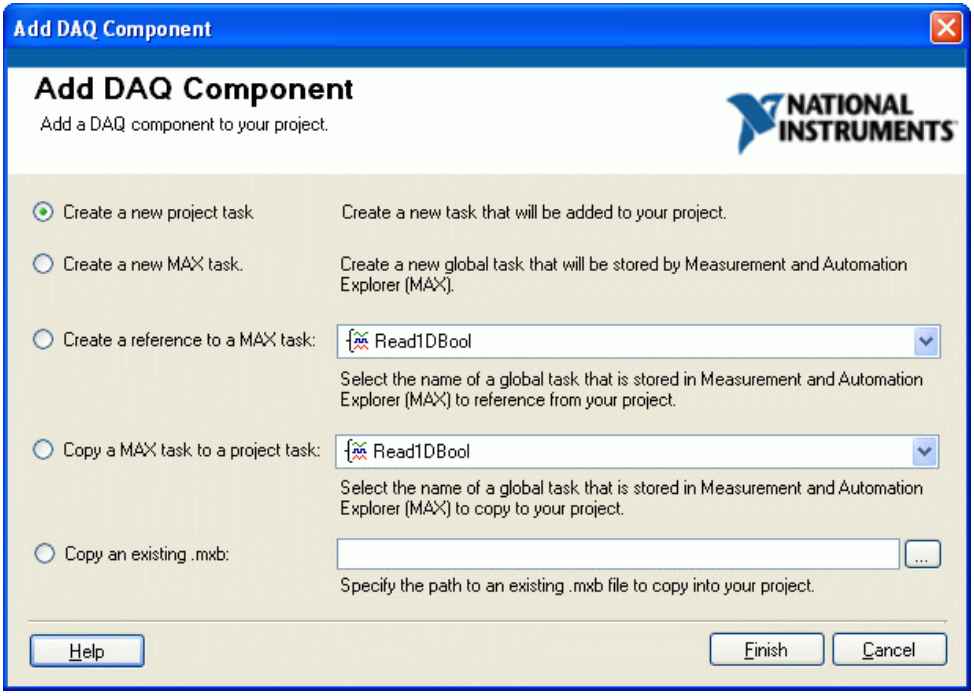
To set up the project

1. Open Visual Studio from **Start»All Programs»Microsoft Visual Studio 2005»Microsoft Visual Studio 2005** or **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, 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**.



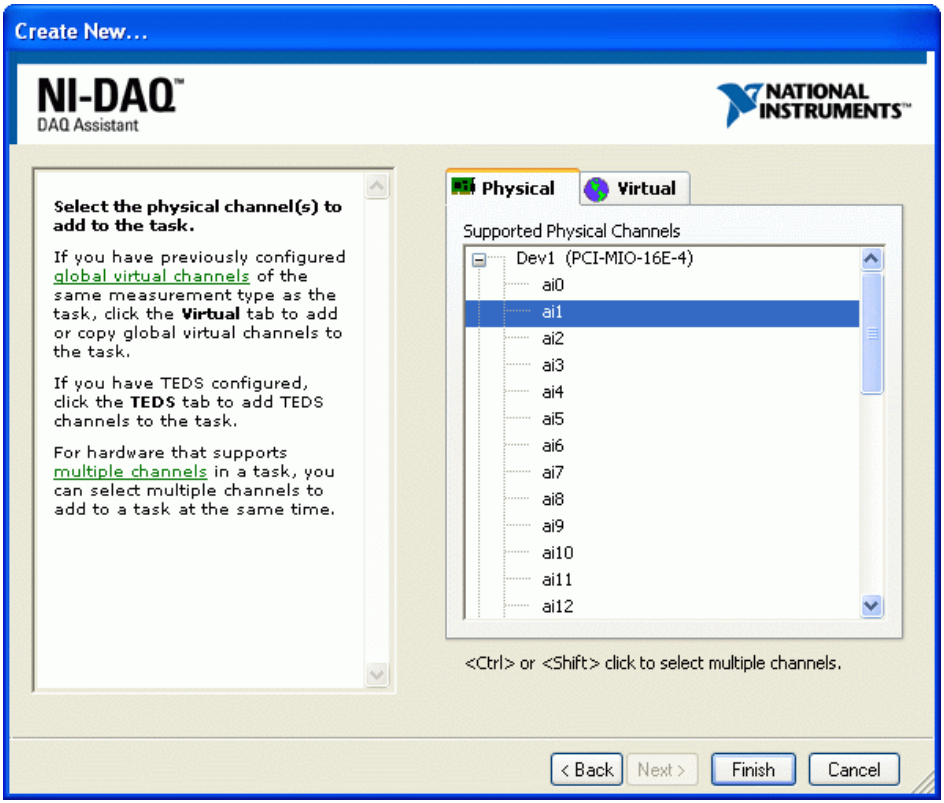
The Measurement Studio DAQ Application Wizard automatically sets up your data acquisition project and launches the DAQ Assistant.

