

# GoDiagram<sup>TM</sup> Express for .NET Windows Forms, Version 2.3.1 Release Notes

Copyright © 2002-2005 Northwoods Software Corporation

This kit includes sets of assemblies and samples for building interactive diagram applications using Windows Forms in the full .NET Framework.

GoDiagram<sup>TM</sup> Express requires the Microsoft<sup>®</sup> .NET Framework 1.0 or 1.1 SDK. It also works with interactive development environments such as Visual Studio .NET 2002 or 2003 or 2005, or with Borland C#Builder. At the time of this writing GoDiagram Express assemblies appear to work with .NET 2.0 beta releases, although we have only done some casual testing.

This installation is initially licensed under the terms of the evaluation license agreement, `ExpressEvalLicense.rtf` in the `docs` subdirectory. Upon purchasing a valid binary development license for GoDiagram Express on a particular computer, and after installing it with the `LicenseManager` application, you may compile a license for `Northwoods.GoExpress.dll`. The binary development license agreement is `docs\ExpressBinaryLicense.rtf`. There are three redistributable assemblies:

- `GoDiagram Express: Northwoods.GoExpress.dll`
- `GoXml Express: Northwoods.GoExpress.Xml.dll`
- `GoSvg Express: Northwoods.GoExpress.Svg.dll`

Please read the **GoDiagram Express Introduction** document, `docs\GoExpressIntro.doc`. It includes a section about how to distribute your applications with the assemblies for which you have purchased a license.

You will find many answers to your questions in the **Frequently Asked Questions (FAQ)** list that is in `docs\GoExpressFAQ.doc`.

You might also want to search the **GoDiagram Forum** at <http://www.nwoods.com/forum>.

## Installation Directory Contents

- `\`
  - Release Notes (this README file)
- `\lib`
  - `Northwoods.GoExpress.dll`, containing the GoDiagram Express controls and classes
  - `Northwoods.GoExpress.Xml.dll`, the library for the customizable reading and writing of XML files
  - `Northwoods.GoExpress.Svg.dll`, the library for the customizable generation of SVG files
  - `*.xml`, XML documentation for code editor tooltips in Visual Studio .NET
- `\docs`
  - `GoExpressIntro.doc`, introduction to GoDiagram Express
  - `GoExpressFAQ.chm`, Frequently Asked Questions list
  - `GoExpressUserGuide.doc`, User Guide for GoDiagram Express
  - `GoExpress.chm`, API Reference Manual for GoDiagram Express
  - `ExpressEvalLicense.rtf`, evaluation license agreement file

- `ExpressBinaryLicense.rtf`, binary development license agreement file
- `.\bin`
  - `LicenseManager.exe`, to install developer's licenses for the GoDiagram products
- `.\Samples`, sample applications written in C#
  - `*.exe`, sample applications using GoDiagram Express
  - Sample source code, in subdirectories:
    - **BasicApp**, a simple editor using basic nodes
    - **Classier**, GoDiagram class hierarchy browser and property/method viewer
    - **Demo1**, GoDiagram demo application, built by modifying prototype app
    - **FamilyTree**, displays the relationships amongst some of the English Tudors
    - **FlowCharter**, a simple flow chart editor
    - **IconicApp**, a simple editor using nodes displaying icons
    - **MinimalApp**, a minimal GoDiagram application
    - **OrgCharter**, a simple organization chart editor
    - **ProtoApp**, prototype application
    - **StateCharter**, a simple state diagram editor
    - **TreeApp**, a two-dimensional tree that supports expand/collapse
    - **WebWalker**, shows references between HTML web pages
  - `BuildSamples.bat`, a command file to recompile the sample applications
- `.\SamplesVB`, the same GoDiagram Express samples translated from C# to VB.NET
  - `BuildSamplesVB.bat`, a command file to recompile the sample applications

## Summary of Significant New Features since 2.2

The complete list of changes and new features is included later in this document.

