

GoDiagram Express Introduction

Copyright © 2002-2008 Northwoods Software Corporation

GoDiagram™ Express for Microsoft® .NET Windows Forms (“Go”) is a .NET class library containing a set of Windows Forms controls for easily building interactive diagrams in .NET-based applications.

Go is intended for building .NET applications using Windows Forms.

Installation

The installation kit is a Windows Installer file. It serves as the full binary product kit for GoDiagram Express.

Before you install Go, you should already have installed version 2.0 of the .NET Framework SDK.

The files, by default, are placed under your “My Documents” folder.

GoDiagram Express Files

Go consists of three assemblies:

- **Northwoods.GoExpress.dll**, holding the **Northwoods.Go** namespace
- **Northwoods.GoExpress.Xml.dll**, holding the **Northwoods.Go.Xml** namespace
- **Northwoods.GoExpress.Svg.dll**, holding the **Northwoods.Go.Svg** namespace

The assemblies are in the **lib** subdirectory of the Go installation. They only depend on the Microsoft .NET System, Windows Forms, and Drawing assemblies. They do not include any unmanaged code and do not require any particular permission beyond what any typical Windows Forms application would need.

Detailed documentation on the types in these libraries is provided in the **GoExpress.chm** compiled HTML help file. This file, along with other documentation, is in the **docs** subdirectory of the Go installation.

It also places some example code in the **Samples** and **SamplesVB** subdirectories. Both sample executable applications and the sample application sources are included. You can rebuild the sample applications in two different manners. You can run the **Samples\BuildSamples.bat** command file in a Visual Studio® Command Prompt window, after first editing that command file to make sure its directory paths are appropriate for your installation. This will recompile all of the sample applications and put the new EXEs in the installation root directory. The **SamplesVB\BuildSamplesVB.bat** will recompile the samples coded in VB.NET. Or you can open the project files in Visual Studio and compile and debug them individually.

Initial Experiences

If you haven’t already run the sample applications, just to get a feel for what Go can do, please try them. Reading the source code for the applications will really help you understand how easily you can implement different kinds of features. Remember that these are sample applications. Sometimes functionality is implemented just for the sake of demonstration—no real application would want to have that combination of features, or so many different ways to achieve the same kind of functionality.

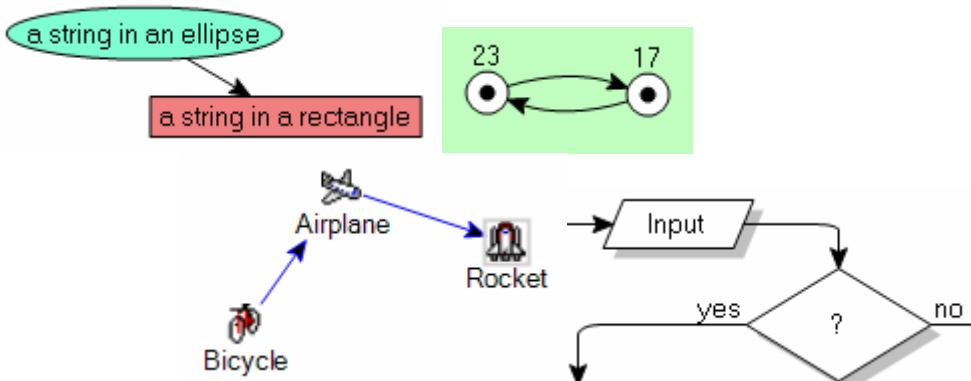
If you have certain features you know you want to implement, but are not sure how to do so, it might help to read the Frequently Asked Questions (FAQ) document, **GoExpressFAQ.chm**, in the **docs** subdirectory. Another source of inspiration can be the GoDiagram forum at <http://www.nwoods.com/forum>.

It might also help to read the entire User Guide, **GoExpressUserGuide.doc**, because it discusses much of the programming model embodied in Go. If you don't have that much time, at least read the *Go Concepts* chapter in the User Guide.

Differences between GoDiagram Express and GoDiagram Win

Although both products share much of the same design and implementation, GoDiagram Express is noticeably simpler and smaller than GoDiagram Win.

You'll want to use GoDiagram Express if you just need to build diagrams containing **GoBasicNodes**, **GoIconicNodes**, or **GoTextNodes**, such as the following screenshots demonstrate:



In GoDiagram Express you can inherit from these node classes to override methods or add your own methods or properties. If you need additional kinds of predefined nodes, or if you want to construct your own kinds of nodes, you'll need GoDiagram Win. Examples of the additional classes in GoDiagram Win include **GoGeneralNode**, **GoMultiTextNode**, **GoSubGraph**, and **GoBoxNode**.