To configure your task

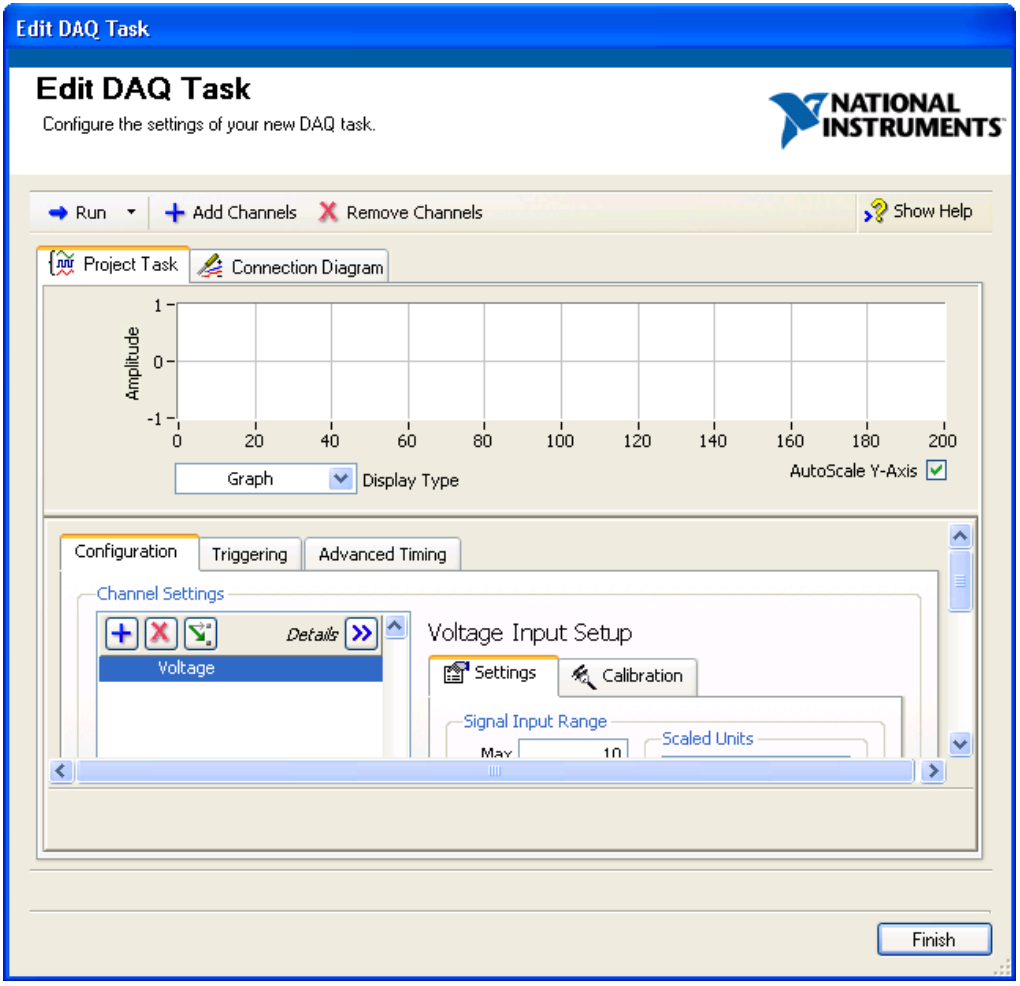
1. In the Create New dialog box of the DAQ Assistant, you can begin to interactively define your DAQ task. Select **Acquire Signals**, and then **Analog Input** as the measurement type for your task.
2. Next, select **Voltage**.

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**.

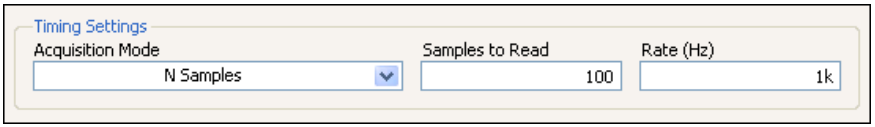
Note You can also use a simulated device in this walkthrough. For more information, refer to *Creating NI-DAQmx Simulated Devices* in the *Measurement & Automation Explorer Help for NI-DAQmx*.



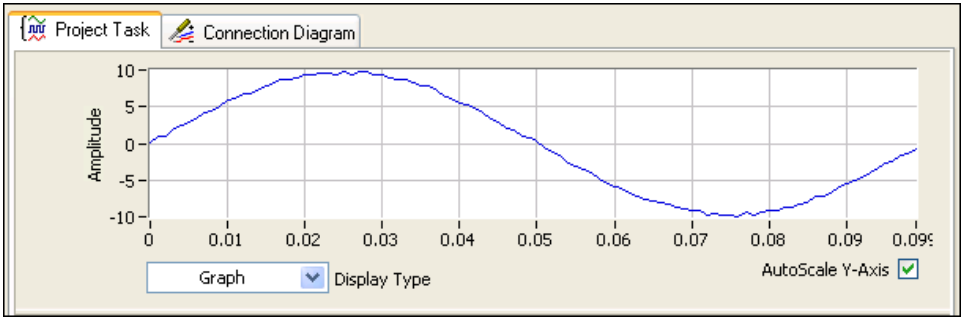
4. In the Edit DAQ Task dialog box, you can edit the configuration of your DAQ task. If the embedded DAQ Assistant help is not open by default, click the **Show Help** button in the upper-right corner of the window to display the help.