- added **GoView.BackgroundLayer** property, behind all document layers
- added **GoView** properties: **TopBar**, **RightBar**, **BottomBar**, **LeftBar**, **TopLeftCorner**, **TopRightCorner**, **BottomRightCorner**, **BottomLeftCorner**
- added **GoView.OnObjectEnterLeave** event (with corresponding methods and delegate), **GoView.DoObjectEnterLeave**, and **GoObject.OnEnterLeave** method; to simplify implementing mouse-over behaviors when entering and leaving document objects
- added **GoView.NewLinkPrototype** property, subsuming the **GoView.NewLinkClass** property
- added **GoView.NewGoLink** and **NewGoLabeledLink** properties as type-specific values for **NewLinkPrototype**, to simplify initializing new links without defining a **GoView.LinkCreated** event handler
- added **GoView.CursorName** property, for setting **GoView.Cursor** using a cursor name rather than a WinForms **Cursor**
- added **PickObjectsInRectangle** method in **GoView**, **GoDocument**, and **GoLayer**, now called by **GoView.SelectInRectangle**
- **GoDocument.StartTransaction**, **FinishTransaction**, **AbortTransaction**, **Undo**, and **Redo** now raise **Changed** events with corresponding hints, to make updating UI easier using event handlers instead of overriding methods, and to support saving changes to a database as each undo or redo occurs (if not using batch-style save and load)
- added **GoObject.GetCursorName** method, to specify a cursor for an object, for WebForms and for SVG, not just for WinForms
- added **GoToolCreating** tool, to let users create objects by background-dragging their bounds, given a prototype object
- Generated SVG now supports limited interaction: tooltips, cursors, custom

- panning/zooming, HREF links, selection by clicking or by rubber-band dragging
- **GoXmlWriter** methods have been generalized to either write XML to a **Stream** or produce an **XmlDocument**
- many other improvements to GoXml
- examples of customizing SVG generation in Demo1

## Upgrading from 2.2 to 2.3

You will need to make sure your projects now refer to the new Northwoods.GoExpress.dll version 2.3 assembly, not an older version.

You will need to change all of your licenses.licx files so that references to GoDiagram Express components of:

Version=2.2.2.0 (or the version you have been using until now)  
are replaced with:

Version=2.3.1.0

## Known Bugs and Misfeatures

- There are a number of Microsoft bugs that you may encounter:
  - License keys are only recognized when they are linked into an executable assembly (.EXE), not a library (.DLL); you need to have your executable's project refer to Northwoods.GoExpress and include the desired component references in the executable project's LICENSES.LICX file (you can leave them in your library that uses GoDiagram, but they are ineffective there)
  - SOAP serialization
  - text clipping
  - when setting properties on **Controls** created by **GoControls**, and the real control needs to be re-created
  - stopping Timers in .NET Framework 1.1; we have rewritten **GoView** as a work-around to avoid this problem
  - rendering within a **PrintPreviewDialog**

## Detailed List of Changes since 2.2.2

### GoView

- added **GoView.BackgroundLayer** property, behind all document layers
- added **GoView.RescaleWithCenter** method, to change the scale while keeping the center of the view at the same document point
- added **GoView** properties: **TopBar**, **RightBar**, **BottomBar**, **LeftBar**, **TopLeftCorner**, **TopRightCorner**, **BottomRightCorner**, **BottomLeftCorner**
- **GoView.CornerControl** is now just implemented as **BottomRightCorner**
- **GoView.VerticalScrollBar** is now just implemented as **RightBar** or **LeftBar**, whichever is a **VScrollBar**
- **GoView.HorizontalScrollBar** is now just implemented as **BottomBar** or **TopBar**, whichever is an **HScrollBar**