Some other features that are only available in GoDiagram Win:

- **GoControls** to host Windows Forms **Controls** as **GoObjects**
- ability to deploy in reduced-trust permission environments, such as when hosted by Internet Explorer
- automatic layout classes (**GoLayout**)
- instrument classes (**GoInstruments**)
- **AvoidsNodes** property on **GoLink**
- **RoutingTime** property on **GoDocument**
- **GoButton** class and **GoToolAction** tool
- reshaping handles for some of the **GoShape** classes

Of course, an advantage of GoDiagram Express is that it is simpler and smaller than GoDiagram Win. Nevertheless, we have tried to put almost all the commonly-used features into the Express version, such as:

- the document-view-controller architecture, for additional options in displaying and editing diagrams, including the use of **GoOverview** and **GoPalette**, and the ability to organize objects into layers that can be seen (or not) in different orders by different views

- the use of tools, for greater flexibility in defining the interactive behavior you need to make your users productive
- the notion of ports as connection objects inside nodes, connected by links, both abstractly (**IGoNode**, **IGoPort**, **IGoLink**) and concretely (**GoBasicNode**, **GoIconicNode**, **GoTextNode**, **GoPort**, **GoLink**, **GoLabeledLink**)
- the customization of **GoLinks** and of the parts of the node classes, for getting both the appearances and the behavior you want for your application. You can define your own classes inheriting from one of the link or node or shape classes, or from **GoDocument** or **GoView**, to override the specialized behavior not already controllable by properties.
- the programmer-extensible undo/redo architecture, so that you can easily keep your application-specific state in synch with the user's undo and redo actions

GoDiagram Win is completely compatible with GoDiagram Express. If you upgrade from GoDiagram Express to GoDiagram Win, you just need to change the assembly references in your project and recompile. No source code changes are required.

Customizing Visual Studio

If you are using Visual Studio, you'll want to customize your Toolbox to include the three controls provided by the **Northwoods.GoExpress.dll** assembly.

1. Start up Visual Studio
2. View the Toolbox, if it isn't already visible.
3. Open up the tab that you want to hold the Go controls. You may want to create a new tab, or you may want to use an existing tab of Windows Forms controls.
4. Context click (right-mouse click) in the desired toolbox tab window. Choose the "Add/Remove Items" or "Choose Items..." context menu command. The Toolbox customization dialog will appear.
5. Select the ".NET Framework Components" tab.
6. Scroll down until you find the **GoView**, **GoPalette**, and **GoOverview** controls, in the Go assembly. If you do not see these controls, you may need to click the "Browse..." button to open the assembly in the **lib** subdirectory of the Go installation. Make sure all three controls have check marks by them.
7. Click OK for this dialog. The three controls should appear in your toolbox.

You can now drag any of the controls onto your Form that you are designing. The Properties window will let you specify many of the properties and events to customize the appearance and behavior of the selected view.

Creating a Standard MDI Windows Forms Application

It is very easy to use Visual Studio and its Form Designer to create a new single document interface application that uses Go. However, when you want to create a "standard" multiple document interface application, there are a lot of details that you must implement.

We have already done much of the work for you, in the form of the ProtoApp sample application. The ProtoApp sample is a complete, runnable application, but you'll want to customize it.

You can use this project to get started on your own application.

1. Depending on whether you want to code in VB.NET or C#, copy the appropriate directory, **samples\ProtoApp** or **samplesVB\ProtoApp** to your own directory.
2. Replace "ProtoApp" with your own project name in all of the files, and then edit the code to suit your needs.

3. Search for “TODO” to find some of the places you’ll want to customize.

You might consider starting off with some of the other sample applications, if they are closer to what you are looking for. The organization chart, state diagram, and flow chart editing programs are popular starting points. The class hierarchy browser is also representative of part of many Go applications, and is useful for learning Go in its own right.

Licensing

Any developer who programs using the Go application programming interface (API) must comply with the license agreement, `docs\ExpressBinaryLicense.rtf`. There are no fees for distributing or running applications that incorporate our Go products.

Once you purchase a license for developing applications using GoDiagram Express, we will send you instructions for how to build your application to avoid getting the “watermark” that you see in each view.

Support

Northwoods Software will only provide e-mail or forum support if you have purchased a support subscription.

However, you can read and search all of the documentation in this kit and the contents of the forum on our web site at any time.