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



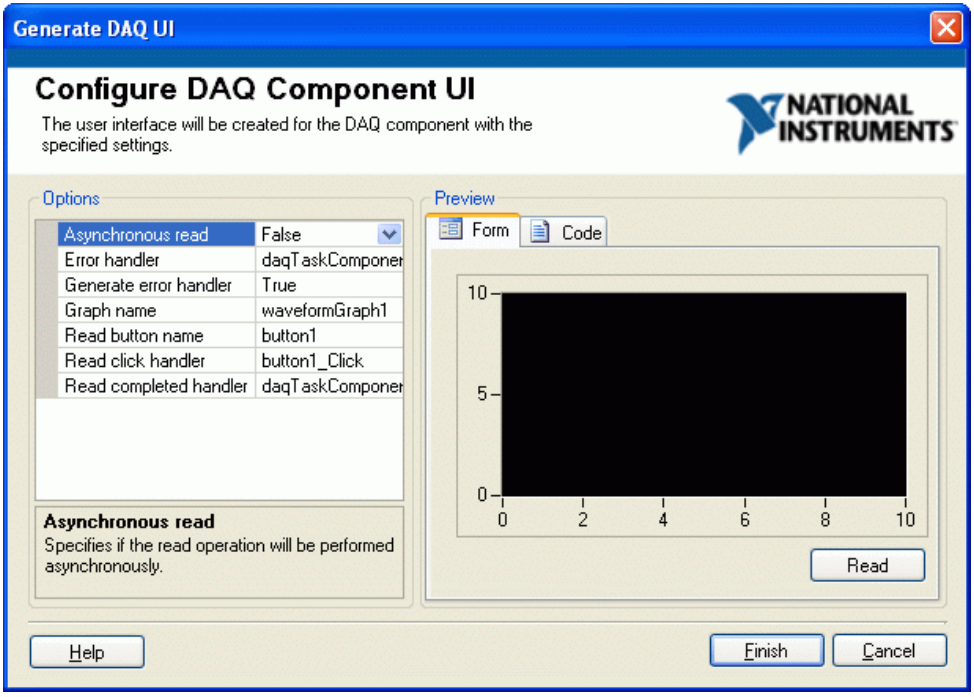
6. Next, click the **Run** button in the toolbar near the top of the Edit DAQ Task dialog box. The test runs automatically. You can run the test in the DAQ Assistant to test the task and make sure you connected the signal properly. If necessary, you can modify the settings before any code is generated.



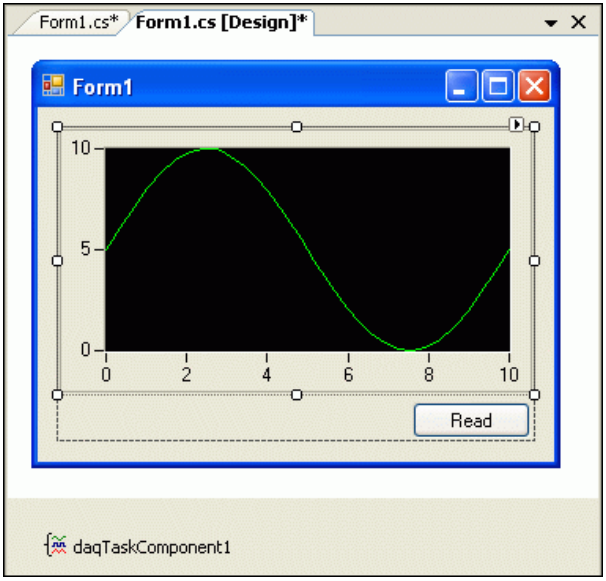
7. Click the **Finish** button in the Edit DAQ Task dialog box to complete the configuration of your DAQ task and to launch the Configure DAQ Component UI wizard.