- added **GoView.ObjectEnterLeave** event, along with **GoObjectEnterLeaveEventHandler** and **GoObjectEnterLeaveEventArgs**, to simplify implementing mouse-enter and mouse-leave event handlers for document objects, and with a reference to the document object that the mouse had been over, to help handle cases with overlapping objects, especially child objects of **GoGroups**; this event is only raised for mouse-over events when the **GoToolManager** or **GoToolDragging** tools are running
- added **GoView.DoObjectEnterLeave**, to call **GoView.RaiseObjectEnterLeave**, and to call **GoObject.OnEnterLeave** on the document objects that the mouse had been over and that the mouse is now over
- added **GoView.NewLinkPrototype** property, subsuming the **GoView.NewLinkClass** property
- added **GoView.NewGoLink** and **NewGoLabeledLink** read-only properties as type-specific values for **NewLinkPrototype**, to simplify initializing new links without defining a **GoView.LinkCreated** event handler
- added **GoView.PrintsViewObjects** property (not Pocket), to easily allow **GoView.PrintView** to print view objects (such as selection handles) as well as document objects
- changed **GoLayer.AllowPrint** property to default to true for layers that belong to views
- added **GoView.CursorName** property, for setting **GoView.Cursor** using a cursor name rather than a WinForms **Cursor**
- added **GoView.ContextClickSingleSelection**, for easier access to the view's **GoToolContext.SingleSelection** property
- added **GoView.SelectInRectangleStyle** property, which controls behavior of **GoView.SelectInRectangle**
- changed **GoView.MatchesNodeLabel** to take **IGoLabeledPart** instead of **IGoLabeledNode**
- added **GoView.FindMouseTool(Type, bool)** to support finding a tool that is an instance of a subclass of a **Type**
- changed **GoView.GetShadowBrush** and **GetShadowPen** to take the shadowed object as a parameter
- added **GoView.ClipboardCopied** event, raised by **GoView.EditCopy** and **EditCut**, to make updating UI easier using event handlers instead of overriding methods
- added **GoView.GetBitmap** method, to match the same method for WebForms
- fixed setting **GoView.Document** so that the view no longer ignores the next mouse down if it didn't have focus
- fixed external drag-and-drop when **GoView.ExternalDragDropsOnEnter** is true, to occur in a single transaction rather than in two, so that undo/redo behaves more naturally

## GoPalette

- fixed bug in **GoPalette.LayoutItems** causing positioning to expand when **Orientation.Horizontal**

## GoDocument

- added **GoDocument.LastPartID** property, to allow for proper restoration of document state when loading a saved document.  
*You should change your persistence code to save this property and to restore it before adding objects to the document.*
- improved **GoDocument.EnsureUniquePartID** to make sure **GoDocument.LastPartID** is at least as large as the largest value of **IGoIdentifiablePart.PartID** used in the document
- added **IGoLabeledPart**, as supertype of **IGoLabeledNode**, to provide access to **IGoLabeledPart.Text** string only, so that objects can be found without requiring them to have **GoText Labels**; used by **GoDocument.FindNode**, by **GoView.SelectNextNode** and **MatchesNodeLabel**, and by **GoPalette** sorting
- fixed **GoDocument.FindNode** when searching subgraphs
- added **PickObjectsInRectangle** method in **GoView**, **GoDocument**, and **GoLayer**, now called by **GoView.SelectInRectangle**
- fixed **GoDocument.PickObjects** to return an empty collection instead of null when passed selectableOnly but **GoDocument.CanSelectObjects()** is false
- added **GoDocument.Initializing** property, set to true during undo or redo calls to **GoDocument.ChangeValue**
- **GoDocument.StartTransaction**, **FinishTransaction**, **AbortTransaction**, **Undo**, and **Redo** now raise **Changed** events with corresponding hints, to make updating UI easier using event handlers instead of overriding methods, and to support saving changes to a database as each undo or redo occurs (if not using batch-style save and load)

## GoUndoManager

- added **GoUndoManager.CommitCompoundEdit** method, to permit optimizing the edits that are saved, and to support saving changes to a database as each **GoDiagram** transaction finishes (if not using batch-style save and load)
- changed **GoUndoManager.EndTransaction** by adding "transaction name" parameter, so that both the transaction name and the presentation name can be passed as values to the **FinishedTransaction Changed** event
- changed some **GoUndoManager** properties to no longer be virtual

## GoObject

- **GoObject.Initializing** property now set to true during undo or redo calls to **GoObject.ChangeValue**
- added **GoObject.GetCursorName** method, to specify a cursor for an object, for WebForms and for SVG, not just for WinForms

- added **GoObject.OnEnterLeave** method, called by **GoView.DoObjectEnterLeave** to handling mouse-overs that enter and leave objects
- improved **GoObject** observer mechanism to also call **OnObservedChanged** when an observed object is added or removed from a layer
- added **GoObject.Copy** method, for convenience in copying a single object without calling **GoDocument.AddCopy**
- added **GoCopyDictionary.FinishDelayedCopies** and **CopyComplete** methods, for convenience when trying to copy objects without calling **GoDocument.CopyFromCollection**
- fixed **GoObject.AddSelectionHandles** not to create resize handles when **GoView.CanResizeObjects()** or **CanReshapeObjects()** is false

## GoGroup

- changed **GoGroup.ContainsPoint** and **GetNearestIntersectionPoint** to check child.**Visible** property instead of calling child.**CanView()** predicate, so some operations work properly on non-**Visible** groups
- added **GoGroup.PickableBackground** property, so that a user's mouse event in the whole rectangular area of a group can cause the group to be picked, even if there are no child objects at that mouse point
- added mapping names to **GoGroup** children: **FindName**, **FindChild**, String indexer; this allows you to refer to particular child objects by name as well as via a property.