To create a custom user interface for the task

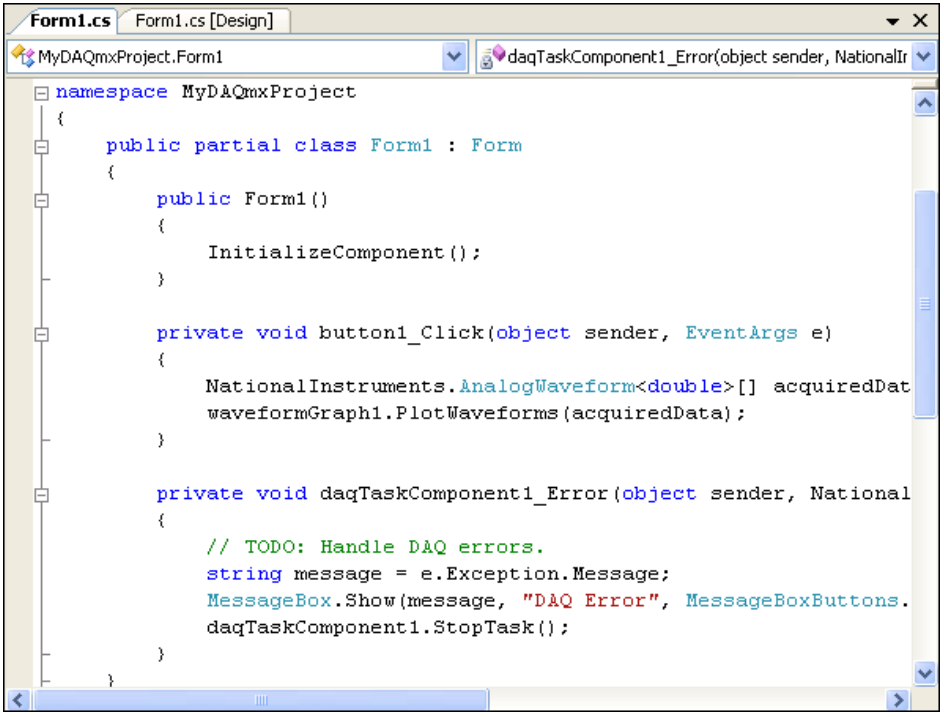
1. In the Configure DAQ Component UI wizard, you can customize and preview a user interface and code for your task.



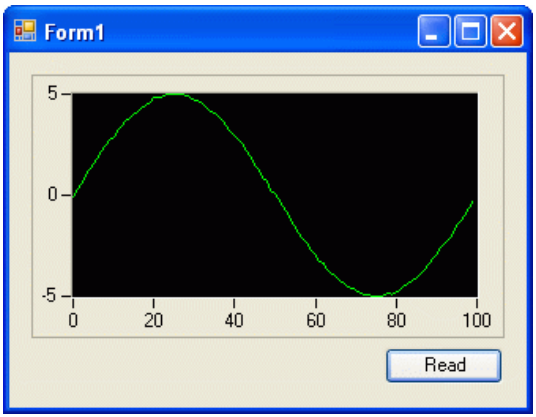
2. Click **Finish** to generate the task user interface in your project form.



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.*

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

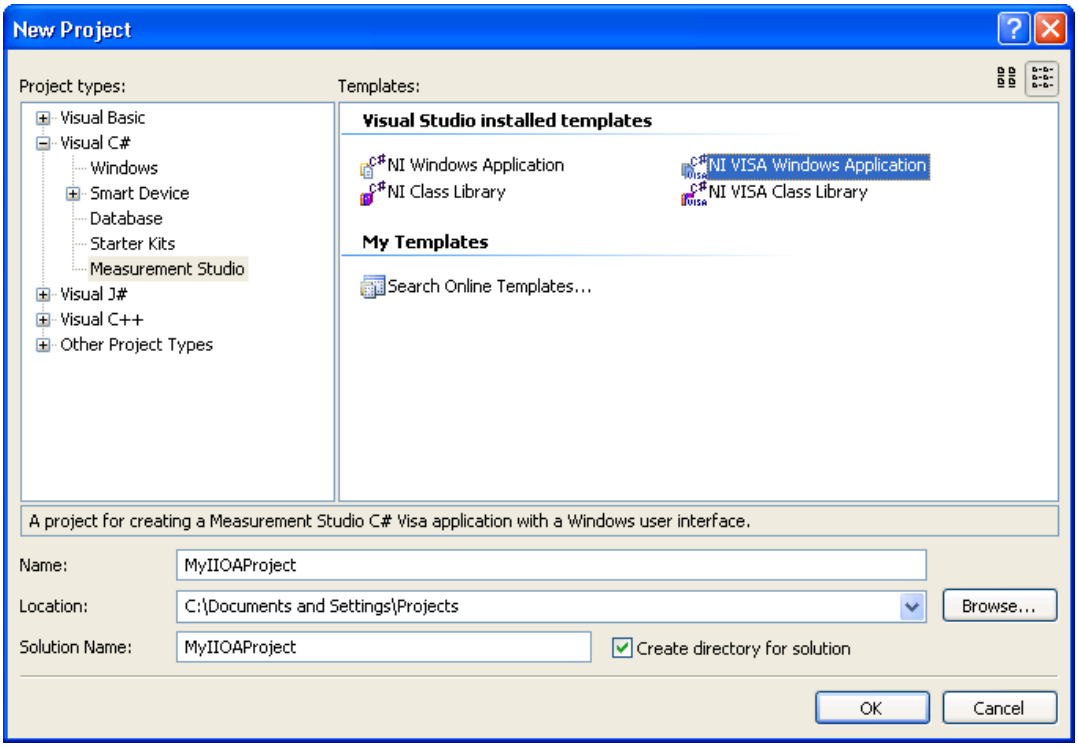
The following components are required to complete this walkthrough:

- Visual Studio 2005 or Visual Studio 2008
- Measurement Studio 8.0.1 or later (Professional or Enterprise package) for Visual Studio 2005 or Measurement Studio 8.5 or later (Professional or Enterprise package) for Visual Studio 2008
- National Instruments Device Driver CD
- 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 2005»Microsoft Visual Studio 2005** or **Start»All Programs»Microsoft Visual Studio 2008»Microsoft Visual Studio 2008**.
2. Select **File»New»Project**. The New Project dialog box launches.



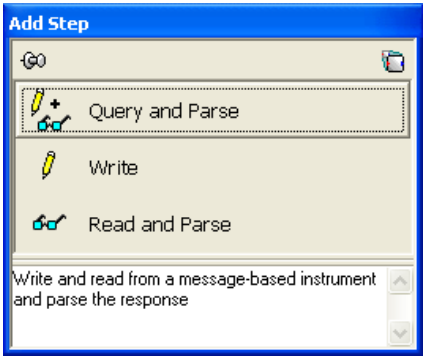
3. In the Project Types pane, select **Measurement Studio** under Visual C# or Visual Basic, depending on which language you want to create the project in. This walkthrough refers to Visual C#, but you can follow the same process if you use Visual Basic .NET.
4. In the Templates pane, select **NI VISA 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

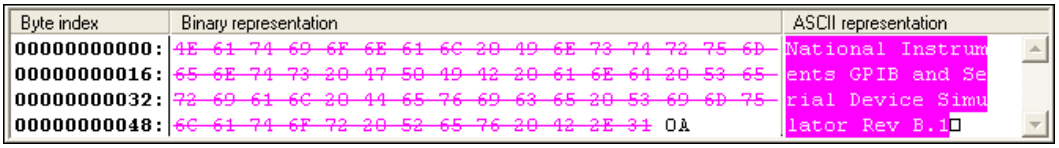
Note This walkthrough was created using the NI Instrument Simulator. Any identification information or sample code generated for this device will be different depending on the instrument actually used.

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.

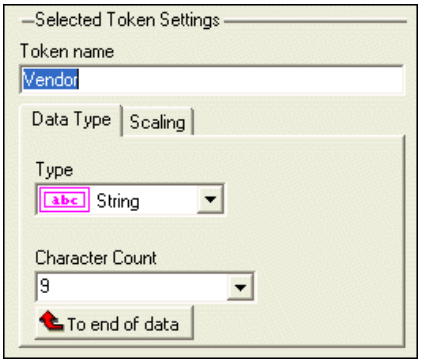


3. Enter the command `*idn?` and click **Run this step**. The `*idn?` command is a standard instrument command for querying an instrument's identification information. If your instrument does not support the `*idn?` command, refer to the documentation for the instrument for more information about the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

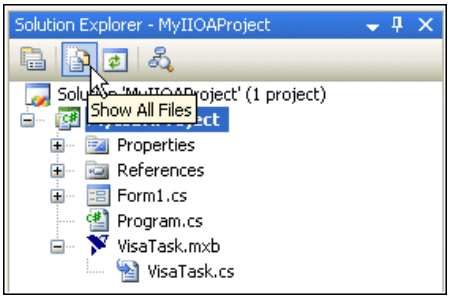


6. In the **Token name** text box, enter `Vendor` to rename the first token. You use this name to reference the token in your application. Ensure that the data type of the token is **String**. You specify the data type of the token using the **Type** drop-down list on the **Data Type** tab.



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.

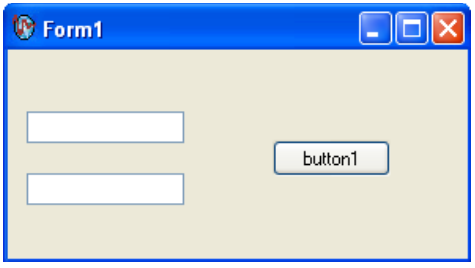
9. Select **View»Solution Explorer** to display the Solution Explorer.
10. Click the **Show All Files** icon and expand the `VisaTask.mxb` node.



11. Double-click the `VisaNSTask1` 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 the `Form1` file 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:



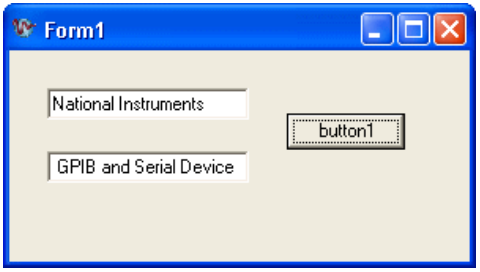
6. Double-click the **Button** control to display the `Form1` code, with the cursor inside the click event handler of the button control.

7. Add the following code to display the vendor and model name of your instrument in the text boxes.

```
[VB.NET]
' 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. Build and 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.

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 resources at ni.com/support include the following:
  - **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.

Glossary

A

ActiveX	Set of Microsoft technologies for reusable software components. Formerly called OLE.
ActiveX control	Reusable software component that adds functionality to any ActiveX control container through exposed properties, methods, and events. The Measurement Studio data acquisition, user interface, and analysis controls are examples of ActiveX controls.
ActiveX control container	Development environment that fully supports ActiveX controls and integrates them into its own environment using COM. An ActiveX control container enables you to specify how ActiveX controls interact with the environment through environment properties. Visual Basic is an example of an ActiveX control container.
analog I/O	Reading or writing data in continuously variable physical quantities, such as voltage or current.
annotate	Adding text, arrows, or shapes to describe or highlight a point or region on a graph.
ANSI C	C programming language defined by the American National Standards Institute.
API	Application Programming Interface. A specification of software functions and their input and return parameters.
array control	An array of Measurement Studio user interface controls that behave as a single unit.
assembly	A collection of one or more files that are versioned and deployed as a unit. An assembly is the primary building block of a .NET Framework application. All managed types and resources are contained within an assembly and are marked either as accessible only within the assembly or as accessible from code in other assemblies.
asynchronous	Function that begins an operation and returns control to the program prior to the completion or termination of the operation.

B

button — A control used to input or display Boolean information or to initiate an action in a program.

C

channel — 1. Physical—a terminal or pin at which you can measure or generate an analog or digital signal. A single physical channel can include more than one terminal, as in the case of a differential analog input channel or a digital port of eight lines. The name used for a counter physical channel is an exception because that physical channel name is not the name of the terminal where the counter measures or generates the digital signal.

2. Virtual—a collection of property settings that can include a name, a physical channel, input terminal connections, the type of measurement or generation, and scaling information. You can define NI-DAQmx virtual channels outside a task (global) or inside a task (local). Configuring virtual channels is optional in Traditional NI-DAQ and earlier versions, but is integral to every measurement you take in NI-DAQmx. In Traditional NI-DAQ, you configure virtual channels in MAX. In NI-DAQmx, you can configure virtual channels in either MAX or in a program, and you can configure channels as part of a task or separately.

3. Switch—a switch channel represents any connection point on a switch. It may be made up of one or more signal wires (commonly one, two, or four), depending on the switch topology. A virtual channel cannot be created with a switch channel. Switch channels may be used only in the NI-DAQmx Switch functions and VIs.

chart — To append new data points to the end of an existing plot over time.

client callback — In Web Forms, page calls back to the server without fully posting back. Callbacks are asynchronous and are accomplished with XML-HTTP. Client callbacks do not include postback data, and they do not force the page to refresh. Client callbacks do require a browser that supports the XML-HTTP protocol.

CodeBuilder — LabWindows/CVI feature that creates code based on a `.uir` file to connect your GUI to the rest of your program. This code can be compiled and run as soon as it is created.

coercion	Automatic conversion that Measurement Studio controls perform to change the numeric representation of a data element.
COM	Component Object Model. Microsoft specification for architecting and developing reusable software components.
complex graph	A control that displays a `ComplexDouble` data type; the `ComplexDouble` data type represents a complex number of type `Double` that is composed of a real part and an imaginary part.
context-sensitive help	Help for dialog boxes, the controls in dialog boxes, and keywords in source code that you can access with the key or a Help button, or by clicking the link that appears in the Dynamic Help window in Visual Studio.
control	1. ActiveX control. *See also* ActiveX control. 2. Object for entering, displaying, or manipulating data on a user interface.
counter/timer I/O	Reading or writing data based on high-precision timing through a counter or timer. By combining a counter with a highly accurate clock, you can create a wide variety of timing and counting applications, such as monitoring and analyzing digital waveforms and generating complex square waves.
cursor	Flashing rectangle that shows where you may enter text on the screen. If you have a mouse installed, there is a rectangular mouse cursor, or pointer.
cursor label	Text object used to display X and Y coordinates that a cursor crosshair points to on a graph.

D

DAQ	Data acquisition. Process of acquiring data, typically from A/D or digital input plug-in boards.
DAQ Assistant	A graphical interface for configuring measurement tasks, channels, and scales.
DAQ device	A device that acquires or generates data and can contain multiple channels and conversion devices. DAQ devices include plug-in devices, PCMCIA cards, and DAQPad devices, which connect to a computer USB or 1394 (FireWire®) port. SCXI modules are considered DAQ devices.

DataSocket	Technology that simplifies live data exchange between applications and HTTP, FTP, OPC, logos (Lookout objects) and file servers over the Internet. It provides one common API to a number of different communication protocols.