## GoShapes, GoText, GoImage, GoControl et al.

- added static/shared method **GoShape.DrawPath**
- added **GoStrokeArrowheadStyle.Slash** and **.BackSlash** as new **GoStroke** arrowhead styles
- fixed **GoStroke.GetSegmentNearPoint** not to assume a minimum line width of 1.0f
- added **GoPolygon.GetSegmentNearPoint**, when you want to see if a point is near the edge of a polygon or poly-Bezier shape, and don't want to check for point containment by using **ContainsPoint**
- replaced **GoHandle.Cursor** property with **GoHandle.CursorName** property
- removed **GoHandle.GetCursorForHandle** method
- defined **GoHandle.GetCursorName** method instead of overriding **GoObject.OnMouseOver**
- fixed sizing text when printing text objects with **GoText.AutoResizes** false
- added **GoImage.ThrowsExceptions** property for controlling whether exceptions are ignored, to help debug failures loading and painting images
- added exception handler when painting image, to catch GDI+ errors when drawing image

## Nodes, Ports, and Links

- changed many properties on nodes to be virtual; a few that should not have been virtual are now not virtual
- improved **GoPort.GetLinkPointForPoint** to work with **.PortObject** when node contains the **PortObject** but is not yet part of a document
- added **GoLabeledLink.CalculateStroke** method, for convenience; this just calls **GoLabeledLink.RealLink.CalculateStroke()**.
- improved **GoLink.AddOrthoPoints** to avoid thin long loop for some cases where the ports are near to each other

## Tools

- added static/shared **GoTool.DragSize** property, to indicate how far a mouse-down-move-up needs to go to be considered a drag and not a click
- added **GoToolCreating** tool, to let users create objects by background-dragging their bounds, given a prototype object
- moved instance of **GoToolLinkingNew** tool from **GoView.MouseDownTools** list to **MouseMoveTools** list; to allow selection of ports, at the expense of requiring users to drag the mouse in order to start drawing a new link
- changed **GoToolLinking.StartRelink** second argument from **IGoPort** to boolean to indicate direction, to fix relinking a link that is not connected at either end to a port
- improved **GoToolLinking.PickPort** to ignore any port that is part of the link being relinked
- removed obsolete property **GoToolLinking.Orthogonal**
- added **GoToolManager.DoMouseOver** and **GoToolDragging.DoMouseOver**, to call **GoView.DoObjectEnterLeave** when current document object at the mouse point changes
- added **GoToolRubberBanding.Active** property
- improved **GoToolResizing** to call **GoToolResizing.DoResizing** with state of **GoInputState.Start** when tool is **Started**
- added **GoInputEventArgs.InputState** property, potentially usable by the **GoToolResizing** and **GoToolDragging** tools

## GoXml

**GoXmlWriter** methods have been generalized to either write XML to a **Stream** or produce an **XmlDocument**. Don't use the **XmlTextWriter** unless you also modify the **XmlDocument/XmlElement** if **XmlTextWriter** is null. Use the new **WriteAttrVal**, **WriteStartElement**, **WriteEndElement**, **WriteTextBody** methods instead, because these methods work with both **XmlTextWriter** and **XmlDocument**