device	An instrument or controller you can access as a single entity that controls or monitors real-world I/O points. A device is often connected to a host computer through some type of communication network. *See also* DAQ device and measurement device.
digital I/O	Reading or writing digital representations of data in discrete units (the binary digits 1 and 0). Digital information is either on or off.
digital waveform graph	A control that displays `DigitalWaveform` data on a Windows Forms or Web Forms user interface; the `DigitalWaveform` data type represents a set of digital states that are grouped by samples or signals.
distribution	Ability to install programs you develop with Measurement Studio to others working on different computers.
DLL	Dynamic Link Library. A library of functions that link to a program and load at run time rather than being compiled into the program. Loading libraries only when they are needed saves memory in software applications.
DMM	Digital Multimeter. A common measurement instrument that measures resistance, current, and voltage in a wide variety of applications.
downlevel browser	Previous generation Web browser with limited client interaction. *See also* uplevel browser.
driver	Software that controls a specific hardware device, such as a data acquisition board or GPIB interface board. *See also* instrument driver.
DSTP	DataSocket Transfer Protocol. Protocol based on TCP/IP to exchange data directly between two applications using DataSocket clients. Data is passed through a DataSocket Server between the applications.

E

Ethernet	Standard connection type for networks, where computers are connected by coaxial or twisted-pair cable.
event	Object-generated response to some action or change in state, such as a mouse click or a completed acquisition. The event calls an event procedure that processes the event.
executable	Program file with a `.exe` extension that you can run independently of the development environment in which it was created.

F

form	Window or area on the screen on which you place controls and indicators to create the user interface for your program.
front panel	Interactive user interface of a virtual instrument. Modeled after the front panel of physical instruments, it is composed of switches, slides, meters, graphs, charts, gauges, LEDs, and other controls and indicators.
FTP	File Transfer Protocol. Protocol based on TCP/IP to exchange files between computers.

G

gauge	A control used to input or display numerical data.
GPIB	General Purpose Interface Bus. The standard bus used for controlling electronic instruments with a computer. Also called IEEE 488 bus because it is defined by ANSI/IEEE Standards 488-1978, 488.1-1987, and 488.2-1987.
graph	A 2D or 3D display of one or more plots.

H

HTTP	HyperText Transfer Protocol. Protocol based on TCP/IP, which is used to download Web pages from an HTTP server to a Web browser.

I

IEEE 488	Shortened notation for ANSI/IEEE Standards 488-1978, 488.1-1987, and 488.2-1987. *See also* GPIB.
IMAQ Vision	National Instruments image acquisition and analysis software that you can use to acquire images from National Instruments image acquisition (IMAQ) boards, display them in your program, perform interactive viewer operations, and analyze the images to extract information.
indicator	A control in read-only mode.
installer	Software program that copies program, system, and other necessary files to computers.
instrument driver	Library of functions to control and use one specific physical instrument. Also a set of functions that adds specific functionality to an application.
Instrument I/O Assistant	Assists in writing code to communicate with devices such as serial, Ethernet, or GPIB instruments. The Instrument I/O Assistant provides a user interface within the Visual Studio environment. You use the Instrument I/O Assistant to interactively write commands to a device, read data that the device returns, and specify how to parse the response.
interface	Connection between one or more of the following: hardware, software, and the user. For example, hardware interfaces connect two other pieces of hardware.
IVI	Interchangeable Virtual Instruments. A technology involving standard programming interfaces for classes of instruments, such as oscilloscopes, DMMs, and function generators, that results in hardware-independent instrument drivers. The IVI standard programming interfaces have been defined by the IVI Foundation, an industry consortium. Refer to `www.ivifoundation.org` for more information.

K

knob	A control used to input or display numerical data.

L

LabVIEW	Laboratory Virtual Instrument Engineering Workbench. Graphical development environment used for developing test and measurement applications.
LabWindows/CVI	ANSI C development environment for building test and measurement applications.
LED	Light-Emitting Diode. An indicator that emits a light when current passes through it. For example, an LED shows if your computer or printer is turned on.
legend	A control that displays symbols and descriptions for a specific set of elements of another object, such as the plots or cursors of a graph.

M

matrix	A rectangular array of numbers or mathematical elements that represent the coefficients in a system of linear equations.
MB	Megabytes of memory.
Measurement & Automation Explorer (MAX)	National Instruments tool for configuring your National Instruments hardware and driver software; executing system diagnostics; adding new devices, interfaces, and virtual channels; and viewing devices and instruments connected to your system.
measurement device	DAQ devices such as the E Series multifunction I/O (MIO) devices, SCXI signal conditioning modules, and switch modules.
Measurement Studio	National Instruments software that includes tools to build measurement applications in Visual Basic .NET, Visual C#, and Visual C++.
meter	A control used to input or display numerical data.
method	Function that performs a specific action on or with an object. The operation of the method often depends on the values of the object properties.
MFC	Microsoft Foundation Class. A framework for programming in Microsoft Windows, MFC provides code for managing windows, menus, and dialog boxes; performing basic input/output; storing collections of data objects; and more.

N

NI-488.2	Driver-level software to control and communicate with National Instruments GPIB hardware.
NI-DAQ	Driver-level software to control and communicate with DAQ hardware. NI-DAQ is an extensive library of VIs and functions you can call from an application development environment (ADE) to program all the features of an NI measurement device, such as configuring, acquiring and generating data from, and sending data to the device.
NI-DAQmx	The latest NI-DAQ driver with new VIs, functions, and development tools for controlling measurement devices. The advantages of NI-DAQmx over earlier versions of NI-DAQ include the DAQ Assistant for configuring channels and measurement tasks for your device for use in LabVIEW, LabWindows/CVI, and Measurement Studio; increased performance such as faster single-point analog I/O; and a simpler API for creating DAQ applications using fewer functions and VIs than earlier versions of NI-DAQ.
NI-IMAQ	Driver-level software to control and communicate with National Instruments image acquisition hardware.
numeric edit	A control used to display and edit numeric values.

O

OCX	OLE Control eXtension. Another name for ActiveX controls, reflected by the `.ocx` file extension of ActiveX control files.
OLE	Object Linking and Embedding. *See also* ActiveX.
OPC	OLE for Process Control. An industry standard based on ActiveX and COM technologies that enables you to create a single client application that can communicate with disparate devices. Refer to `www.opcfoundation.org` for more information.
oscilloscope	Measurement instrument widely used in high-speed testing applications, such as telecommunication physical layer testing, video testing, and high-speed digital design verification.

P

PCI	Peripheral Component Interconnect. High-performance expansion bus architecture commonly found in PCs.
PID	Proportional-Integral-Derivative. A three-term control mechanism combining proportional, integral, and derivative control. You might use a PID algorithm to control processes such as heating and cooling systems, fluid level monitoring, flow control, and pressure control.
plot	1. Trace (data line) on a graph representing the data in one row or column of an array. 2. To display a new set of data while deleting any previous data on the graph.
point	Structure that contains two 16-bit integers that represent horizontal and vertical coordinates.
postback	The process in which a Web page sends data back to the same page on the server.
property	Attribute that defines the appearance or state of an object. The property can be a specific value or another object with its own properties and methods. For example, a value property is the color (property) of a plot (object), while an object property is a specific Y axis (property) on a graph (object). The Y axis itself is another object with properties, such as minimum and maximum values.
property editor	A control used to configure properties for Windows Forms controls at run time.
property pages	Window or dialog box that displays current configuration information and allows users to modify the configuration.
PXI	PCI eXtensions for Instrumentation. Rugged, open platform for modular instrumentation with specialized mechanical, electrical, and software features. Refer to `www.pxisa.org` for more information.

R

range	Region between the limits within which a quantity is measured, received, or transmitted. The range is expressed by stating the lower and upper range values.

S

scalar	Number that a point on a scale can represent. The number is a single value as opposed to an array.
scale	Part of graph, chart, and some numeric controls and indicators that contains a series of marks or points at known intervals to denote units of measure.
scatter graph	A control that displays two-dimensional data on a Windows Forms or Web Forms user interface; displays a graph of X and Y data pairs.
scope	*See* oscilloscope.
serial	Standard serial bus on a computer used to communicate with instruments. Also known as RS-232.
slide	A control used to input or display numerical data.
slider	Moveable part of a slide control.
smart tag	A glyph attached to a Measurement Studio control or component that exposes commonly performed tasks.
switch	A control used to receive and control Boolean input in an application user interface.
synchronous	Property or operation that begins and returns control to the program only when the operation is complete.

T

tank	A control used to input or display numerical data.
task	NI-DAQmx—a set of channels and the channel configurations, timing, and triggering, and other details that define a measurement or generation you want to perform.

TCP/IP	Transmission Control Protocol/Internet Protocol. A standard format for transferring data in packets from one computer to another. The two parts of TCP/IP are TCP, which deals with the construction of data pockets, and IP, which routes them from computer to computer.
TestStand	Ready-to-run test executive from National Instruments for organizing, controlling, and executing your automated prototype, validation, or manufacturing test systems.
thermometer	A control used to input or display numerical data.

U

UI	User Interface.
uplevel browser	Recent generation Web browser that supports rich client interaction and functionality. *See also* downlevel browser.

V

vector	1D array.
virtual instrument (VI)	Program in Measurement Studio that models the appearance and function of a physical instrument.
VISA	Driver-software architecture developed by National Instruments to unify instrumentation software for serial, GPIB, and VXI instruments or controllers. It has been accepted as a standard for VXI by the VXI*plug&play* Systems Alliance.
VXI	VME eXtension for Instrumentation. Instrumentation architecture and bus based on the VME standard. Used in high-end test applications.

W

waveform graph	A control that displays two-dimensional data on a Windows Forms or Web Forms user interface; displays data that is uniformly spaced in one dimension.

Index

K

L

M

N

O

P

R

S

T

U

V

W