- **GoXmlReader** is no longer abstract and now has public constructor

- changed **GoXmlReader.UseDOM** initial value from true to false
  - added **GoXmlReader.RootObject** property, to provide an **IList** or **IGoCollection** for **GoXmlReader.Consume** to **Add** objects to
  - renamed **GoXmlReader.CurrentNode** property to **ReaderNode**
  - renamed **GoXmlReader.ConstructObject** method to **ConsumeObject**
  - added **GoXmlReader.InvokeConsumeChild** and **InvokeConsumeObjectFinish** methods
  - added **GoXmlReader.ReadTextBody** and **ReadAttrVal** methods--these work with either **XmlTextReader** (stream) or with **XmlDocument** (DOM)
  - added **GoXmlReader.ObjectStack** property, to hold the stack of **Objects** that have been **Allocated** during the tree walk
  - reimplemented **GoXmlReader.ParentObject** as read-only property that refers to the top of the **ObjectStack**
  - added **GoXmlReader.GrandParentObject** property, that refers to the penultimate object on the **ObjectStack**
- 
- **GoXmlWriter** is no longer abstract and now has public constructor
  - added **GoXmlWriter.Generate()** method to produce **XmlDocument** (DOM) instead of writing to a **Stream**
  - added **GoXmlWriter.XmlDocument** public read-only property and protected **SetXmlDocument** method
  - added **GoXmlWriter.WriterElement** property, the current **XmlElement** being constructed/written
  - renamed **GoXmlWriter.GenerateRenderings** method to **GenerateObjects**
  - added **GoXmlWriter.NodesGeneratedFirst** property
  - renamed **GoXmlWriter.Define** method to **DefineObject**
  - renamed **GoXmlWriter.Render** method to **GenerateObject**
  - added **GoXmlWriter.InvokeGenerateElementFinish** method
  - added **GoXmlWriter** method: **DefineAndGenerateSharedObject**
  - added **GoXmlWriter** methods: **WriteStartElement**, **WriteEndElement**, **WriteTextBody**, **WriteAttrVal**, **SetWriterElementAttribute**
  - added **GoXmlWriter** methods: **GetNamespaceUri** and **SetNamespaceUri**
- 
- added **GoXmlReaderWriterBase** class, inherited by both **GoXmlReader** and **GoXmlWriter**, to share implementation of transformer management code, and to permit having a single static method that registers all transformers for both reader and writer
- 
- added **IGoXmlTransformer.ConsumeChild**, **ConsumeObjectFinish**, and **GenerateElementFinish** methods
  - added **GoXmlTransformer.ConsumeChild**, **ConsumeObjectFinish**, and **GenerateElementFinish** methods
  - added **GoXmlTransformer.IdAttributeUsedForSharedObjects** property
  - added **GoXmlTransformer.BodyConsumesChildElements** property
  - removed all the **GoXmlTransformer.BaseMMM** methods -- in your method overrides you should just call **base.MMM** instead
  - renamed **GoXmlTransformer.CurrentNode** property to **ReaderNode**
  - added **GoXmlTransformer.WriterElement** property
  - added **GoXmlTransformer** convenience methods: **WriteStartElement**, **WriteEndElement**, **WriteTextBody**, **ReadTextBody**

- replaced **GoXmlTransformer.StringRef** method with **WriteAttrRef** method
- replaced **GoXmlTransformer.StringVal** overloaded methods with **WriteAttrVal** overloaded methods
- added **GoXmlTransformer** methods for converting from a string to various types
- added **GoXmlTransformer.IsAttrPresent** method
- changed **GoXmlTransformer.TypeAttr** to call **Type.GetType** with case insensitive search
- changed **GoXmlTransformer.Allocate** to return an instance of the **GoXmlTransformer.TransformerType**, if it is a class

## GoSvg

Generated SVG now supports limited interaction: tooltips, cursors, custom panning/zooming, HREF links, selection by clicking or by rubber-band dragging.

- fixed SVG generation for **GoStrokes** with non-polygon **Style** arrowheads
- added **GoSvgWriter** constructors that produce **XmlDocument** instead of writing to a **Stream**
- removed **GoSvgWriter.FixedSizeDisplay** property; generated SVG assumes this old property is true
- added **GoSvgWriter.Scripting** and **ScriptFile** properties
- added **GoSvgWriter.PanAndZoomControls**, **ToolTips**, and **SupportsSelection** properties
- added **GoSvgWriter.ObjectsLimitedToDocExtent** property
- added **GoSvgWriter.GetSelectionId** and **GetHref** methods
- added **GoSvgWriter.RenderControlsAndForms**, **RenderPanAndZoomControls**, **RenderToolTip**, and **RenderRubberBand** methods
- added **GoSvgWriter.GenerateScript**, **GenerateBackgroundDecoration**, and **GenerateSelectionHandles** methods

## Support

Northwoods Software provides e-mail support during the 30-day evaluation period and for 30 days after purchase. If you purchase the optional support subscription, you receive e-mail support for a year after purchase plus all new versions that are released during that period.

For general sales and licensing questions, send e-mail to [GoSales@nwoods.com](mailto:GoSales@nwoods.com).